Safe-event pruning in spacecraft conjunction management

Sébastien Henry[1](✉), Roberto Armellin[2], and Thibault Gateau[3]

1. *Georgia Institute of Technology, Atlanta, GA 30332, USA*
2. *The University of Auckland, Auckland, 1010, New Zealand*
3. *Institut Supérieur de l'Aéronautique et de l'Espace, Toulouse, 31055, France*

ABSTRACT

Spacecraft conjunction management plays a crucial role in the mitigation of space collisions. When a conjunction event occurs, resources and time are spent analyzing, planning, and potentially maneuvering the spacecraft. This work contributes to a subpart of the problem: Confidently identifying events that will not lead to a high collision probability, and therefore do not require further investigation. The method reduces the dimensionality of the data via principal component analysis (PCA) on a subset of features. High-risk regions are then determined by clustering the projected data, and events that do not belong to a high-risk cluster are pruned. A genetic algorithm (GA) is developed to optimize the number of clusters and feature selection of the model. Furthermore, an ensemble learning framework is proposed to combine the suboptimal models for better generalization. The results show that the first set of parameters pruned approximately 50% of the events in the testing set with no false negatives, whereas the second set of parameters pruned 70% of the events and maintained a near-perfect recall. These results could benefit the optimization of operational resources and allow operators to focus better on the events of interest.

KEYWORDS

space situational awareness
collision avoidance
genetic algorithm (GA)
ensemble learning

Research Article
Received: 10 October 2022
Accepted: 12 May 2023
© Tsinghua University Press 2023

1 Introduction

The advent of the new space economy has resulted in various challenges in spacecraft operations. Each year, the European Space Agency (ESA) tallies an estimated population of objects around the Earth, which has expanded quite considerably over the last decade, with no sign of stopping any time soon [1, 2]. In general, space is becoming more accessible to private companies, and this induces more satellites to be launched, including so-called mega-constellations such as Starlink, OneWeb, and Project Kuiper [3, 4]. Simultaneously, the consequences of a lack of proper end-of-life management in current and previous missions create space junk. Each collision can generate more debris, thereby increasing the likelihood of future collisions. This vicious circle is often referred to as the Kessler syndrome or collisional cascading [5, 6]. There are multiple ways to prevent the situation from worsening, including proper spacecraft end-of-life management [7], active space debris removal [8, 9], and improved operations to mitigate collision risk [4]. This study focuses on the latter aspect.

When a close approach between two spacecraft is suspected, a conjunction data message (CDM) is sent to the control center with several parameters and computed risks. Several CDMs are issued for a single event. A few days before the time of closest approach (TCA), the last CDM is analyzed, and in the case of a risky event, the operations team starts to plan for a potential collision avoidance maneuver [10, 11]. This time-sensitive process requires human and computational resources. There is a risk threshold, above which the spacecraft must be maneuvered. The computed analytical risk may evolve rapidly after the last CDM is received, that is, two days before the TCA. In some cases, the risk of a dangerous encounter decreases below the maneuver threshold. This is referred to as a false positive; manpower and spacecraft

✉ seb.henry@gatech.edu

Nomenclature

n_{clusters}	Number of clusters in the K-means
$n_{\text{components}}$	Number of principal components in the PCA
n_{elites}	Number of individuals that are kept through the next iteration of the genetic algorithm
n_{features}	Number of features in the dataset
n_{gen}	Maximum number of generations in the genetic algorithm
n_{pop}	Size of population in the genetic algorithm
$n_{\text{tournament}}$	Number of individuals participating in the selection tournament
P_{C}	Probability of collision
r_{mut}	Mutation rate in the genetic algorithm
r_{xover}	Crossover rate in the genetic algorithm
γ_{hr}	Concentration of high-risk events in a cluster
δ	Small positive real number much smaller than one
ϵ	Threshold of high-risk concentration above which a cluster is dangerous
θ	Majority voting threshold

fuel can be spared by not maneuvering if it is possible to identify those events in advance with confidence. Conversely, an event deemed low-risk can become a high-risk event within two days before the closest approach. This is referred to as a false negative, and it is crucial to identify such events in advance to take action as soon as possible. As in many other fields, the importance of lowering risks as much as possible cannot mean flagging any possible encounter as dangerous because it renders the information dull. The arguments above highlight the importance of automating the collision avoidance process and making it more robust.

Data science is a promising path to use in addition to analytical methods computing the probability of collision (for example, Refs. [12, 13]), which motivated ESA to organize a machine learning competition [14] with a dataset containing real conjunction events. The objective of the competition was to develop a model that can classify whether an event is dangerous. A variety of methods have been proposed to solve this problem with data, but not all of them involved elaborate models. The winning team did not use machine learning but rather proposed a set of decision rules based on data statistics [14]. Although it is a light model, the main advantage of this approach is that the chain of decisions and relevance of each feature are easily interpretable. Gradient-boosted decision trees (GBT) [15] are a type of ensemble learning model that has been applied to this problem [14, 16] and have shown good results while also providing some type of feature relevance. The ensemble learning framework combines multiple models in the decision process to improve the generalization of the output. Recurrent neural networks [17] have also been investigated owing to their inherent ability to manage sequential inputs [14, 16, 18, 19]. Notably, the team in third place used Manhattan long short-term memory (LSTM) siamese networks as an ensemble learning framework. The purpose of the siamese architecture is to analyze the similarity between pairs of events that are anomalous (i.e. that become dangerous although they were safe at the decision time, or vice versa) and non-anomalous (i.e., whose risk level remains the same after the decision time). In addition, studies proposed the use of Bayesian inference in addition to LSTM to predict the distributions of CDMs [18]. This method has several advantages: It can predict all the features of the CDM with uncertainties and generate synthetic data [20, 21]. Another notable study investigated the Dempster–Shaffer theory of evidence to account for epistemic uncertainty in the computation of collision probabilities. This study coupled neural networks, random forests, support vector machines, and k-nearest neighbors on simulated datasets [22, 23].

All the aforementioned practices lead to good overall classification but still classify a non-negligible number of dangerous events as safe. This has lead this work to investigate a method that favors a very low number of false negatives by design. This would eventually allow for the reduction or optimization of computational resources by filtering superfluous events. This is achieved using clustering techniques coupled with a genetic optimizer to identify dangerous regions in the dataset. An ensemble voting scheme, which has been proven to work well in previous studies, is also integrated. The remainder of this paper is organized as follows. First, the problem is described using data and objective functions. Second, a pruning algorithm is developed via the identification of high-risk clusters, optimization, and ensemble learning. Finally, the algorithm is tested using multiple configurations.

2 Problem description

This study approaches the topic as a classification problem using the conjunction event timeline originally presented during the competition. A conjunction event consists of a series of timed CDMs, in which only the CDMs at least two days before the TCA can be used to predict the collision probability. If the last CDM of the series is within one day of the TCA, it is used as the solution. Hence, to be usable, an event must have at least one CDM two days before the TCA and one CDM within one day of the TCA. Figure 1 illustrates a usable conjunction event graphically. The original training set for the competition comprised 13,154 events, of which only 66 were high-risk and usable. The testing set comprised 2,167 usable events, of which 150 were high risk. Thus, there was a strong imbalance between the two sets. This study uses the raw data comprising 18,379 events and splits the test/train to obtain a homogeneous distribution of high-risk events. The same standard as that used in the competition is used: An event is considered high-risk if the collision probability P_C is greater than 10^{-6}, where P_C is computed using the method developed in Ref. [12].

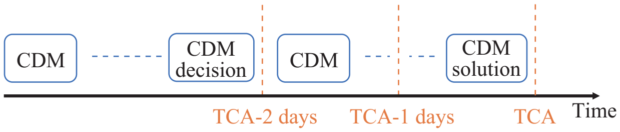

Fig. 1 Graphical representation of a usable conjunction event.

The benchmark for comparison during the competition was the latest risk predicate (LRP), which simply estimates the final risk using the risk of the CDM at the decision time. This solution is trivial, yet difficult to improve, as highlighted in the results of the competition [14]. While the competition aimed to find a globally well-balanced model, the purpose of the subsequent work is to prune as many events as possible, including as few high-risk events as possible. A few metrics for analyzing the solutions are highlighted here. The *recall* is the fraction of true positive events over the number of positive events in the solution:

$$recall = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{1}$$

To keep the operations as safe as possible, the number of false negatives should remain as low as possible; therefore, *recall* should be as close to one as possible. However, considering *recall* alone is too conservative as one could flag every event as dangerous, even though the majority is not. Therefore, another function for evaluating the model performance is the *precision*, which is the fraction of the number of true positive answers over all events detected as positive:

$$precision = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \tag{2}$$

To maintain the value of this function high, the number of false positives must be reduced. Again, considering this function alone is not a good strategy, as it does not require a low false-negative rate. Instead a balance between *precision* and *recall* must be found, which is commonly done with the F_β measure [24]. $\beta = 2$ is selected to assign more weight to the *recall*,

$$F_2 = 5\frac{precision \cdot recall}{4precision + recall} \tag{3}$$

This paper aims to confidently prune as many events as possible; thus, it aims to guarantee a near-perfect *recall* and maximize *precision*. The function to optimize loses one dimension as *recall* is fixed. Let $\delta \in \boldsymbol{R}^+$ be an arbitrarily chosen threshold ($\delta \ll 1$); the fitness function can be interpreted as

$$F = \begin{cases} \text{true negative}, & \text{if } recall > 1-\delta \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

3 Methodology

3.1 Data processing

The following pipeline is used to preprocess the data. For an exhaustive description of the data features, please refer to Ref. [14] and the competition dataset[①]. The *c_object_type* column has been removed because it is not a number (NaN). Column *c_rcs_estimate* is removed because it is often empty. All the CDMs containing empty or NaN values for the remaining attributes are excluded. Only events in which there is at least one CDM two days before the TCA *and* a CDM within one day of the TCA are kept. The features *t_position_covariance*, *c_position_covariance*, and

[①] https://kelvins.esa.int/collision-avoidance-challenge/data/

max_risk_scaling are considered as their $\log_{10}$ values. For each event, only the CDM closest to the decision time is used as the input and the CDM closest to the TCA is used as the solution. This last step assumes a Markov process.

After preprocessing, 9,666 events remain, of which 206 are high-risk. 73 high-risk events have a CDM at the decision time that is low-risk, meaning that the latest risk predicate (LRP) would wrongly classify them as safe. This scenario should be avoided for safety reasons. The data is divided as follows: 75% for training and 25% for testing. The inputs are transformed using a standard scaler from Scikit-Learn [25]. Only the training set is used to fit the standard scaler; however, both the training and testing sets are scaled.

3.2 Pruning algorithm

The strategy adopted in this study involves separating the data into clusters, where each cluster can either be concentrated in dangerous events or not. An event is considered safe to prune if it does not belong to a cluster with a high density of dangerous events. Because of the large number of features in the data, it is practical to first select a subset of attributes and reduce the dimensionality of the data before clustering. With the training data solutions available, the concentration of high-risk events in a cluster γ_{hr} can be computed. The quantity ϵ is defined as the threshold concentration of high-risk events above which a cluster is identified as dangerous. All clusters in which $\gamma_{\text{hr}} > \epsilon$ are considered dangerous. The lower the value of ϵ, the more conservative the classification; however, fewer events are pruned. Therefore, for the training set, it is guaranteed that each cluster deemed safe has a false negative concentration lower than ϵ. This favors a globally high *recall*. The false negative concentration is not bounded on the testing set but is expected to remain low if the model generalizes well.

Principal component analysis (PCA) [26] is commonly used in statistics to reduce the dimensionality of a dataset, and it is chosen for this study because it is very mature and fast. Mini-batch K-means [27] is chosen as the clustering algorithm because it has a similar performance and is faster than regular K-means, a standard algorithm for clustering. Thus, a simple model consists of PCA and K-means, and is parameterized by the number of components for the PCA, $n_{\text{components}}$ (1 integer), number of clusters for the K-means, n_{clusters} (1 integer), and whether a specific feature is selected (n_{features} booleans). The selection of parameters can be either arbitrary or determined using an optimization scheme (see Section 3.3). Note that ϵ is not considered in these parameters because it enforces the quality of the solution.

3.3 Optimized parameters

Multiple approaches can be used to optimize the parameters; however, here, genetic algorithms (GA) are selected because of the discrete nature of the parameters. Brute force is not an option; assuming a fixed number of PCA components and clusters, there are still 100 features to select, which sums up to 2^{100} possibilities. GAs are metaheuristic procedures that allow the identification of optima by imitating nature-like evolutionary principles. A certain set of parameters must be adapted to achieve better convergence [28–30]. The main idea behind tweaking each parameter is the global balance between exploration and exploitation. The details of the implementation of this study are shown in Table 1.

A population of parameters consists of n_{pop} sets of parameters that are individually tested. A small random population of five individuals, with only three features, is represented in Table 1. A large population usually performs better because it covers a larger part of the domain, especially at the beginning when randomly generated. However, large populations require additional computational power and memory per generation. A

Table 1 Representation of a population of sets of parameters, where $n_{\text{pop}} = 5$

	gene 1	gene 2	gene 3	gene 4	gene 5
	$n_{\text{components}}$	n_{clusters}	*risk*	*max_risk_scaling*	*time_to_tca*
Individual 1	2	7	0	1	0
Individual 2	4	3	0	1	1
Individual 3	3	10	1	0	1
Individual 4	5	7	0	1	0
Individual 5	7	5	1	1	0

small population may converge faster but more likely towards a suboptimal solution. In general, a larger population is preferred to more generations [31]. The initial parent population is generated randomly or otherwise, and the performance of each individual is assessed. The offspring population is generated from the parent population. The pipeline from the parent to offspring consists of four main steps:

(1) **Selection**: The most usual way to select an individual from the parent population is either randomly or via a tournament. The latter is chosen here because it tends to raise the quality of the population [32]. In this case, the tournament makes $n_{\text{tournament}}$ randomly picked parents compete, and the one with the best fitness is selected.

(2) **Crossover**: Once two parents have been selected (from two different tournaments), their genes are mixed to create two new individuals. The idea is to try to obtain the best features from both parents. This implementation uses a uniform crossover. The two new individuals start as a copy of their respective parent. Then, per gene, they have a chance r_{xover} to switch to the particular gene of the other parent.

(3) **Mutation**: After crossover, each gene can randomly mutate. A high mutation rate allows the offspring to differ from the parents and thus allows the exploration of a greater part of the domain. Conversely, a low mutation rate is more suitable when one attempts to refine the previous generation and converge. The mutation used here is a random integer mutation, characterized by the probability of mutation per gene, r_{mut}. If there is mutation, the random integer can be anywhere in the allowable domain and can also be the same as the previous gene (so the binary elements have a chance $r_{\text{mut}}/2$ to change).

(4) **Fitness survival**: Finally, once the entire offspring population is generated (it contains n_{pop} individuals), its n_{elites} worst individuals are replaced with the n_{elites} best individuals of the parent population. It ensures that the best solutions do not disappear.

The offspring population becomes the parent population for the next step, and the algorithm stops after the maximum number of generations. At the end of the algorithm, the parameters of the best individual in the offspring population are maintained. The genetic optimization scheme is summarized in Algorithm 1 and Fig. 2.

F_2 is chosen as the fitness function for optimization because it indirectly maximizes the number of true negatives, provided that ϵ is extremely small. Suppose the number of false negatives remains low and nearly constant; consequently, the number of true positives also remains nearly constant. Furthermore, assuming that only a small percentage of positive events are incorrectly classified, *recall* remains high and constant. In this case, increasing the F_2 function naturally increases *precision*. This means reducing the number of false positives (i.e., keeping the number of true positives constant) or raising the number of true negatives. As with other machine learning schemes, the algorithm is prone to overfitting and thus may not generalize well to the testing set. The more generations there are, the more optimal the solution is on the training set, but not necessarily on the testing set.

3.4 Ensemble learning

Although the pruning algorithm presented in Section 3.2 can be used as a standalone algorithm, it is prone to exploring a local optimum and overfitting the training data when there are too many generations. Instead, aggregating multiple weaker models can result in better generalization, which is the underlying idea behind

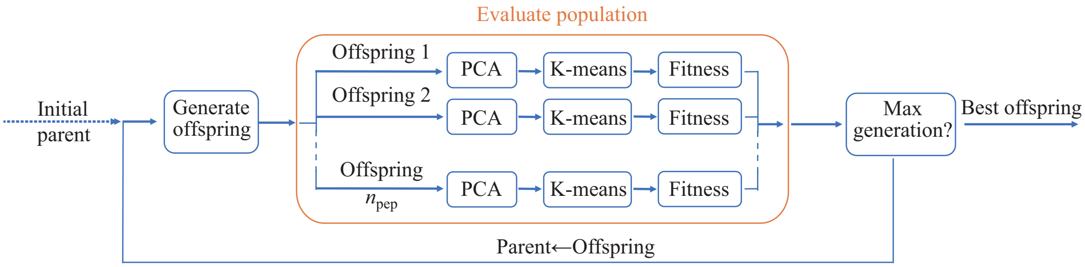

Fig. 2 Flowchart of training via genetic optimization.

Algorithm 1 Search for optimal parameters with a genetic algorithm

```
parent ← initialization()
evaluate performance of parent (PCA + $K$-means)
while $i \leqslant n_{\text{gen}}$ do
    offspring ← {}
    while |offspring| $< n_{\text{pop}}$ do
        individuals ← {}
        while $k \leqslant 2$ do                          /*selection of 2 individuals for crossover*/
            individual ← tournament(parents, $n_{\text{tournament}}$)      /*tournament selection*/
            individuals ← individuals ∪ {individual}
            $k \leftarrow k + 1$
        end while
        individuals ← crossover(individuals, $r_{\text{xover}}$)  /*random swap of genes between the individuals*/
        individuals ← mutation(individuals, $r_{\text{mut}}$)                /*random mutation of genes*/
        offspring ← offspring ∪ {individuals}
    end while
    evaluate performance of offspring (PCA + $K$-means)
    offspring ← elite_survival(offspring, parent, $n_{\text{elites}}$)          /*keep the best parents*/
    parent ← offspring
    $i \leftarrow i + 1$
end while
return best individual in parent
```

ensemble learning [15, 33]. Gradient-boosted trees, following ensemble learning logic, have already been proven to enhance the performance of the LRP in the competition summary [14, 16]. Similar to the standard approaches of forests of trees, one could use an ensemble of sub-models created as described in Section 3.3 to improve their specific performance.

An ensemble learner comprises several sub-models or learners. The number of learners in the ensemble model is called the ensemble size and can be determined in multiple ways [33, 34]. A fixed number of learners is chosen in this case. Each of the sub-models is created and trained as in Algorithm 1. The training of each learner is important. The training depth is characterized by the number of iterations. By adjusting this parameter, stronger or weaker learners can be combined. A few configurations with shallow and deep models are presented in Section 4.3.

When an event needs to be classified, it is first estimated using all the sub-models. There are different ways of combining models [33], from which we choose a simple variant of majority voting. Typically, an event that is estimated to be positive by at least 50% of the sub-models is classified as positive. However, this is not the rule chosen here. Given that the method favors a very low false negative concentration, the threshold θ is placed at a higher percentage. The greater the ensemble size, the lower the importance of a particular learner in the final classification. This has a mixed impact: Whether a model is particularly good or bad, its importance is only moderate in the final classification. A flowchart of the training of the ensemble learning model is displayed in Fig. 3.

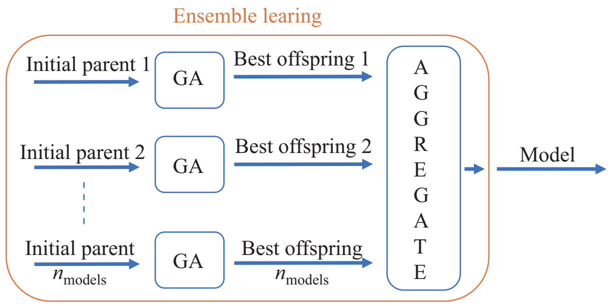

Fig. 3 Flowchart of the training of the ensemble model.

4 Results

4.1 Handpicked feature selection

The results are based on a random data split, the statistics of which are presented in Table 2. Several handpicked features are selected from the list of important features in Ref. [14]: *risk*, *miss_distance*, *max_risk_scaling*, *c_position_covariance_det*, *c_sigma_t*, *t_position_covariance_det*, and *c_obs_used*. An explanation of these features is provided in the Appendix. After the training set is used to identify the principal components ($n_{\text{components}} = 5$), the data can be projected, as illustrated in Fig. 4. The high-risk events are mainly concentrated in one region of the projection. A smaller region contains two high-risk events in the training set, but not in the testing set. A K-means algorithm is used to cluster the data ($n_{\text{clusters}} = 5$) and identify the dangerous clusters. The clusters used in this example are shown in Fig. 5, and Tables 3 and 4 highlight the proportion of high-risk events in each cluster. If one is ready to accept a maximum concentration of $\epsilon = 0.1\%$ high-risk events in the discarded clusters, then it would be possible to rule out Clusters 2, 4, and 5 or 4,386/7,249 = 61% of events. Pruning the same clusters on the testing set results in 1,465/2,417 = 61% of discarded events. The metrics of the fitness functions in this example are listed in Tables 7 and 8 for the training and testing sets, respectively.

Table 2 Training and testing sets obtained from a random split of the data

	# events	# high risk	# false negative	# false positive
Train	7,249	163	57	180
Test	2,417	40	16	77

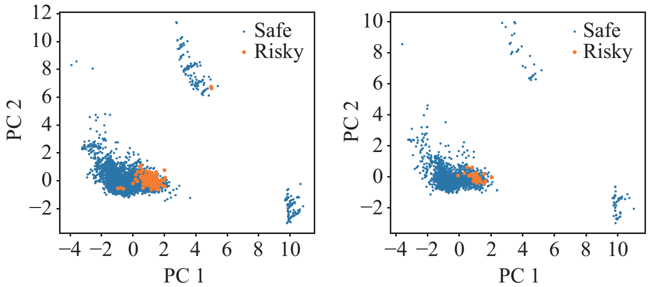

Fig. 4 Scatter of the data on its first two principal components. Left: training set. Right: testing set.

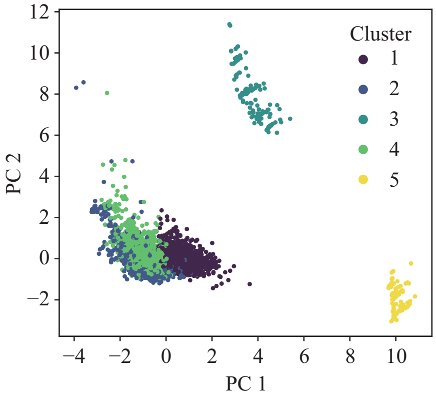

Fig. 5 Clusters obtained with a K-means algorithm of five clusters on the training data.

Table 3 K-means results over the training set

Cluster	# in cluster	# high risk	% high risk
1	2,750	158	5.75
2	1,461	1	0.068
3	113	2	1.77
4	2,814	2	0.07
5	111	0	0

Table 4 K-means results over the testing set

Cluster	# in cluster	# high risk	% high risk
1	922	39	4.23
2	482	0	0
3	30	0	0
4	938	1	0.11
5	45	0	0

4.2 Optimized feature selection

The Pymoo library [35] is used for the GA. The population size is set to $n_{\text{pop}} = 100$ because it was found to have a good balance between diversity and computation time. The number of participants in the selection tournament is maintained at the default value of $n_{\text{tournament}} = 2$. The mutation rate is set to $r_{\text{mut}} = 0.05$ and the crossover rate is set to $r_{\text{xover}} = 0.05$. These values are selected in the low range because there are many features, and excessively high values make the algorithm unsuitable for exploitation. The number of elites is maintained at a default value of $n_{\text{elites}} = 1$. The authors do not claim that the aforementioned genetic optimizer parameters are optimal. By using the principles described in Section 3.3, the parameters were determined by experimenting with the training data on the population size, the mutation rate, and the crossover rate. A typical learning curve for specific GA training on F_2 is shown in Fig. 6, where it can be observed that the F_2 function increases with generations because of *precision* but not because of an increase in *recall*.

The overall result of the optimization depends on (1) the split of the data and (2) the randomness of the GA. Consequently, it is more useful to examine the statistics

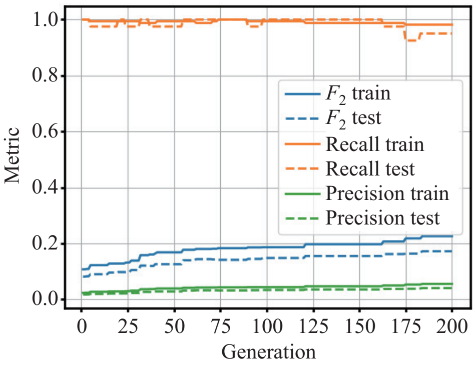

Fig. 6 Example of the learning curve obtained during genetic optimization ($\epsilon = 0.001$).

over multiple experiments rather than a single case. The following experiment simulates 50 data splits, for which a GA is trained. The experiment is repeated using the same 50 data splits but with different numbers of iterations. ϵ is maintained at zero, indicating that the *recall* should be perfect in the training set. The mean performances of the obtained solutions, given the number of iterations, are presented in Tables 5 and 6 for the training and testing sets, respectively. It is observed that maintaining perfect *recall* on the training set does not necessarily guarantee perfect *recall* on the testing set. The *recall* of the testing set decreases as the number of generations increases.

4.3 Ensemble learning

A total of 50 models are trained and assembled. If θ percent or more of the models classify an event as dangerous, then it is classified as dangerous; otherwise, it is not. Two approaches are tested in the study. First, there are only three generations per GA, which means that the sub-models are shallower and more general. The majority voting threshold for this particular ensemble model is set at $\theta = 85\%$. Second, there are 50 generations per GA, meaning that the sub-models are more optimized on the training set (but not necessarily on the testing set). The majority voting threshold is slightly lowered to $\theta = 80\%$ for better generalization. The threshold ϵ is set to 0.001 in both cases and the remaining parameters for the GA follow the values in Section 4.2. The experiment is conducted using the same data split as in Section 4.1. A typical training curve as the number of models increased is shown in Fig. 7. The *recall* of the training data consistently stays near one by design. The *recall* of the test set drops after the fourth model is added; however, ensemble voting allows to filter out the mistakes when

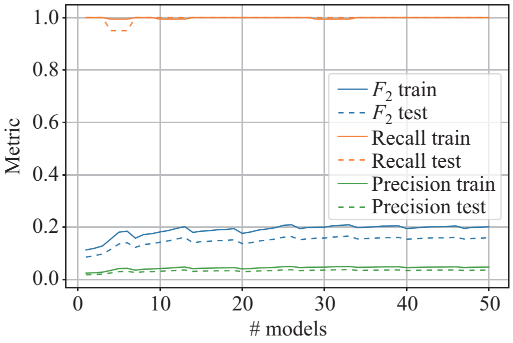

Fig. 7 Learning curve obtained during ensemble learning with sub-models of three generations.

Table 5 Mean (μ) and (σ) standard deviation of the classification metrics on 50 data splits depending on the depth of the genetic algorithm and training set

Generations	*recall*		*precision*		F_2		# pruned	
	μ	σ	μ	σ	μ	σ	μ	σ
3	1	0	0.0250	0.0016	0.1134	0.0066	1,063.6	330.43
20	1	0	0.0403	0.0068	0.1730	0.0247	2,840.7	581.34
50	1	0	0.0477	0.0068	0.1998	0.0236	3,967.2	418.32
100	1	0	0.0518	0.0070	0.2139	0.0239	4,231.1	356.79

Table 6 Mean (μ) and standard deviation (σ) of classification metrics on 50 data splits depending on the depth of the genetic algorithm and testing set

Generations	*recall*		*precision*		F_2		# pruned	
	μ	σ	μ	σ	μ	σ	μ	σ
3	0.9867	0.0159	0.0236	0.0036	0.1074	0.0151	356.14	113.76
20	0.9665	0.0257	0.0372	0.0073	0.1603	0.0274	1,114.2	189.36
50	0.9579	0.0311	0.0436	0.0076	0.1835	0.0270	1,323.2	136.37
100	0.9515	0.0351	0.0468	0.0075	0.1949	0.0261	1,409.5	120.22

more models are added. The curve of the F_2 function smoothens as the number of models increases because of voting.

The metrics for the shallow and deep approaches are presented in Tables 7 and 8 for the training and testing sets, respectively. In the shallow approach, the ensemble model achieves a perfect *recall*, and prunes 1,328 events or 55% of the testing set. This means that in this particular data split, it is possible to safely remove the majority of events without any mistakes. On the testing set for the deep approach, the ensemble learning algorithm manages to prune 255 or 10% more cases than the handpicked selection while maintaining the same *recall*. In total, 70% of the events are pruned, and only one high-risk event is removed. For comparison, classifications from LRP and GBT (see Section 4.4) are also provided in Tables 7 and 8, and the values of their *recall* show that they have significantly more false negatives. This illustrates that the method presented in this paper is more cautious than the compared literature and shows that an event deemed as low-risk by this approach is safer to prune.

Table 7 Results between different approaches on the training set

Cluster	*recall*	*precision*	F_2	# pruned
LRP	0.6503	0.3706	0.5650	6,963
GBT	0.6380	0.5333	0.6139	7,086
Handpicked	0.9816	0.0559	0.2276	4,386
Ensemble shallow	1	0.0482	0.2019	3,864
Ensemble deep	0.9816	0.0759	0.28975	5,137

Table 8 Results between different approaches on the testing set

Cluster	*recall*	*precision*	F_2	# pruned
LRP	0.6000	0.2376	0.4597	2,316
GBT	0.5750	0.2949	0.4832	2,377
Handpicked	0.9750	0.0410	0.1754	1,465
Ensemble shallow	1	0.0367	0.1601	1,328
Ensemble deep	0.9750	0.0537	0.2066	1,690

Histograms of event populations before and after pruning are shown in Figs. 8 and 9 for shallow and deep models, respectively. The more conservative approach

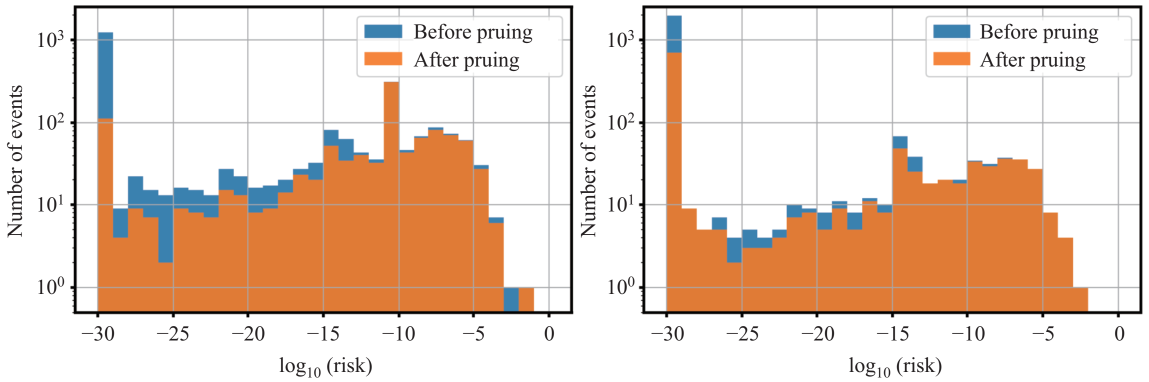

Fig. 8 Histogram of the number of events before and after pruning, for the shallow ensemble model. Left: risks at decision time. Right: risks at the solution.

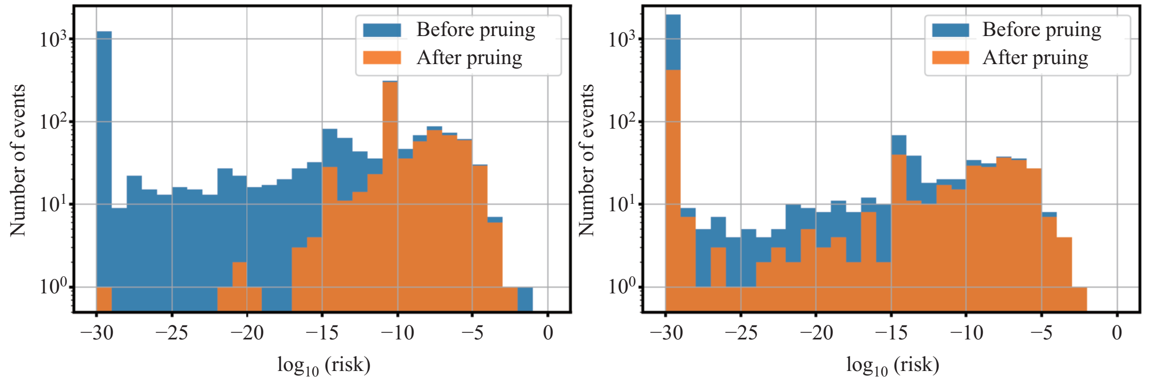

Fig. 9 Histogram of the number of events before and after pruning, for the deep ensemble model. Left: risks at decision time. Right: risks at the solution.

correctly removes three events that were considered high-risk at decision time, and the other removed events already were under the "high-risk" threshold. The deeper approach does not perform better in pruning events at high-risk during decision time but prunes a larger portion of low-risk events overall.

The ensemble method also offers the possibility of ranking the importance of attributes by counting the number of times each attribute is used as the best parent of the sub-models. Table 9 displays the feature relevance obtained by the ensemble model composed of deep sub-models. Risk and maximum risk scaling are the most relevant features obtained by the algorithm, as they are used in all the sub-models, and the third most relevant feature is the Mahalanobis distance. The relevance of these features coincides with the findings of GBT [14].

4.4 Gradient boosted decision trees

It is demonstrated in the paper summarizing the argument that GBT can slightly improve F_2 compared to the LRP [14]. The experiment in this section applies the same methodology as described in Ref. [14], with the difference that the number of estimators is increased to *n_estimators* = 50 and the learning rate is lowered to *learning_rate* = 0.02. The LGBMRegressor model from the LightGBM library [36] is used as a framework. For the particular data split, this model also results in a higher F_2 compared to the LRP, as shown in Tables 7 and 8. The increase in F_2 originates from a higher *precision* rather than a higher *recall*. The GBT provides better overall models for classification, but it is not designed to constrain the false negative rate, in contrast to the algorithm created in this study.

Table 9 Feature relevance from ensemble learning with sub-models of 50 generations. An explanation of these can be found in the Appendix

Rank	Feature	Fraction used
1	*risk*	1
2	*max_risk_scaling*	1
3	*mahalanobis_distance*	0.96
4	*c_sedr*	0.72
5	*c_position_covariance_det*	0.7
6	*miss_distance*	0.68
7	*t_ctdot_r*	0.68
8	*c_j2k_inc*	0.62
9	*c_obs_used*	0.6
10	*t_sigma_ndot*	0.6
11	*t_sedr*	0.58
12	*c_cr_area_over_mass*	0.56
13	*t_cndot_rdot*	0.54
14	*t_sigma_tdot*	0.54
15	*relative_position_r*	0.52
16	*t_time_lastob_start*	0.52
17	*c_residuals_accepted*	0.52
18	*t_time_lastob_end*	0.5
19	*t_cndot_t*	0.5
20	*c_time_lastob_start*	0.5

5 Conclusions

In conclusion, the present study defined a technique to cautiously prune a considerable proportion of conjunction events. It does not enable the removal of all low-risk events, and thus directly classifies an event as high-risk. Nevertheless, it can act as a first filter to determine whether it is worth executing more in-depth or resource-consuming analyses on a conjunction event. A method consisting of a projection and clustering chain was used to achieve this mean. The parameters can be selected manually or with the help of a genetic optimizer. With the latter, having more generations increases the number of pruned events in both the training and testing sets. By design, the *recall* on the training set remained nearly perfect as the number of generations increased, but the metric decreased on the testing set. This motivated the use of an ensemble learning model that aggregates different suboptimal clustering configurations. Two ensemble models were proposed and tested on the data split. One removed nearly 70% of events while removing only 3% of high-risk events, whereas the other, which was more conservative, removed 55% of the events while removing no high-risk events. It was emphasized that the methods for improving F_2 upon LRP do not necessarily do so by lowering the false negative rate.

Because of the focus on weaker models for ensemble learning, this study did not consider adaptive GAs, such as Ref. [37]. This may be a path to improve the implemented framework, along with the consideration of other dimensionality reduction and clustering techniques [15]. Other anomaly detection resources [38] may also be promising for improving this work. Future research should determine whether this algorithm, or similar safe pruning methods, can save operational resources.

Appendix

Table A1 Explanation of the dataset features referenced in this paper

Feature	Explanation
risk	Risk of collision in $\log_{10}$ base
time_to_tca	Time from CDM creation to time of closest approach (day)
max_risk_scaling	Max risk scaling factor
mahalanobis_distance	Mahalanobis distance
miss_distance	Relative position between target and chaser at TCA (m)
relative_position_r	Relative position between chaser and target in radial direction (m)

Table A2 Explanation of the object-specific dataset features referenced in this paper. The format *x_feature* is such that it refers to the target when $x = t$ and to the chaser when $x = c$. Orbit determination is abbreviated as OD

Feature	Explanation
x_j2k_inc	Inclination (deg)
x_cr_area_over_mass	Solar radiation coefficient
x_sedr	Energy dissipation rate (W/kg)
x_obs_used	Number of observations used in OD
x_residuals_accepted	OD residuals
x_time_lastob_start	Start of the time interval in days of the last accepted observation used in OD
x_time_lastob_end	End of the time interval in days of the last accepted observation used in OD
x_position_covariance_det	Determinant of the state covariance
x_sigma_t	Along-track position standard deviation (m)
x_sigma_tdot	Along-track velocity standard deviation (m/s)
x_sigma_ndot	Cross-track velocity standard deviation (m/s)
x_ctdot_r	Covariance in along-track velocity and radial position
x_cndot_t	Covariance in cross-track velocity and along-track position
x_cndot_rdot	Covariance in cross-track velocity and radial velocity

Tables A1 and A2 present the different features used in this study. A more exhaustive list of features is available on the competition website[1]. The reference frame in the data is a local (different for chaser and target) radial, along-track and cross-track frame. It is also commonly referred to as radial tangential normal (RTN) frame.

Acknowledgements

The authors thank the reviewers for their constructive comments, which have improved the manuscript.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] ESA Space Debris Office. ESA's annual space environment report. **2022**. Available at https://www.esa.int/Space_Safety/Space_Debris/ESA_s_Space_Environment_Report_2022.

[2] Klinkrad, H. *Space Debris: Models and Risk Analysis*. Springer Berlin Heidelberg, **2006**.

[3] Boley, A. C., Byers, M. Satellite mega-constellations create risks in Low Earth Orbit, the atmosphere and on Earth. *Scientific Reports*, **2021**, 11: 10642.

[4] Tao, H. C., Zhu, Q. Y., Che, X. K., Li, X. H., Man, W. X., Zhang, Z. B., Zhang, G. H. Impact of mega constellations on geospace safety. *Aerospace*, **2022**, 9(8): 402.

[5] Kessler, D. J., Cour-Palais, B. G. Collision frequency of artificial satellites: The creation of a debris belt. *Journal of Geophysical Research*, **1978**, 83(A6): 2637.

[6] Kessler, D. J. Collisional cascading: The limits of population growth in low earth orbit. *Advances in Space Research*, **1991**, 11(12): 63–66.

[7] Cornara, S., Beech, T., Belló-Mora, M., Martinez de Aragon, A. Satellite constellation launch, deployment, replacement and end-of-life strategies. In: Proceedings of the 13th Annual AIAA/USU Conference on Small Satellites, Logan, Utah, USA, **1999**.

[8] Shan, M. H., Guo, J., Gill, E. Review and comparison of active space debris capturing and removal methods. *Progress in Aerospace Sciences*, **2016**, 80: 18–32.

[9] Zhao, P. Y., Liu, J. G., Wu, C. C. Survey on research and development of on-orbit active debris removal methods. *Science China Technological Sciences*, **2020**, 63(11): 2188–2210.

[1] https://kelvins.esa.int/collision-avoidance-challenge/data/

[10] Merz, K., Virgili, B. B., Braun, V., Flohrer, T., Funke, Q., Krag, H., Lemmens, S. Current collision avoidance service by ESA's Space Debris Office. In: Proceedings of the 7th European Conference on Space Debris, Darmstadt, Germany, **2017**.

[11] ESA Space Debris Office. Proc. 8th European Conference on Space Debris (virtual).

[12] Alfriend, K. T., Akella, M. R., Frisbee, J., Foster, J. L., Lee, D. J., Wilkins, M. Probability of collision error analysis. *Space Debris*, **1999**, 1(1): 21–35.

[13] Akella, M. R., Alfriend, K. T. Probability of collision between space objects. *Journal of Guidance, Control, and Dynamics*, **2000**, 23(5): 769–772.

[14] Uriot, T., Izzo, D., Simões, L. F., Abay, R., Einecke, N., Rebhan, S., Martinez-Heras, J., Letizia, F., Siminski, J., Merz, K. Spacecraft collision avoidance challenge: Design and results of a machine learning competition. *Astrodynamics*, **2022**, 6(2): 121–140.

[15] Hastie T, TibshiraniR, Friedman J. *The Elements of Statistical Learning.* New York: Springer, **2009**.

[16] Metz, S. Implementation and comparison of data-based methods for collision avoidance in satellite operations. Master Thesis. Germany: Technische Universität Darmstadt, **2020**.

[17] Yu, Y., Si, X. S., Hu, C. H., Zhang, J. X. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Computation*, **2019**, 31(7): 1235–1270.

[18] Pinto, F., Acciarini, G., Metz, S., Boufelja, S., Kaczmarek, S., Merz, K., Martinez-Heras, J. A., Letizia, F., Bridges, C., Baydin, A. G. Towards automated satellite conjunction management with Bayesian deep learning. In: Proceedings of the AI for Earth Sciences Workshop at NeurIPS 2020, Vancouver, Canada, **2020**. Available at https://arxiv.org/abs/2012.12450.

[19] Tulczyjew, L., Myller, M., Kawulok, M., Kostrzewa, D., Nalepa, J. Predicting risk of satellite collisions using machine learning. *Journal of Space Safety Engineering*, **2021**, 8(4): 339–344.

[20] Acciarini, G., Pinto, F., Letizia, F., Martinez-Heras, J. A., Merz, K., Bridges, C., Baydin, A. G. Kessler: A machine learning library for spacecraft collision avoidance. In: Proceedings of the 8th European Conference on Space Debris, Darmstadt, Germany, **2021**.

[21] Acciarini, G., Baresi, N., Bridges, C., Felicetti, L., Hobbs, S., Baydin, A. G. Observation strategies and megaconstellations impact on current LEO population. In: Proceedings of the 2nd NEO and Debris Detection Conference, Darmstadt, Germany, **2023**.

[22] Sanchez, L., Vasile, M., Minisci, E. AI to support decision making in collision risk assessment. In: Proceedings of the 70th International Astronautical Congress, Washington D.C., USA, **2019**.

[23] Sánchez Fernández-Mellado, L., Vasile, M. On the use of Machine Learning and Evidence Theory to improve collision risk management. *Acta Astronautica*, **2021**, 181: 694–706.

[24] Chinchor, N. A., Sundheim, B. MUC-5 evaluation metrics. In: Proceedings of the 5th Conference on Message Understanding, Baltimore, Maryland, USA, **1993**: 69–78.

[25] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Müller, A., Nothman, J., Louppe, G., *et al.* Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, **2011**, 12: 2825–2830

[26] Tipping, M. E., Bishop, C. M. Mixtures of probabilistic principal component analyzers. *Neural Computation*, **1999**, 11(2): 443–482.

[27] Sculley, D. Web-scale k-means clustering. In: Proceedings of the 19th International conference on World Wide Web, Raleigh, North Carolina, USA, **2010**: 1177–1178.

[28] Grefenstette, J. J. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, **1986**, 16(1): 122–128.

[29] Schaffer, J. D., Caruana, R., Eshelman, L., Das, R. A study of control parameters affecting online performance of genetic algorithms for function optimization. In: Proceedings of the 3rd International Conference on Genetic Algorithms, George Mason University, Fairfax, Virginia, USA, **1989**: 51–60.

[30] Baeck, T., Fogel, D., Michalewicz, Z. *Evolutionary Computation 1: Basic Algorithms and Operator*s. Bristol and Philadelphia: Institute of Physics Publishing, **2000**.

[31] Vrajitoru, D. Large population or many generations for genetic algorithms? Implications in information retrieval. In: *Soft Computing in Information Retrieval. Studies in Fuzziness and Soft Computing, Vol. 50.* Crestani, F., Pasi, G. Eds. Heidelberg: Physica, **2000**: 199–222.

[32] Blickle, T., Thiele, L. A mathematical analysis of tournament selection. In: Proceedings of the 6th International Conference on Genetic Algorithms, Pittsburgh, PA, USA, **1995**: 9–16.

[33] Rokach, L. Ensemble-based classifiers. *Artificial Intelligence Review*, **2010**, 33(1): 1–39.

[34] Bonab, H., Can, F. Less is more: A comprehensive framework for the number of components of ensemble classifiers. *IEEE Transactions on Neural Networks and Learning Systems*, **2019**, 30(9): 2735–2745.

[35] Blank, J., Deb, K. Pymoo: Multi-objective optimization in python. *IEEE Access*, **2020**, 8: 89497–89509.

[36] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T. Y. LightGBM: A highly efficient gradient boosting decision tree. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, **2017**: 3149–3157.

[37] Srinivas, M., Patnaik, L. M. Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, **1994**, 24(4): 656–667.

[38] Chandola, V., Banerjee, A., Kumar, V. Anomaly detection: A survey. *ACM Computing Surveys*, **2009**, 41(3): 15.

Sébastien Henry after his bachelor's study in mechanical engineering at UCLouvain in Belgium, Sébastien obtained his master's degree in aerospace engineering at ISAE-SUPAERO in France. There, he specialized in astrodynamics and carried out a research project on collision avoidance. His master's thesis covered multiple object tracking for orbit determination, which he redacted at the French space agency, CNES. He received the "Exceptional academic achievement award" for his thesis and overall master's performance. Sébastien is now enrolled in a Ph.D. program at the Georgia Institute of Technology, USA, where he studies image-based spacecraft navigation. E-mail: seb.henry@gatech.edu

Roberto Armellin received his M.Sc. and Ph.D. degrees in aerospace engineering from Politecnico di Milano, Italy in 2003 and 2007, respectively. Since November 2020 he has been a professor at Te Pūnaha Ātea – Space Institute, University of Auckland, New Zealand. His current research interests include space trajectory optimization, spacecraft navigation and guidance, and space situational awareness. E-mail: roberto.armellin@auckland.ac.nz

Thibault Gateau received his Ph.D. degree in artificial intelligence at Toulouse University. He has been a space system engineer at ISAE-SUPAERO, Toulouse, France since 2017. He is currently the project manager of the next ISAE-SUPAERO's Cubesat mission, CREME. His research interests include planning, autonomous systems decision making, multi-agent systems, system engineering, and preliminary design optimization. E-mail: thibault.gateau@isae.fr