ARTIFICIAL INTELLIGENCE FOR ALL VS. ALL CONJUNCTION SCREENING

E. Stevenson[(1)], **V. Rodriguez-Fernandez**[(1)], **H. Urrutxua**[(2)], **V. Morand**[(3)], **and D. Camacho**[(1)]

[(1)]*School of Computer Systems Engineering, Universidad Politécnica de Madrid, Alan Turing Street, 28038 Madrid, Spain, Email: {emma.stevenson, victor.rfernandez, david.camacho}@upm.es*
[(2)]*European Institute for Aviation Training and Accreditation, Universidad Rey Juan Carlos, Camino del Molino 5, 28942 Fuenlabrada, Spain, Email: hodei.urrutxua@urjc.es*
[(3)]*CNES, 18 avenue Edouard Belin 31400 Toulouse France, Email: vincent.morand@cnes.fr*

ABSTRACT

This paper presents a proof of concept for the application of artificial intelligence (AI) to the problem of efficient, catalogue-wide conjunction screening. Framed as a machine learning classification task, an ensemble of tabular models were trained and deployed on a realistic *all vs. all* dataset, generated using the CNES BAS3E space surveillance simulation framework, and consisting of 170 million object pairs over a 7-day screening period. The approach was found to outperform classical filters such as the apogee-perigee filter and the Minimum Orbital Intersection Distance (MOID) in terms of screening capability, with the number of missed detections of the approach controlled by the operator. It was also found to be computationally efficient, thus demonstrating the capability of AI algorithms to cope and aid with the scales required for current and future operational *all vs. all* scenarios.

Keywords: Space Debris; Conjunction Assessment; Artificial Intelligence.

1. INTRODUCTION

In recent months, there have been several high-profile near-miss conjunction events [3], most notably of which, and of greatest concern, were those between two non-manoeuvrable space debris objects. To detect such a potentially catastrophic event, it is essential to search for conjunctions between all possible sets of catalogued objects, both active and debris. However, this scenario, so-called *all vs. all*, is extremely computationally expensive, with hundreds of millions of possible collision pairs to continually screen for conjunctions over the screening period. These computational challenges will be exacerbated by the future space environment, with increasing space traffic and improved observational capabilities resulting in a substantial increase in the number of catalogued objects [22] and, consequently, the number of possible conjunction pairs.

One potential approach to handling the computational burden required for current, and future, conjunction screening of the full space object population, is the application of Artificial Intelligence (AI). AI techniques such as machine learning (ML) and deep learning (DL) are promising in a wide variety of problems owing to their ability to process and exploit large datasets, infer hidden correlations and also reduce computational time during model prediction, all of which are very pertinent to the all vs. all problem. The application of AI to Space Traffic Management in general has recently emerged as a promising area of research [25], with possible uses stretching from offering support in reducing operator workload [12], to the classification of high-risk conjunction events [27], to the planning and implementation of collision avoidance manoeuvres [29].

In this work, we propose to apply these techniques to the problem of efficient large-scale conjunction screening, or more specifically, close conjunction detection in a catalogue-wide all vs. all scenario. As such, we do not propose to replace the entire conjunction assessment pipeline with an end-to-end black box AI system, but rather use it as an efficient first filter or initial screening to complement existing systems, as illustrated in Figure 1. This enables a significantly reduced set of high risk candidates to be passed to pre-existing operationally used, and trusted, numerical or analytical methods for performing further refinement and collision risk evaluation.

To the best of our knowledge, this is the first time AI techniques, or more specifically machine learning algorithms, have been applied in this way for the all vs. all problem, and in this paper we present a proof of concept of this new approach based on a realistic dataset of conjunction pairs generated using the CNES BAS3E space surveillance simulation framework [21]. The paper is structured as follows. In Section 2, we provide a description of the generation of the all vs. all dataset using the BAS3E framework and an input two-line element set (TLE) catalogue, on which our approach is based. Section 3 details the proposed AI approach for the problem of large-scale conjunction screening, which we phrase as a machine learning tabular classification task. In this section, particular attention is paid to the number of false negatives generated by the approach, cases in which high risk events

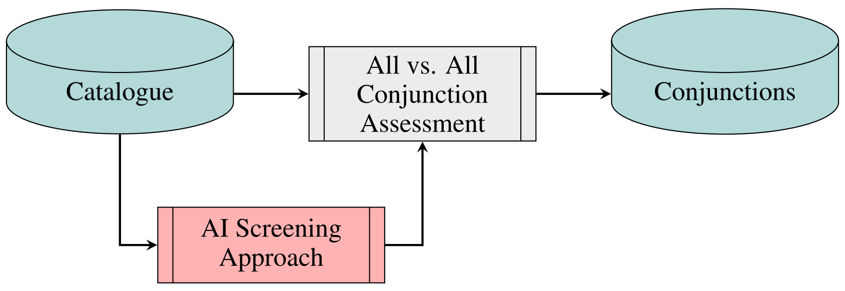


Figure 1: Illustration of AI screening concept to aid and complement existing operational all vs. all conjunction assessment.

are misdiagnosed as low risk and thus wrongly, and potentially critically, discarded by the screening process. In Section 4, we perform a comparison against two classical, non-AI filters (the apogee-perigee filter and Minimum Orbital Intersection Distance) in terms of both screening capability and computational expense, and demonstrate the capability of an AI algorithm to cope with the scales required for operational all vs. all scenarios. Finally, in Section 5 we discuss the conclusions of the paper, and outline avenues for future research.

2. ALL VS. ALL DATASET

2.1. BAS3E

BAS3E (**B**anc d'**A**nalyse et de **S**imulation d'un **S**ysteme de **S**urveillance de l'**E**space – Simulation and Analysis Bench for Space Surveillance System) is a CNES-owned tool developed in JAVA whose objective is to model and estimate the performances of space surveillance systems, both on ground and on board [21]. BAS3E capabilities are wide and include space object population orbit propagation, sensor modelling and scheduling, correlation, orbit determination and lately simulation of services such as the collision avoidance services.

BAS3E adopts a modular approach as depicted in Figure 2: each step of the computation, called stage, can be executed independently while results are stored in databases to be accessed from latter stages. From the beginning, BAS3E has been designed and developed for parallel computing in, for instance, a High Performance Computing (HPC) service. BAS3E is generally used on the CNES own HPC service, which is dedicated to scientific projects demanding great computing and processing capacity.

To generate the all vs. all dataset, a rather simple but time consuming computation is set: an initial population of TLE is imported, then propagated over a given timespan while conjunctions are checked for every possible pair of objects. In the end, only pairs of objects meeting a given criterion are included in the dataset.

The considered population is the TLE population at June 1st 2020 retrieved from space track. Only Low Earth Orbit (LEO) objects with an altitude under 2000 km are kept. The orbit dataset is then composed by 18415 objects. This population is then propagated over a 7-day period using the force model detailed in Table 1.

Table 1: Force model used to propagate the TLE population.

Perturbation	**Model for population propagation**
Earth potential	WGS84 Earth model with 12x12 development
Atmospheric Drag	MSIS00 atmospheric model with constant solar activity ($F10.7 = 140$ sfu, $Ap = 9$)
Solar radiation pressure	Cannonball model (Earth eclipses considered)
3rd body perturbation	Sun and Moon considered

It is worth noting that the area to mass ratio has been set to a constant value of 0.01 m^2/kg for all objects, ignoring the Bstar value from the TLE. Covariances are initialised using typical uncertainties on TLEs from [11]. A diagonal form of the covariance matrix is assumed, with [102; 471; 126] m deviations at one sigma for the position in QSW frame [28], and [1e-9; 1e-9; 1e-9] for the velocity. The covariance matrix is then propagated using the same force model as for the orbit propagation.

An all vs. all conjunction analysis is performed, which represents about 170 million pairs to be evaluated. For this proof of concept study, the criterion for conjunction was based on a distance threshold of 20 km (spherical shape of the safety volume), rather than the covariances. A first perigee and apogee filter of 50 km is used, meaning that only pairs whose apogee and/or perigee altitude differs less than 50 km are analysed. This allows to filter out about 105 million pairs, with about 65 million left for evaluation. A simple, non-optimised from CPU time point of view, detector is then used to ensure that no conjunctions are missed: multiple encounters (repetitive conjunctions between same objects) are considered. In the end, just over 1.5 million conjunctions are detected: the

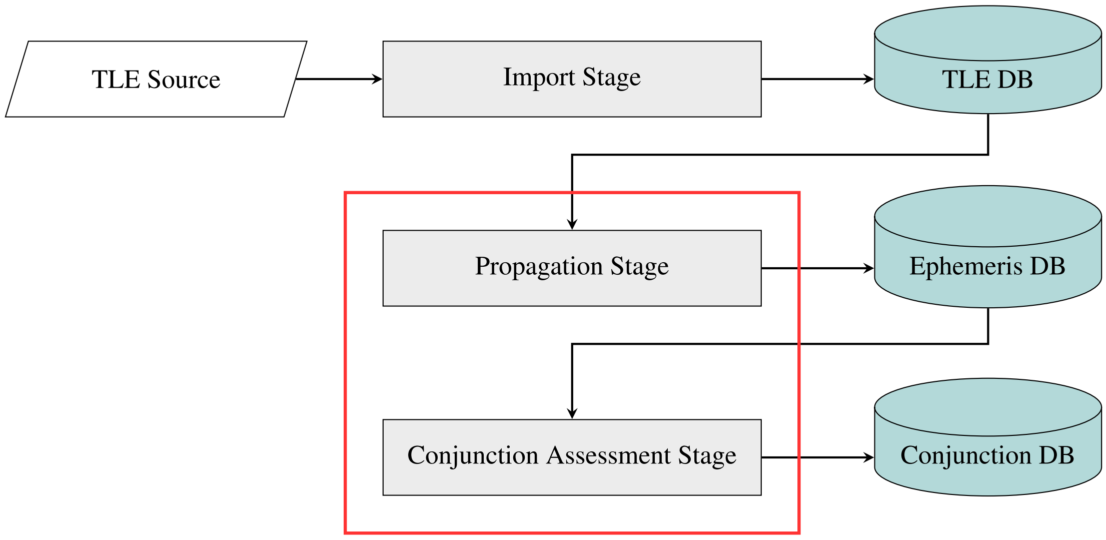


Figure 2: BAS3E modular approach. Modules with opportunities for computational savings by AI screening approach highlighted in red.

objects involved, as well as the time of closest approach (TCA) and distance of closest approach, are included in the dataset. The orders of magnitude are worth emphasising: an all vs. all analysis from a LEO TLE population represents the evaluation of 170 million pairs of objects, and more than 1.5 million conjunctions based on a 20 km threshold have been detected over a one-week period.

The whole computation exhibits a total of about 4 days of wall time in CNES HPC service, which includes the waiting time to access resources on the shared cluster of machines. Although BAS3E algorithms are really not optimised to perform such an all vs. all conjunction analysis, the very high number of pairs to evaluate shows the interest of investigating AI screening concepts.

2.2. Subset Selection

In such a large-scale and imbalanced scenario, for which there are many millions of object pairs but less than 1% are conjunction pairs, we chose to focus first on a subset of the dataset where the conjunction density is highest. This is shown in Figure 3, with altitude, and follows the same trend as the object spatial density. As such, objects were selected with the following criteria, with the perigee and apogee altitudes bounded by $h_p > 700$ km and $h_a < 1000$ km respectively. This resulted in a subset consisting of 4857 objects and more than 600,000 conjunctions. On the one hand, this altitude band is arguably the most important for the model to capture well, as debris-debris collisions would pose a risk to a greater number of operational satellites in such as well-populated orbital regime, and thus early detection is more crucial. The model would also have to cope with the largest scales over this regime, in which the majority of LEO objects reside. However, it was also important to limit the regime in order to select “interesting” non-conjunction cases to feed the model with, cases which are not necessarily easy to classify owing to large orbit separation.

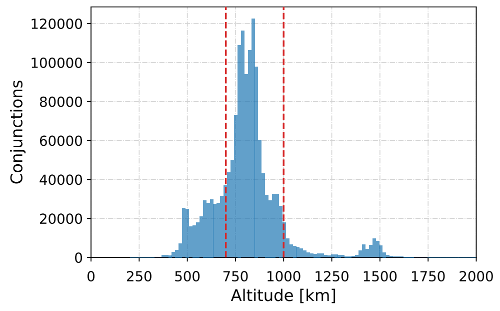


Figure 3: Distribution of conjunctions with altitude. Chosen altitude regime of 700 - 1000 km indicated by red dashed lines.

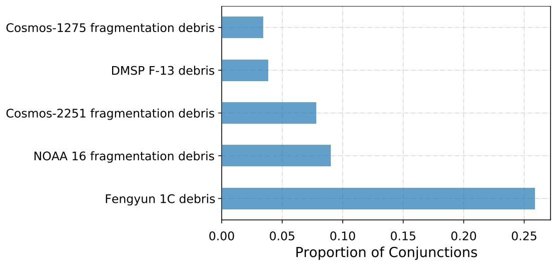


Figure 4: Top 5 objects (in all cases, fragment clouds) responsible for the most conjunctions in the 700 - 1000 km altitude regime .

Figure 4 shows the objects responsible for the most conjunctions for this data subset. It can be seen that the fragment cloud from the 2007 Fengyun 1C anti-satellite missile test accounts for more than a quarter of conjunctions in this regime. This is followed by fragments from the breakup of the NOAA 16 satellite in 2015, the Cosmos 2251 - Iridium 33 collision in 2009, and two other major break up events, which are described in more detail in [6].

3. MODELLING FRAMEWORK

3.1. Screening Objectives

In this work, we phrase the concept of conjunction screening, or close conjunction detection, as a machine learning classification task. Classification is an instance of supervised learning, in which the model learns to predict a class label given a training set of observations for which the true class is already known [4]. Here, we learn to predict a binary class as to whether a given object pair will have a conjunction, based on orbital data as an input, and class labels provided by the BAS3E dataset (Section 2.1). In this way, the trained machine learning model can be used to predict and subsequently filter or discard non-conjunction pairs based on their orbital data, thus reducing the computational burden on the full operational all vs. all conjunction assessment procedure.

In a classification context, the predictive performance of the model against known class labels can be described using the confusion matrix. The constituents of this, framed within the all vs. all problem, are as follows:

- True Positive (TP): correctly classified conjunction pairs.
- True Negative (TN): correctly classified non-conjunction pairs.
- False Positive (FP): non-conjunction pairs misclassified as conjunction pairs (false alarms).
- False Negative (FN): conjunction pairs misclassified as non-conjunction pairs (missed detections).

Consequently, the primary objective of the conjunction screening is to maximise the number of correctly filtered object pairs (TNs). However, conjunction cases which are misdiagnosed as non-conjunction cases (FNs), and thus wrongly discarded by the screening process, are also critical to the conjunction assessment problem. For the AI filter to be effective in reducing the computational cost of the BAS3E conjunction assessment stage, as shown in Figure 2, it must therefore:

- Maximise the number of correctly filtered non-conjunction pairs (TNs), subject to minimising missed conjunction detections (FNs).
- Minimise model inference time.

To further reduce the computational cost of the full conjunction assessment pipeline, this work is focused on the subclass of *tabular* classifiers. These models are designed to learn effectively from tabular data, for which a dataset consists of a database table, with each column representing a particular variable, or feature, and each row a given record [23]. For this problem, each row corresponds to a particular object pair and their respective orbital data, and thus the tabular model attempts to learn from a single initial state for each object pair. Feeding the model using only single states rather than full ephemerides has computational advantages, as well the potential to reduce propagation costs if certain objects are filtered out entirely by the AI screening process. The screening approach described in this paper therefore addresses both the propagation and conjunction assessment stages highlighted in red in the BAS3E pipeline shown in Figure 2.

To demonstrate the potential of this approach, we compare its performance in both of the aforementioned objectives, TNs and computational efficiency, to two classical, non-AI baselines. These baselines were chosen as, similar to our tabular approach, they are based on a single state, and not on the time series of the relative position and velocity of the object pairs. First, we consider an efficient and commonly used first filter for conjunction detection routines, the apogee-perigee filter (hereafter denoted as APSIS) [10]. This filter discards object pairs whose differences in apogee and perigee are greater than some critical distance, and was implemented as originally described in [15].

For the second baseline, we consider the Minimum Orbital Intersection Distance (MOID), which was implemented following the computational methods developed in [13]. In previous works, the MOID has been proposed as an effective second filter, following the apogee-perigee filter, to discard object pairs whose orbits do not come closer than some critical distance, regardless of the positions of the objects along the orbit [8]. However, in this initial work we treat the MOID as a stand-alone filter, independent of the APSIS, in order to be directly comparable with our approach. In both cases, the critical distance was chosen to be the same threshold as the collision criteria used in the generation of the BAS3E dataset (20 km).

3.2. Model Training

To successfully train a model under these objectives, we must first consider the imbalanced nature of the all vs. all dataset, for which less than 1% of object pairs are conjunctions. In such cases of class imbalance, the model will ignore the outlier conjunction cases, achieving a misleading 99% accuracy by always predicting the non-conjunction class. This is a common problem when dealing with highly imbalanced datasets for classification problems [18]. The model must therefore, if possible, be trained on a balanced dataset for it to successfully capture the different characteristics of the conjunction cases, before it can be applied to a representative test set.

Fortunately, the dataset is sufficiently large to be amenable to undersampling of the non-conjunction cases in order to rebalance the training set. In this way, a training set of 50,000 conjunction pairs and 50,000 non-conjunction pairs was randomly selected over the chosen altitude regime discussed in Section 2.2. Of the original 4857 objects in this regime, every object was included in at least one conjunction pair, and one non-conjunction pair. Each object pair was then included twice, with its order reversed, in order to prevent biases, resulting in a balanced data subset of 200,000 tabular rows. The use of such a comparatively small-scale subset for training compared to the size of the original dataset, is motivated and justified by the learning capacity of tabular models. Such models are not as "data greedy" as more complex architectures used for image or text learning, as they have less parameters, and consequently, achieve competitive performance with less observations.

This balanced subset was divided following the nominal 80% to 20% splitting strategy into training and validation sets. In both cases, the model has access to the true class label, with the latter used evaluate the performance of the model training and to tune higher level hyperparameters that are pre-set by the user and not learnt by the model, for example architectural parameters. The term test set is reserved for deploying the model on unseen data, which would exhibit the true scales and imbalance of the problem and for which in reality the class label would not be known, which is discussed in Section 4.

In addition to the problem of class imbalance, another important consideration for the model training is the number of missed detections, or FNs, generated by the model. Without special attention, the model would assign equal importance to FNs as to false alarms (FPs). We therefore design the training process to give importance to these events using several techniques: ensembling; weighted loss functions; and class thresholds.

First, we employ an ensemble of models [24]. As mentioned in Section 3.1, we elect to use tabular learning models as the dataset is represented as a database table. More specifically, we train an ensemble consisting of a deep neural network and a Gradient Boosted Decision Tree (GBDT). The predictions are averaged to provide the final answer, a procedure commonly known as *bagging* [7]. This is based on the idea that different models will produce different errors, so taking the average of their predictions should result in a more robust prediction that is closer to the correct answer [14].

As input features for the model, we chose the initial state vector and corresponding uncertainty of the propagated ephemerides used in the BAS3E pipeline, for each object. This was taken at the same epoch for all objects, in the EME2000 and TNW reference frames [28] respectively. We also elected to use the corresponding orbital elements, as well as the apogee and perigee, in order to better understand which characteristics the model was able to capture, and whether it was able to fully replicate the geometric APSIS and MOID baselines, which are both based on the osculating elements. It was found that the orbital elements had a higher feature importance in general for the model, but that inclusion of the state vector did result in improved performance. No further feature engineering was performed, resulting in a total of 40 features, per object pair. These features were normalised by transforming the mean of each feature to 0 and standard deviation to 1.

The neural network architecture we employ is a Multi-Layer Perceptron (MLP), since it is the most classical and popular architecture for modelling tabular data using neural networks. We use 3 hidden linear layers, each of which includes a previous batchnorm layer [17] and ReLU as activation function. The number of neurons in each of the three linear layers is indicated in Table 2. The network has been trained with a batch size of 1024 elements, a one-cycle learning rate schedule with its maximum learning rate set up as suggested in [26], and the *Lookahead* optimizer [31] (also known as RAdam). The hyperparameters shown in Table 2 were optimised for this architecture using a grid search, with remaining hyperparameters set as the default values from the *fastai* library[1].

Table 2: Hyperparameter settings for the deep learning model architecture and training procedure.

Parameter	**Value**
Hidden Layer Units	[200, 100, 50]
Loss Function	Focal Loss
Focal Parameter (γ)	3
Optimiser	RAdam
Weight Decay	0.01
Batch Size	1024

We make use of `xgboost` [9] as an implementation of the GBDT. GBDTs have held the state of the art for modelling tabular data in recent years, and have become extremely popular in machine learning competitions, since they achieve high prediction accuracy, are fast to train and easy to interpret [16]. The term *boosting* refers to another approach to ensembling which, unlike *bagging*, adds models instead of averaging them. In the case of GBDTs, an ensemble of boosted trees is created, where each new tree is trained to fit the error of all the previous trees combined [14]. This method is known to be very sensitive to the choice of hyperparameters, and therefore a bayesian hyperparameter search [5] was performed in order to find those that worked best for our dataset. The best settings found are shown in Table 3.

Following the training of the two models, we found that the optimal bagging strategy was a weighted average of the predicted probabilities of the two models, weighted more heavily (90%) towards the DL model, and that the ensemble predictions did improve upon the two individual models. This was found to be more successful than a

[1]https://github.com/fastai/fastai

Table 3: Hyperparameter settings for the XGBoost model architecture and training procedure.

Parameter	**Value**
max_depth	10
max_delta_step	0
$gamma$	0.4578
min_child_weight	3
$scale\_pos\_weight$	1
Loss Function	Logistic Binary
Learning Rate	0.0147

stacking procedure [30], in which an additional learning model was used to learn the combination weights.

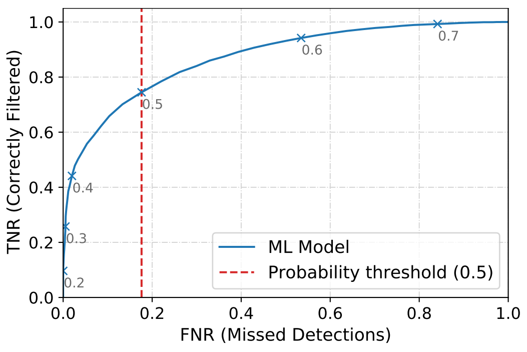
Second, we chose the loss function, which is used to optimise the neural network, as the *focal loss* [19]. A typical classification task in deep learning would make use of the cross-entropy loss, in which we first take the softmax of the model output and then the log likelihood of that [14]. The focal loss extends this loss by down-weighting easy-to-classify observations, with a strength proportional to the size of the focal parameter, γ. We have found that this helps in reducing the number of misclassified conjunction cases, which suggests that these are more difficult to classify, even in a balanced dataset. This can be explained by the fact that, while a large number of non-conjunction cases can be easily separated with an apogee-perigee like constraint, which uses very few features, many more need to be employed for detecting conjunction cases. Although the focal loss is normally used in practice with a weighting factor to deal with class imbalance, we found this not to be helpful in our experiments.

The third and final technique for forcing the importance of FNs, the tuning of the class thresholds, is discussed in more detail in the following section.

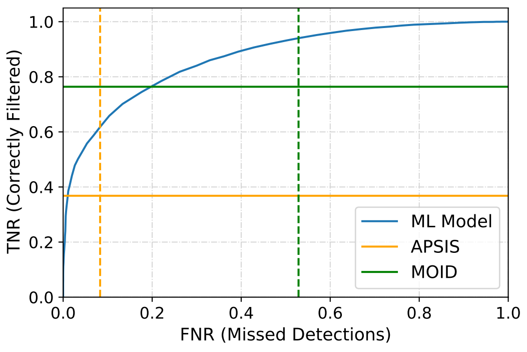
3.3. Missed Detection Threshold

Classification models such as the ones that were employed for model training do not just provide the expected class label for a given input, but also the predicted probability of the class. As such, the value of the probability threshold, from which the final class is decided, can be varied. In fact, this results in a trade-off between the successful screening capacity of the model, and the number of missed conjunctions or detections. This is shown in Figure 5a, where more conservative limits on the False Negative Rate (FNR) results in more conservative filtering, the True Negative Rate (TNR), and better filtering comes at the expense of more missed detections.

In this scenario, the acceptable number of missed detections should assuredly be an operator decision. To account for this, the balanced validation dataset was used to learn the class threshold required for a given percentage constraint that might be specified by the operator.



(a)



(b)

Figure 5: Trade-off of screening capability (TNR) with missed detections (FNR) for different class probability thresholds (a) With class thresholds annotated in grey and example of default threshold of 0.5 illustrated by the red dashed line (b) Comparison with baselines, whose performance is not tunable with choice of probability threshold.

This is illustrated in Figure 5a, in which the class probability thresholds are annotated along the curve, and the usual threshold of 0.5 (probabilities over 0.5 are assigned to class 1, conjunction, and conversely, probabilities below 0.5 are assigned to class 0, non-conjunction) is indicated by the dashed red line. In the nominal case of a probability threshold of 0.5, the model chooses to sacrifice 18% of all conjunction cases, in order to correctly classify and filter 75% of non-conjunction cases, with the remaining 25% flagged as false alarms. This proportion of misdiagnosed conjunction cases (FNs) is likely to be unacceptably high from an operator point of view, and thus this probability threshold can be modified for given percentage constraints on the FNR. For example, if it is acceptable for 10% of all conjunction cases to be misdiagnosed, this would result in the correct filtering of 65% of non-conjunction pairs.

This 10% limit is also likely to be high from an operator point of view, but it should be noted that this percentage is the total proportion of missed detections over the 7 day screening period, and thus does not contain a temporal dimension. In fact, it was found that the model performs better, with less missed detections, for conjunction events whose TCA was closer to the initial single state used to train the model. This is an expected behaviour, as the orbits of objects with late TCAs (towards the end of the 7 day screening period) will have been perturbed, and thus their conjunctions are more difficult to detect. The detection of these later events is less crucial, for a scenario in which the all vs. all analysis is run regularly, and thus this 10% constraint may be acceptable.

In Figure 5b, this tunable performance of the ML model is compared against the two baselines, whose FNR and TNR are fixed. It can be seen that for the same level of missed detections as the APSIS filter (8%), the ML model successfully filters out more than 20% more non-conjunction pairs. Similarly for the MOID, which appears to have a higher proportion of TNs for low FNs, but actually has a very high proportion (52%) of missed conjunction pairs itself, the ML model filters out 15% more object pairs for the same level of FNs. It can therefore be concluded that the ML approach outperforms both baselines for the same level of missed detections.

4. APPLICATION TO CONJUNCTION SCREENING

In this section, the performance of the model (hereafter denoted as the ML model) on unseen data which is representative of the scales and inherent imbalance of the all vs. all problem (i.e., the test set), is evaluated in terms of both screening capability and computational cost. For this, one million object pairs were randomly sampled from the altitude regime discussed in Section 2.2. Any pairs that were also present in the training or validation sets (Section 3.2) and thus whose labels were used to train or tune the model were subsequently removed, leaving 33,396 conjunction pairs and 961,059 non-conjunction pairs.

4.1. Screening Capability

As discussed in Section 3.1, we characterise the screening capability of a given model or approach as its ability to maximise the number of correctly filtered non-conjunction pairs (TNs), whilst minimising the number of misclassified conjunction pairs (FNs).

The performance of the ML model for these metrics, and comparison against the chosen baselines, is shown in Figure 6. From this figure, it can be seen that these two metrics fully describe the performance, with the number of correctly classified conjunctions (TPs) related to the number of missed detections (FNs) for the minority conjunction class, and correspondingly, the number of false alarms (FPs) determined by the number of correctly filtered non-conjunction pairs (TNs) for the majority non-conjunction class.

For the ML model, three possible constraints (1%, 5%, 10%) are given that may be feasibly chosen by an operator in order to limit the number of FNs, using the learnt probability thresholds established in Section 3.3 (see Figure 5a). As for Figure 5, it can again be seen that as the number of allowed missed detections increases, so too does the number of successfully filtered objects. It can also be seen that the learnt thresholds translate well from the validation to testing sets.

Comparing with the baselines, as expected, the APSIS filter is conservative, with a relatively low proportion of FNs resulting in a low level of TNs. In comparison, the ML model has a higher level of TNs for the same level of FNs, and thus can be concluded to be a more effective filter, which can be used even more conservatively than APSIS, with less FNs, if desired by the operator. On the other hand, the ML model can be seen to have a smaller number of TNs than the MOID for the chosen acceptable FN constraints. However, the use of the MOID in this way, as a stand alone filter, has an unacceptably high level of FNs for the all vs. all problem.

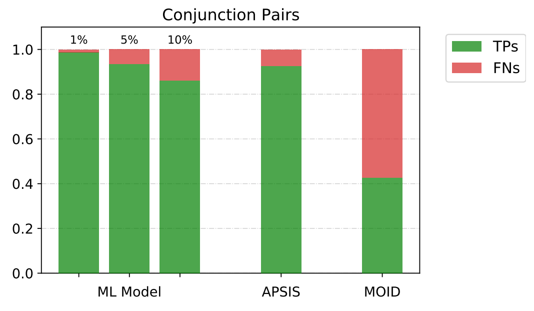
It should be noted that the critical distance used for the APSIS and MOID (20km) in this comparison is not a conservative value, as the altitude of the apogee and perigee of the orbit may vary more than this over a 7-day period due to perturbations. This warrants further investigations in future work for different critical distances to ensure that the better performance of the ML model is not only for this chosen threshold.

4.2. Computational Cost

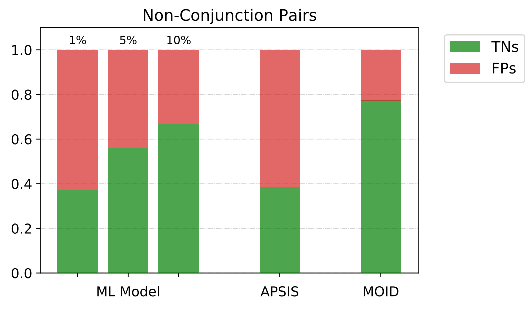
The overall computational cost, or rather computational saving, of employing an initial AI screening as a part of the conjunction assessment pipeline, is two-fold. On the one hand, we have the reduction in object pairs screened by the model, which translates to savings during the operational conjunction assessment process. On the other, the computational expense of the screening approach itself. The model must therefore be successful in both aspects, in maximising the number of filtered pairs (Section 4.1), and in minimising its inference or execution time.

A comparison of these runtimes between the ML model and baselines are given in Table 4 for the one million pair testset. To clarify, for the ML model this is the time taken to evaluate the trained model on new data. Naturally, this is less than the time taken to originally train the model (in this case, a modest 8.5 minutes), though this training needs only be performed once. It should also be noted that, unlike MOID and APSIS, in this comparison the ML model is run on a GPU to speed up inference time, although it can also be used in a CPU if necessary. The comparison was performed in a computing server with a Intel Xeon Bronze 3206R CPU and a Nvidia Quadro RTX 8000 GPU.

From Table 4, it can seen that the simplest approach to implement, the APSIS baseline, has the fastest runtime



(a)



(b)

Figure 6: Proportion of (a) Correctly classified conjunction pairs (TPs), missed detections (FNs) for the minority conjunction class (33,396 object pairs) (b) Correctly filtered (TNs), false alarms (FPs) for the majority non-conjunction class (961,059 object pairs) for a representative testset of 1 million object pairs, with different operator FN percentage constraints.

Table 4: Runtime for each screening approach over 1 million object pairs.

Screening Approach	**Runtime**
APSIS	0.1 seconds
ML Model	25 seconds
MOID	12.5 hours

of the three methods. It is clear, however, from Figure 6 that despite having the benefit of a low execution time, its use in this way, as a single pass filter, is not sufficiently effective. This therefore serves to justify its nominal use as an efficient and conservative initial filter, as a part of a multistage conjunction "sieve" [2]. The MOID baseline, on the other hand, can be seen from Table 4 to be too computationally expensive to be feasibly employed in this way for the all vs. all problem.

This leaves the ML approach, with a runtime of 25 seconds for the ensemble of models on a single GPU, which would scale up to approximately 70 minutes for the whole catalogue of 170 million objects pairs. This is arguably an acceptably low cost for the large-scales involved in this problem, as it would be conducive to being run regularly with updated of orbital data. In the face of increasing catalogue size, N, and consequently N^2 possible conjunction pairs, this approach would also be amenable to further improvement in computational efficiency through the use of parallel computing, as the pairs are treated independently and thus could be run concurrently. Also, the use of larger batch sizes in the neural network of the ML model would account for another efficiency boost, since less iterations would be needed in the inference.

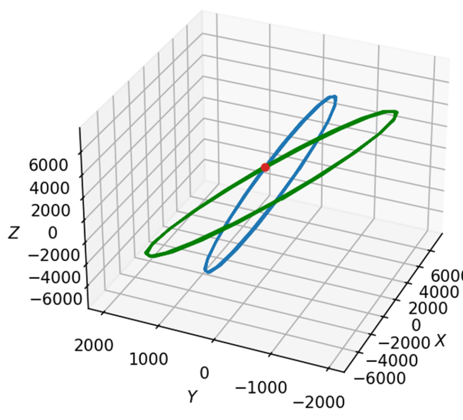
With a viable cost for the ML screening algorithm itself, the computational savings for the full operational conjunction assessment process can then also be quantified. Following the example given in Section 3.3, if an operator were to elect an acceptable missed detection constraint of 10%, this would result in a reduction of 65% of object pairs from the AI screening (Figure 6). As a consequence, the 4 days of HPC time used to generate the full dataset used in this work (Section 2.1), would be reduced to approximately 1.5 days. It should also be noted that this time is for the conjunction assessment stage only and does not include the propagation time, which would also be reduced if certain objects were filtered out entirely by the tabular ML model, which is based on single states. However, the cost of propagation is significantly less of that of the conjunction assessment stage.

These results demonstrate the promise of using AI in such an approach, with a substantial improvement in computational efficiency by employing an ensemble of relatively simple tabular models. This therefore justifies further investigation into the further improvement of the model, to increase the number of successfully filtered pairs in order to reach a point in which the computational expense is such that it would be feasible to run the full all vs. all analysis daily.

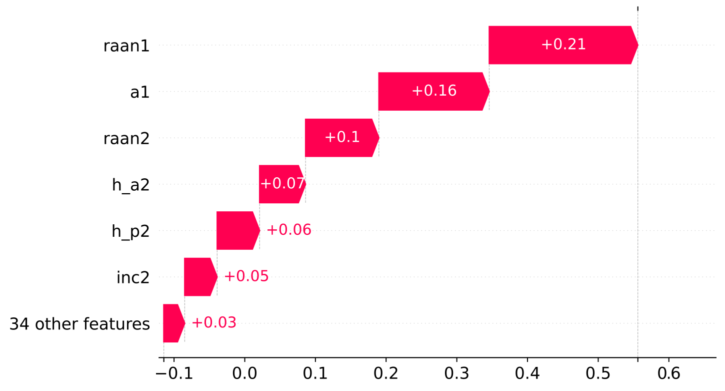
4.3. Explainable Predictions

From the point of view of space operations, the main drawback of using AI as a part of the conjunction assessment pipeline is its "black box" or non-transparent nature. In the same way that we want operator input on an acceptable missed detection constrain (Section 3.3), we also want an operator to be able to understand why a given prediction has been made, and have confidence that the tool is behaving as expected. This is a common obstacle for the adoption of AI in application domains, and has lead to the emergence of the field of Explainable AI (XAI) which aims to make the predictions of an AI system understandable and interpretable for humans [1].

To give insight into whether our ML model is behaving as expected, and facilitate an operator in understanding the prediction given for any object pair in the test set, we use the SHAP (SHapley Additive exPlanations) framework [20]. SHAP values are used to break down how the

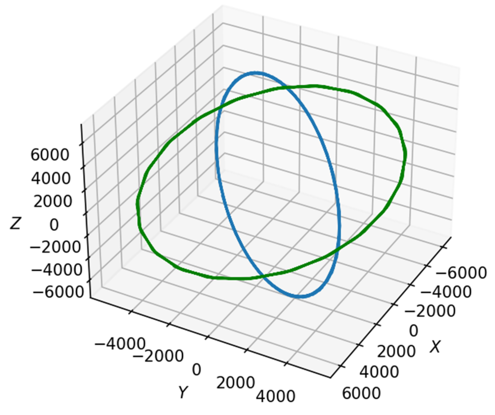


(a)

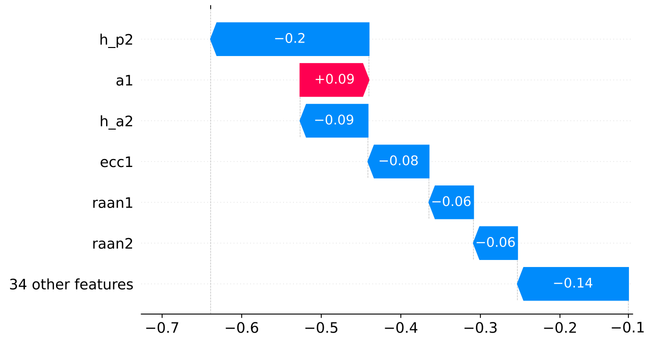


(b)

Figure 7: Example case of conjunction pair (NORAD IDs 13649, 3909). (a) Section of orbit ephemerides [km] around conjunction, with conjunction illustrated in red (b) SHAP waterfall plot of contributions to the prediction of the conjunction class. Right ascension of ascending node of the first and second object, semi-major axis of the first, apogee and perigee of the second can be seen to have the most impact on the final prediction.



(a)



(b)

Figure 8: Example case of non-conjunction pair (NORAD IDs 10777, 41228). (a) Section of orbit ephemerides [km] (b) SHAP waterfall plot of contributions to the prediction of the conjunction class (resulting SHAP values therefore negative). Apogee and perigee values of the second object, semi-major axis and eccentricity of the first, and right ascension of ascending node of both objects can be seen to have the most impact on the final prediction.

model works for an individual prediction by showing the contribution of each feature to the final prediction.

Two examples of objects pairs from the test set, for which the ML model was confident in assigning the class label, are shown in Figures 7 and 8. A waterfall plot is given for each case, displaying features with the largest impact on the model prediction. Here, the explainability plots are provided for the DL tabular model, as the ensemble was weighted heavily towards these predictions, but the SHAP values can similarly be applied to tree based methods for the second ensemble model, the GBDT. Positive SHAP values (pink) contribute to the prediction of a conjunction pair (the positive class), and negative values (blue) to a non-conjunction pair (the negative class). The SHAP values themselves represent the contribution of each feature to the final model prediction (a class probability), based on a prior expectation calculated on the background data distribution. Due to the imbalanced nature of the test set, this expectation is naturally shifted towards negative values and thus the total SHAP value does not represent the final predicted class probability, but rather can be used to infer the importance of particular features.

The first case, Figure 7, is an example of a conjunction between a rocket body and piece of debris (NORAD IDs 13649, 39094 respectively). The second, Figure 8, between a rocket body and space debris from the NOAA 16 fragmentation (NORAD IDs 10777, 41228 respectively). It can be seen in both cases that the most important features contributing to the predictions of the model for these events include the orbital elements of both objects, and also appear to well capture the apogee-perigee filter, as supported by Figure 6. This serves to validate that the ML approach is learning as might be expected.

5. CONCLUSIONS AND FUTURE WORK

This paper presents a demonstration of proof of concept for the use of AI techniques in aiding catalogue-wide all vs. all conjunction assessment. The all vs. problem is crucial to space situational awareness, but is a computational challenge owing to the vast and growing number of possible conjunction pairs. This motivates the need to look for new tools and approaches such as those provided by AI, and in particular by machine learning and deep learning.

In this work, we proposed to apply these techniques for initial conjunction screening of a large-scale all vs. all scenario in order to reduce the computational burden on pre-existing operationally used, and trusted, conjunction assessment tools. For this, a realistic catalogue-wide dataset of 18415 LEO TLE objects, and consequently 170 million possible conjunction pairs, was generated using the CNES BAS3E space surveillance simulation framework for a 7-day screening period.

Framed as a classification task, it was found that a machine learning approach based on an ensemble of tabular models outperformed classical non-AI filtering techniques (the apogee-perigee filter and the MOID) in terms of the number of correctly filtered conjunction pairs, given an operator-based constraint on the number of acceptable missed detections. The approach was also found to be computationally efficient and feasible for large-scale conjunction screening, and able to keep the operator in the loop. This performance using tabular models, which have a comparatively low representation capacity compared to other models, based only on single orbital states, demonstrates the potential for AI to aid in this problem. In future work, we will therefore work to improve the modelling approach using different data representations and more complex neural network architectures, for example based on the full ephemerides of the objects in order to better capture the effects of perturbations.

It should also be noted that this work was focused on the 700 - 1000 km altitude regime, for which the object and conjunction density is highest in LEO. In future work we will therefore also ensure that this approach extrapolates well to other altitude regimes, as well as expand the comparison to include additional classical filters.

ACKNOWLEDGMENTS

This research is supported by the EU H2020 MSCA ITN Stardust-R, grant agreement 813644. D. Camacho and V. Rodriguez-Fernandez would like to acknowledge support from the DeepBio project (TIN2017-85727-C4-3-P) granted by Ministry of Science and Education. H. Urrutxua would like to acknowledge support from grant ESP2017-87271-P (MINECO/AEI/FEDER, UE) and Project Grant F663 - AAGNCS by the "Dirección General de Investigación e Innovación Tecnológica, Consejería de Ciencia, Universidades e Innovación, Comunidad de Madrid" and "Universidad Rey Juan Carlos".

REFERENCES

1. Adadi, A. and Berrada, M. (2018). Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160.
2. Alarcón Rodríguez, J. R., Martínez Fadrique, F., and Klinkrad, H. (2002). Collision Risk Assessment with a 'Smart Sieve' Method. In *Proceedings of Joint ESA-NASA Space-Flight Safety Conference*, volume 486, pages 159–164. European Space Agency, ESA.
3. Alfano, S., Oltrogge, D., Krag, H., Merz, K., and Hall, R. (2021). Risk assessment of recent high-interest conjunctions. In *Proceedings of the 71st International Astronautical Congress, IAC*, Cyberspace Edition.
4. Alpaydin, E. (2020). *Introduction to machine learning*. MIT press.
5. Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *25th annual conference on neural information processing systems (NIPS 2011)*, volume 24. Neural Information Processing Systems Foundation.
6. Braun, V., Lemmens, S., Reihs, B., Krag, H., and Horstmann, A. (2017). Analysis of breakup events. In *Proceedings of the 7th European Conference on Space Debris*, volume 7. ESA Space Debris Office.
7. Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.
8. Casanova, D., Tardioli, C., and Lemaître, A. (2014). Space debris collision avoidance using a three-filter sequence. *Monthly Notices of the Royal Astronomical Society*, 442(4):3235–3242. Publisher: Oxford Academic.
9. Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.
10. Escobar, D., Águeda, A., Martín, L., and Martínez, F. M. (2012). Efficient all vs. all collision risk analyses. *Journal of Aerospace Engineering, Sciences and Applications*, 4(2):40–48.
11. Flohrer, T., Krag, H., and Klinkrad, H. (2008). Assessment and Categorization of TLE Orbit Errors for the US SSN Catalogue. In *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference*.
12. Flohrer, T., Krag, H., Merz, K., and Lemmens, S. (2019). CREAM - ESA's Proposal for Collision Risk Estimation and Automated Mitigation. In *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*.
13. Gronchi, G. F. (2005). An Algebraic Method to Compute the Critical Points of the Distance Function Between Two Keplerian Orbits. *Celestial Mechanics and Dynamical Astronomy*, 93(1):295–329.

14. Gugger, S. and Howard, J. (2020). *Deep Learning for Coders with Fastai and PyTorch: AI Applications Without a PhD*. O'Reilly Media, Incorporated.

15. Hoots, F. R., Crawford, L. L., and Roehrich, R. L. (1984). An analytic method to determine future close approaches between satellites. *Celestial mechanics*, 33(2):143–158.

16. Huang, X., Khetan, A., Cvitkovic, M., and Karnin, Z. (2020). Tabtransformer: Tabular data modeling using contextual embeddings. *arXiv preprint arXiv:2012.06678*.

17. Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

18. Lemnaru, C. and Potolea, R. (2011). Imbalanced classification problems: systematic study, issues and best practices. In *International Conference on Enterprise Information Systems*, pages 35–50. Springer.

19. Lin, T.-Y., Goyal, P., Girshick, R. B., He, K., and Dollár, P. (2017). Focal Loss for Dense Object Detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, Venice, Italy. IEEE.

20. Lundberg, S. M. and Lee, S. I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

21. Morand, V., Yanez, C., Perez, J. C. D., Fernandez, C., Roussel, S., Pucel, X., and Vidal, V. (2019). BAS3E: A framework to Conceive, Design, and Validate Present and Future SST Architectures. In *Proceedings of the 1st NASA International Orbital Debris Conference*, page 10.

22. Muelhaupt, T. J., Sorge, M. E., Morin, J., and Wilson, R. S. (2019). Space traffic management in the new space era. *Journal of Space Safety Engineering*, 6(2):80–87.

23. Piccolo, S. R., Lee, T. J., Suh, E., and Hill, K. (2020). Shinylearner: A containerized benchmarking tool for machine-learning classification of tabular data. *GigaScience*, 9(4):giaa026.

24. Sagi, O. and Rokach, L. (2018). Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249.

25. Sanchez, L., Vasile, M., and Minisci, E. (2020). AI and Space Safety: Collision Risk Assessment. In *Handbook of Space Security: Policies, Applications and Programs*, pages 1–19. Springer International Publishing, Cham.

26. Smith, L. N. (2017). Cyclical Learning Rates for Training Neural Networks. In *2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017, Santa Rosa, CA, USA, March 24-31, 2017*, pages 464–472. IEEE Computer Society.

27. Sánchez Fernández-Mellado, L. and Vasile, M. (2020). On the use of Machine Learning and Evidence Theory to improve collision risk management. *Acta Astronautica*.

28. The Consultative Committee for Space Data Systems (CCSDS) (2019). Navigation Data—Definitions and Conventions, issue 4. *CCSDS 500.0-G-4*.

29. Vasile, M., Rodríguez-Fernández, V., Serra, R., Camacho, D., and Riccardi, A. (2017). Artificial Intelligence in Support to Space Traffic Management. In *Proceedings of the 68th International Astronautical Congress*.

30. Wolpert, D. H. (1992). Stacked generalization. *Neural networks*, 5(2):241–259.

31. Zhang, M., Lucas, J., Ba, J., and Hinton, G. E. (2019). Lookahead optimizer: k steps forward, 1 step back. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.