Robust Classification with Belief Functions and Deep Learning Applied to STM

Luis Sánchez
Aerospace Centre of Excellence
University of Strathclyde
Glasgow, G1 1XJ
United Kingdom
luis.sanchez-fdez-mellado@strath.ac.uk

Victor Rodríguez-Fernández
Applied Intelligence and Data Analysis
Universidad Politécnica de Madrid
Madrid, 28031
Spain
victor.rfernandez@upm.es

Massimiliano Vasile
Aerospace Centre of Excellence
University of Strathclyde
Glasgow, G1 1XJ
United Kingdom
massimiliano.vasile@strath.ac.uk

***Abstract*—This work proposes an approach to robust conjunction risk assessment given a sequence of Conjunction Data Messages (CDM). Dempster-Shafer theory of evidence (DSt) is used to account for epistemic uncertainty in the sequence of CDMs and derive a robust classification of conjunction events. We then use Artificial Intelligence (AI) to bypass the computationally expensive parts of DSt and directly produce a robust classification from a given sequence of CDMs. Five AI techniques are proposed: Random Forest (RF) with DSt structures, RF with CDMs, Light Gradient Boosting machine (LGBm) with CDMs, autoregressive LGBm (aLGBm), and Transformers for time series. These methods were trained and tested both on synthetic and in real datasets to study their applicability to real scenarios. The results show the potential of AI techniques, especially LGBm, for robustly classifying encounters from the sequence of CDMs, provided balanced datasets are available.**

***Index Terms*—Space Traffic Management, Transformers, Random Forest, Light Gradient Boosting machine, Dempster-Shafer.**

I. Introduction

The space environment has experienced a dramatic change during the last years due to the appearance of new mega-constellations and small satellites [1], the improvement on Space Situational Awareness (SSA) capabilities and the inclusion of thousands of new objects in the catalogues and the commercialisation of space bringing new actors in space operations [2]. The combined effect brings the Space Traffic Management (STM), designed for a different context, to the limit. The new Low Earth Orbit (LEO) environment loads operators with thousands of conjunction alerts, whose actual risk must be analysed, and if high, a detailed and cumbersome analysis to mitigate the risk is required. The STM system needs to be upgraded to quickly and automatically analyse large numbers of encounters and provide robust decisions [2].

The conjunction risk is usually evaluated using the Probability of Collision (PoC) [3], whose value is determined by the expected objects' relative position and their position uncertainty. The main driver on the PoC is the object's uncertainty, coming from errors in the observation sensors, the propagation dynamical models or the uncertainty models themselves. There are efforts to improve covariance realism [4]. However, to provide robust decision-making, it is necessary to take into account the confidence in the available information, including also epistemic uncertainty in the analysis [5].

The increase in conjunctions requires automation and faster methods to safely operate the increasing number of space objects. Artificial Intelligence (AI) appears as the right tool to handle great amounts of data, providing faster data-driven models, and assisting operators in decision-making. There are already some works initiating on the use of AI for space safety, including space agencies [6], or other examples as the use of Machine Learning (ML) to assist operators [5], to conjunction detection [7] or Conjunction Data Messages (CDMs) forecasting [8].

In this work, we bring a methodology to model mixed uncertainty on sequences of CDMs using Dempster-Shafer theory of evidence (DSt) and Dvoretzky-Kiefer-Wolfowitz (DKW) inequalities to provide robust support on decision-making tasks. CDMs are the most common protocol to communicate conjunction information among operators, but currently, they do not account for epistemic uncertainty. We propose the use of AI techniques like Random Forest (RF), Light Gradient Boosting machine (LGBm) and Transformers to automate the conjunction risk assessment given a sequence of CDMs. Current methods using CDMs for Conjunction Assessment Risk Analysis (CARA) suffer from lack of automation and high operators' work-load [9], [10] and either do not consider information from previous CDMs [9] or make strong assumptions on the CDMs time series [10]. The novelty of this paper is in simultaneously modelling the epistemic uncertainty in the sequence of CDMs and the use of AI to bypass the expensive computations required for a robustly classifying conjunction events and automatically produce a robust classification from the time series of CDMs. This novel and unique combination of AI and DSt would enable the treatment of large catalogues of events and the robust automation of STM.

The rest of the paper is structured as follows. In Section II an overview of the approach to perform robust CARA is provided. Section III presents the synthetic and real datasets with CDMs employed in the rest of the paper. Section IV explains the ML and Deep Learning (DL) architectures used. In Section V, the models' performances are compared. Finally, Section VI concludes the paper.

II. Robust Conjunction Risk Assessment

CDMs are the most common conjunction communication protocols currently used by space operators [11]. They are generated by SSA entities from the information collected from their networks of sensors (like radars and telescopes) and the propagation of the detected objects' state and uncertainty. CDMs contain a standard set of information relative to: i) metadata (identifier of the event, date of the CDM creation), ii) the encounter, like the relative geometry, the date of the encounter or the risk of the encounter, iii) the objects themselves (object identifier, type of object, state vector and covariance matrix, object observation information).

The most common conjunction assessment methodologies employing CDMs use the uncertainty information and the computed risk included in the messages, some implementing covariance realism techniques, but epistemic uncertainty is not often considered [9], [12]. In previous works, the authors proposed an approach to provide robust decision-making support quantifying the epistemic uncertainty in the CDMs [13].

This methodology uses DKW inequalities [14] and DSt [15] to model the epistemic and aleatory uncertainty in the CDMs. From each CDM, the encounter is obtained (the miss distance vector and the combined covariance matrix in the impact plane), conforming the uncertain space $\mathbf{u} \in U$, so that $\mathbf{u} = [\mu_\xi, \mu_\zeta, \sigma^2_\xi, \sigma^2_\zeta, \sigma_{\xi\zeta}]$. Each CDM is assumed to be a sample drawn from an unknown underlying distribution, from which no further assumptions are made. The method in [13] extracts information from the CDMs to generate the interval-valued variables required by DSt by creating a confidence region around all distributions compatible with the sequence's empirical Cumulative Distribution Function (eCDF).

The DKW inequalities define the proposed bounding region:

$$F_n(x) - \sqrt{\frac{\ln\frac{2}{\delta}}{2n}} \leq \mathcal{F}(x) \leq F_n(x) + \sqrt{\frac{\ln\frac{2}{\delta}}{2n}} \tag{1}$$

given n CDMs are available and the confidence level $1-\delta$ that the exact distribution $\mathcal{F}(x) \in F_n(x) \pm \varepsilon$, where $\varepsilon = \sqrt{\frac{\ln\frac{2}{\delta}}{2n}}$.

From the DKW bands, the method then proposes to obtain the p-box [16] bounding the region of compatible distributions. The p-box bounds are defined as a weighted sum of Gaussian distributions centred at the samples:

$$\mathcal{F}(x) \sim \mathcal{P}(x) = \int_{-\inf}^{\inf} \sum_i w_i \mathcal{N}(x_i, \sigma_i; x)\, dx. \tag{2}$$

The upper and lower bounds are obtained by solving the following optimisation problem:

$$\begin{gathered} \left\{ \begin{array}{l} \overline{\mathcal{P}}(x) = \max_{w_i,\sigma_i} \mathcal{P}(x; w_i, \sigma_i) \\ \underline{\mathcal{P}}(x) = \min_{w_i,\sigma_i} \mathcal{P}(x; w_i, \sigma_i) \end{array} \right. , \\ s.t. \left\{ \begin{array}{l} \overline{\mathcal{P}}(x) \leq \min(1, F_n(x) + \varepsilon) \\ \underline{\mathcal{P}}(x) \geq \max(0, F_n(x) - \epsilon) \end{array} \right. , \end{gathered} \tag{3}$$

which defines the outer distributions better approximating the DKW bands. Due to the equivalence between the p-boxes and the DSt structures [16], it is possible to discretise the p-boxes by performing horizontal α-cuts, whose intersection with the upper and lower p-box bounds define the lower and upper value of the variables intervals. The number of intervals N is equal to the number of α-cuts minus one. The height of the interval on the resulting DSt structure indicates the value of its basic probability assumption (*bpa*).

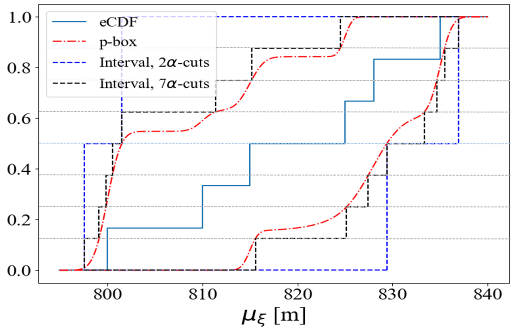

Fig. 1: Intervals partition in μ_ξ from CDM: eCDF (light blue), p-box (pointed-dashed red), DSt structure with 2 partitions (dashed black) and 7 partitions (dashed blue).

Having a set of intervals for the encounter relative geometry, the authors proposed in [5], [13] a methodology to provide robust decision-making based on the confidence of the value of the PoC. This method propagates the epistemic uncertainty on the uncertain values to the risk metric of interest (the PoC) by computing the *Plausibility* (Pl) and *Belief* (Bel) of the $PoC \geq PoC_0$. The value of the Pl at PoC_0 indicates the possibility of the encounter happening according to the available information. A greater area between the curves is associated with more epistemic uncertainty or conflict on the inputs, making it harder to make confident decisions, while a smaller gap indicates a low degree of uncertainty on the inputs.

The outcome of the methodology is a 6-fold classification indicating the operators the best action to be taken to address the conjunction according to the available information:

- *Class 0.* Perform a Conjunction Avoidance Manoeuvre (CAM) due to uncertainty on an immediate close encounter, making it impossible to make a confident decision or to collect more data.
- *Class 1.* Perform an avoidance manoeuvre due to an immediate high-risk event.
- *Class 2.* Design a CAM due to support to high-risk on a mid/long-term encounter.
- *Class 3.* Allocate new measurements and collect more information, since the uncertainty affecting the available information makes it impossible to make a confident decision in a mid/long-term conjunction.
- *Class 4.* Non-escalated event due to low support on high values of risk in a mid/long-term conjunction.
- *Class 5.* No mitigation action required due to high support for low values of risk in an immediate encounter.

Five thresholds are defined to classify the events: T_1 and T_2 for the time to the encounter, a risk threshold PoC_0 and the two epistemic thresholds Pl_0 and $A_{Pl,Bel}$ [5], [13].

However, some of the steps to robustly classify the events are time-consuming. The computation of the Pl and Bel curves on the value of the PoC requires two optimisation problems to obtain the maximum and minimum value of the PoC at each Focal Element whose number grows with N^m, where N is the number of intervals and m the number of uncertain variables ($m = 5$ in this case). A Focal Element is each of the resulting intervals on the multi-variable space with $bpa \neq 0$ resulting from the Cartesian products of the variables' intervals [5]. The higher the number of intervals, the more accurate the bounds of the underlying distribution, but the higher the computational cost. The computation of the p-boxes from the DKW bands bounding the underlying distribution, used to obtain the intervals, requires another two optimisation problems in Eq. (3), whose complexity increases with the number of CDMs in the sequence. Moreover, this step has to be repeated for each uncertain variable.

In this work, we propose ML and DL architectures to predict the evidence-based class of an encounter, given its sequence of CDMs, without explicitly executing the aforementioned expensive steps. For more information on the architecture of the framework combining epistemic uncertainty and AI, refer to [5]. In the following, the databases used to train the models and the proposed architecture are presented.

III. Databases

The architectures proposed in the next section are tested on three different databases of close encounters. One of the databases is composed exclusively of virtual encounters, while the other two correspond to encounters faced by two real mission operators by the European Space Agency (ESA). The databases include the uncertain variables derived from the CDM sequences of a number of encounters, including the object ID (if any), the encounter identification, the number of the CDM on the list, the time to the encounter from the CDM creation and the miss distance and combined covariance matrix at the impact plane corresponding to each CDM. For each new sample, that is, for each CDM in the sequence, the database associates a label indicating the class, according to the methodology presented in the previous section, accounting for all the event CDMs received up to that stage.

$$\begin{aligned} s_{gen.} =& [EventID - ObjID_1 - ObjID_2 - \\ & \#CDM - t2TCA - \mu_\xi - \mu_\zeta - \sigma^2_\xi - \sigma^2_\zeta - \sigma_{\xi\zeta}] \end{aligned} \quad (4)$$

For the rest of this work, the class in the databases is computed using the following parameters and thresholds: $\delta = 0.5$, $T_1 = 3$ days, $T_2 = 5$ days, $PoC_0 = 10^{-4}$, $Pl_0 = 1/243$, $A^*_{Pl,Bel} = 0.1$, $\underline{PoC} = 10^{-30}$ and $N = 3$. For more information, refers to Section II and [5], [13].

A. Synthetic database

A synthetic database composed of 1,000 virtual encounters was generated. The reason for using a synthetic database is twofold. First, knowing the real orbits of the objects (called nominal orbits in the remainder of the paper), provides the ground truth for each event. This means it is possible to know whether the objects are on a collision course. Second, suppose the ML models trained exclusively on the synthetic database perform well in the real ones. In that case, it is possible to integrate unbalanced real databases and create datasets tailored to the operators' necessities.

This database comprises 1,000 encounters, 50% of whom are collision scenarios. Each event was created as follows: the primary object's Keplerian elements at the nominal Time of Closest Approach (TCA) were randomly drawn from $a \in [6,850, 7,200]$km, $e \in [0, 10^{-6}]$, $i \in [0, \pi]$ rad, and $\Omega, \omega, \theta \in [0, 2\pi]$ rad. The associated nominal position is expressed with $\mathbf{x}$. The Hard Body Radius (HBR) of both objects and the nominal miss distance were drawn from $HBR \in [1, 12]$ m, $x_b \in [0.02, 200]$ m, with $x_b = \sqrt{\mu^2_\xi + \mu^2_\zeta}$, ensuring the proportion of collision/no-collision events remained balanced. The secondary object's position was derived from the miss distance and its velocity was randomly selected so that its Keplerian elements at the encounter fall within the same boundaries used for the primary object. Both objects were back-propagated assuming Keplerian motion to the first observation epoch, drawn from $[1.5, 7]$ days to TCA.

Once the nominal orbit was defined, an error was added to both objects' state vector assuming a Gaussian distribution, $\mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$, expressed in the object's $< R, T, H >$ frame (i.e. simulating the filtering process outcome from a set of observed positions). The position offset's components were drawn from the interval $[0.02, 5]$m for the collision cases and from $[0.02, 200]$m for the no collision cases, ensuring the expected miss distance, $x_0 = \|\boldsymbol{\mu}_0 - \mathbf{x}\|$, also fell within those intervals. No off-set in the velocity was considered. The covariance matrix at the first observation epoch was assumed to be diagonal with the values drawn from $\sigma_{rr,0} \in [0.1, 0.15] \cdot 10^{-4}$, $\sigma_{tt,0} \in [0.5, 0.6] \cdot 10^{-4}$, $\sigma_{hh,0} \in [0.1, 0.15] \cdot 10^{-4}$, $\sigma_{rr,0} \in [5, 6] \cdot 10^{-8}$, $\sigma_{rr,0} \in [1, 1.5] \cdot 10^{-8}$, $\sigma_{rr,0} \in [1, 1.5] \cdot 10^{-8}$, in km^2 and km^2/s^2. For each object, between 15 to 30 noisy observations were determined. The observation error was assumed to follow a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, expressed in the object's frame. One of these three alternatives was considered:

i) no position off-set was assumed in any observation (including the first one), $\boldsymbol{\mu}_i = \mathbf{0}$, and the same covariance error as in the first observations was assumed at every observation, $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}_0$;

ii) the same observation off-set of the first observations was assumed for the rest of the observations, $\boldsymbol{\mu}_i = \boldsymbol{\mu}_0$, while the same covariance matrix was assumed for each observation ($\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}_j$), derived from a rotated and resized version of $\boldsymbol{\Sigma}_0$;

iii) at each observation, a different off-set and covariance matrix were assigned ($\boldsymbol{\mu}_i \neq \boldsymbol{\mu}_j$, $\boldsymbol{\Sigma}_i \neq \boldsymbol{\Sigma}_j$), with $\boldsymbol{\mu}_i$ obtained in the same way as $\boldsymbol{\mu}_0$, and $\boldsymbol{\Sigma}_i$ being a rotated and resized version of $\boldsymbol{\Sigma}_0$.

From each observation epoch, the ellipsoids were propagated with Monte Carlo sampling to the TCA and projected onto the impact plane and fitted to a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{TCA}, \boldsymbol{\Sigma}_{TCA})$ to obtain the corresponding CDM.

TABLE I: Class distribution on the three databases.

	Cl. 0	Cl. 1	Cl. 2	Cl. 3	Cl. 4	Cl. 5
Synth. (%)	25.3	46.6	10.0	6.03	5.07	7.00
Real 1 (%)	0.46	0.03	0.01	10.7	30.9	57.9
Real 2 (%)	0.48	0.01	0.01	10.5	31.3	57.7

A total of 17,051 messages were generated, from which 52.6% corresponds to collision scenarios and the other 47.4% to no-collision cases whose robust-classes distribution is in Table I. The database is relatively well distributed, although some classes (especially, Classes 3, 4 and 5) present a lower percentage of samples. This effect is due to the epistemic threshold selection, which moves some low-risk cases to the uncertainty classes. Nevertheless, the threshold tuning falls out of the scope of this work.

B. Real databases

Two real databases are used in this work. They correspond to two different missions operated by ESA in the same LEO region (within the boundaries of the synthetic database). The databases are compounded by the CDMs received and analysed by the ESA's Space Debris Office (SDO) during the period 2015-2022. The objective of the real databases is double: first, to analyse the performance of the ML models trained on the synthetic database; second, to study the prediction capacity of the model when applied to another mission, by training the models in one of the two real databases and making predictions on the other.

The first real database contains 36,071 encounters and a total of 239,521 CDMs. The second real database contains 36,160 events and a total of 249,943 CDMs. The class distribution for both databases after the robust conjunction analysis appears in Table I.

The main characteristic of the real databases is the marked unbalanced structure, where the immense majority of cases correspond to low-risk scenarios, with only a handful of events falling in classes associated with high-risk or CAM execution.

C. Database split

To avoid overfitting, during the training stage, we decided to split the databases into a training set (80%) and a test set (20%). The test set remained unseen for the models during the training phase and is only used to make predictions and compare the models' performance. Thus, the same test split is shared by all the models. A further split was made during training, so 80% of the samples on this subset are used to adjust the model parameters, while the remaining 20% forms the validation set, which is used to evaluate the trained model and perform the hyperparameter search. Both subsets on the training set keep the same class distribution on these two sets.

Since from each event, several samples are obtained (one sample per each new CDM on the sequence), to avoid having two very similar samples coming from the same event in the training and the test set, the division is made at an encounter level, splitting by the *Event ID*. The training set is compounded by the 80% of the events, which is close, but not necessarily equal, to the 80% of the samples due to the difference in the CDM sequences lengths.

TABLE II: Class distribution on the training/test sets.

	Cl. 0	Cl. 1	Cl. 2	Cl. 3	Cl. 4	Cl. 5
Synth Train. (%)	25.9	45.6	10.3	6.13	4.98	7.09
Synth Test (%)	23.2	50.4	8.64	5.66	5.09	7.01
Real Train. (%)	0.46	0.02	0.02	10.7	30.8	58.0
Real Test (%)	0.54	0.05	0.01	10.7	31.1	57.6

Table II shows the distribution of classes in both training and test sets for the synthetic database and one of the real databases. The remaining database is not divided since it is not used for training, but only for evaluating the capabilities of the models to predict on a database from a different mission.

IV. Model Architectures

This section presents the different architectures analysed in this work to predict the class obtained with the evidence-based classification criterion presented in Section II. Five different architectures are proposed, two of them using RF, another two using LGBm and a last one employing Transformers. From previous works in similar classification problems for conjunction risk assessment, [5], the authors identified that RF outperformed other classification techniques like Support Vector Machine and k-Nearest Neighbours. RF is set as a fast and robust baseline. LGBm is selected for its capacity to handle continuous and categorical features in tabular data as well as for its higher speed relative to RF. Finally, Transformers are used due to their potential to address the problem as a time-series classification, which comes from their success in handling sequences in other fields such as Natural Language Processing.

RF [17] is an ensemble method that combines several independent Decision Trees during the training step, feeding each of them with different subsets of the training set. The predicted class is the mode of the output of every single tree. This ensemble approach allows for overcoming the overfitting and bias problem presented by Decision Trees while maintaining the simple architecture.

LGBm [18] is a variant of the Gradient Boosting methods, also based on the ensemble of Decision Trees (*Boosting*). LGBm presents the advantages of other Gradient Boosting models, like the simplicity of implementation, and the reduced number of parameters required to be tuned, but it allows for faster training and higher accuracy. The main difference to other Decision Trees-based algorithms is that it does not present a level-wise growth (growing a new row from the previous nodes at a time), but a more efficient leaf-wise growth, where only the most promising node generates a new row. One important advantage to other architectures is that it accepts simultaneously continuous and categorical variables without any pre-processing.

The Transformer architecture [19], originally proposed for the task of machine translation in the field of Natural Language Processing (NLP) and now applied to a wider range of tasks, leverages the ideas from attention-based models and proposes to construct a model to process sets and sequences by using only an attention mechanism between a data encoder and a decoder. The term "attention mechanism" in neural networks is used to represent a specific class of algorithms, in which the model looks at each element of the sequence in turn, and compares it to every other point, attempting to determine the most relevant part of the sequence for each point. This overcomes the limitation of local connectivity, at the cost of the quadratic complexity that the attention matrix has in terms of memory usage. The network implemented here follows the structure used in [20].

A. Random forest with intervals

This architecture uses RF to classify the events using the uncertain variables intervals bounds, allowing skipping the explicit computation of the Pl and Bel (Section II). It still requires the derivation of the p-boxes from the CDMs.

RF requires tabular inputs with the same length. Since sequences of CDM have different lengths, this approach takes advantage of the tabular format of the DSt structures after performing the α-cuts. However, the number of inputs grows with the number of cuts, thus a trade-off between accuracy (of the DSt structure) and complexity (of the input data) should be achieved. As indicated before, in this work, two α-cuts per variable are performed, thus $N = 3$ for each variable.

The features are structured for this architecture so they take the time to the TCA and, for each of the uncertain variables, the lower and upper bound of each interval and its *bpa*. Thus, the number of features is equal to: $\#feat_{RF,int} = 1+3Nm = 1 + 3 \times 3 \times 5 = 46$, with N the number of intervals and m the number of uncertain variables.

$$\begin{aligned} s_{RF,int} =&[\underline{\mu_{\xi,1}}, \overline{\mu_{\xi,1}}, bpa_{\mu_\xi,1}, \underline{\mu_{\xi,1}}, \overline{\mu_{\xi,2}}, bpa_{\mu_\xi,2}, ..., \\ &\underline{\sigma^2_{\xi,1}}, \overline{\sigma^2_{\xi,1}}, bpa_{\sigma^2_\xi,1}, ...] \end{aligned} \tag{5}$$

The method was implemented using Python's "scikit-learn" library. A hyper-parameters search was carried out among the values included in Table IV. The rest of the arguments took the default values, including the loss function (cross-entropy).

B. Random forest with CDMs

This architecture also uses RF, but it takes directly the information from the CDMs, skipping the two computationally expensive optimisation steps. To avoid the different lengths of the sequences, a lag window is used. Only a certain number of CDMs previous to the latest one in the sequence are selected, solving the problem of the tabular inputs, at the cost of losing some information on the sequence. This allows training the model on a database with more accurate classes obtained from a finer p-box partition without increasing the number of inputs. However, to compare the different models' predictions across the different alternatives in this work, the same partition of 3 intervals partition as in the previous case was used here.

The structure of the features takes the last CDM uncertain variables and the time to the encounter plus the same variables of the previous CDMs included on the lag window: $\#feat_{RF,lag} = (m+1)(l+1)$, where l is the lag window and $l = 0$ meaning only the last CDM is considered. In Table III, an example of the sample can be seen (note the last column is not used in this case).

The same implementation, hyperparameters and loss function as in the previous model were employed.

C. LGBm with CDMs

The same approach is followed here, but using LGBm architecture instead. The inputs and output are the same (Table III). Thus, the influence of the model can be analysed as a mid-step between the previous and the next approaches.

The method was implemented using Python's "LightGBM" library, following the same approximation as in the previous scenarios. The set of hyperparameters considered in the search appears in the Table IV, with the rest of the argument's values set as default, with cross-entropy as loss function.

D. Autoregressive LGBm with CDMs

This approach follows a similar approach to the previous one, but instead, it uses the previous class as a feature. Due to the possibility of combining numerical and categorical features in the input data, this alternative applies a sort of autoregressive implementation, including the class to be predicted in the previous time series instance as a feature.

To have a tabular structure on the input data, a lag window is also employed to take the information from the last and the previous l CDMs, and additionally, the class associated with those previous cases. Thus, the number of features is equal to $\#feat_{LGBm} = m + (m+1)l$. In Table III, an example of the sample can be seen, including the $Class_{t-1}$ column.

The aim of this *autoregressive* technique is to include the sequential character of the inputs, expecting that the classification is influenced by the incremental amount of information received with the new CDMs. With this approach, some of the information from previous CDM lost with the lag window is expected to be recovered, since the previous class implicitly contains information from the whole sequence. The same implementation as the previous case was followed.

E. Transformers with CDMs time series

Finally, this last proposal implements Transformer architecture. to classify the time series of CDMs. As in the two previous implementations, from an event with N CDMs, $N-1$ samples were extracted (at least two samples are required to perform the proposed methodology). However, in this case, each sample is compounded by a set of eight time series corresponding to the five uncertain variables, the time to the encounter, the previous class (categorical) and a padding flag. To have regularly spaced time series of equal length, the samples in the time series were sorted according to their index on the series, including the time as a feature (time to the encounter), and the length of the time series was set equal to the maximum length, filling shorter times series with a padding value. The padding flag indicates if the value comes from the CDM or is a filling value. The variables are normalised before

TABLE III: Synthetic samples with lag step 1 used by the LGBm architectures and the RF with CDM approach. The $Class_{t-1}$ columns would be used only by the aLGBm method. Units in m, m^2 and $days$.

#Sample	$\mu_{\xi,t}$	$\mu_{\zeta,t}$	$\sigma^2_{\xi,t}$	$\sigma^2_{\zeta,t}$	$\sigma_{\xi\zeta,t}$	$t2TCA_t$	$\mu_{\xi,t-1}$	$\mu_{\zeta,t-1}$	$\sigma^2_{\xi,t-1}$	$\sigma^2_{\zeta,t-1}$	$\sigma_{\xi\zeta,t-1}$	$t2TCA_{t-1}$	$Class_{t-1}$
0	37.51	$7{\cdot}10^{-11}$	$9.4{\cdot}10^5$	$5.1{\cdot}10^5$	$-6.9{\cdot}10^5$	2.79	177.1	$-5{\cdot}10^{-10}$	$1.8{\cdot}10^6$	$2.6{\cdot}10^4$	$2.1{\cdot}10^5$	2.86	-
1	25.37	$4{\cdot}10^{-10}$	$3.4{\cdot}10^5$	$1.6{\cdot}10^6$	$6.3{\cdot}10^5$	2.64	37.51	$7{\cdot}10^{-11}$	$9.4{\cdot}10^5$	$5.1{\cdot}10^5$	$-6.9{\cdot}10^5$	2.79	1
2	22.59	$-7{\cdot}10^{-11}$	$1.7{\cdot}10^{-5}$	$8.5{\cdot}10^5$	$-3.8{\cdot}10^5$	2.36	25.37	$4{\cdot}10^{-10}$	$3.4{\cdot}10^5$	$1.6{\cdot}10^6$	$6.3{\cdot}10^5$	2.64	1
3	32.99	$4{\cdot}10^{-10}$	$5.9{\cdot}10^5$	$4.7{\cdot}10^5$	$5.3{\cdot}10^{-10}$	2.23	22.59	$-7{\cdot}10^{-11}$	$1.7{\cdot}10^{-5}$	$8.5{\cdot}10^5$	$-3.8{\cdot}10^5$	2.36	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...

TABLE IV: Set of hyperparameters considered to select the best model during training.

Random Forest		Light GB machine		Transformer	
Hyperparam.	**Values**	**Hyperparam.**	**Values**	**Hyperparam.**	**Values**
`n_estimators`	{50, 100, 200, 400}	`n_estimators`	{1, 2, 5, 10}	`depth`	{1, 2, 3, 4, 5}
`max_depth`	{`None`, 50, 100}	`max_depth`	{2, 7, 10, 15}	`attn_dropout`	{0, 0.1, 0.2, 0.3, 0.5 }
`min_samples_split`	{2, 20}	`subsample`	0.7	`res_dropout`	{0, 0.1, 0.2, 0.3, 0.5 }
`min_samples_leaf`	{10^{-7}, 10^{-4}, 1}	`colsample_bytree`	0.8	`wd`	{0, 0.1, 0.3}
`max_features`	{`'auto'`,`'log2'`,0.5}	`boosting_type`	{`'gbdt'`,`'rf'`}	`n_epochs`	{10, 25, 50}

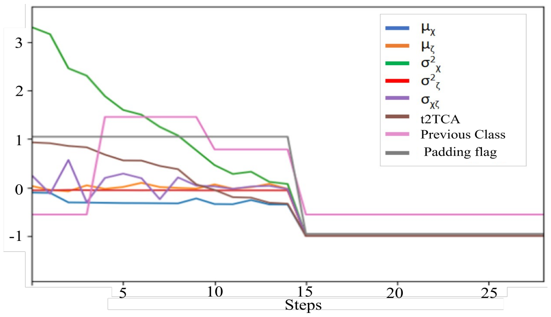

Fig. 2: Multi-channel time series synthetic sample for the transformer, including padding and the padding flag.

being fed to the network. For more details on the algorithm's architecture, refer to [20].

The model was implemented using Python's "tsai" library. The models are compared using the F2-score metric. For the optimiser, we use Ranger, an extension of the Adam optimiser [20]. Weights and momentum are instantiated as default, the loss function used is also cross-entropy and a variable "one-cycle" learning rate was applied.

F. Training and hyperparameter search

The same hyperparameter search process was performed on each architecture during training. The process performed a random search on the space defined in Table IV. The model defined by the selection of hyperparameters was trained on the training set and evaluated on the validation set. The process was repeated for several combinations of hyperparameters, and the best model was selected as the one performing better on the validation set. Due to the class imbalance and the higher importance of avoiding miss encounters than the false alert, the performing metric used was the average F2 score, $\overline{F_2}$,

$$\overline{F_2} = \frac{\Sigma_i^N F_{2,i}}{N}, \quad F_{2,i} = \frac{5{\cdot}precision_i{\cdot}recall_i}{4{\cdot}precision_i+recall_i}, \tag{6}$$

being N the number of labels. For the architectures using a lag window, the best model for different window lengths was saved for further analysis in the next section.

V. Results

In this section, the different models' performances across the different databases, having been trained on the synthetic and the real databases, are compared. In the first case, the different models were trained on the synthetic training set and their performance was evaluated both in the synthetic test set and in one of the real databases. In the second case, the models were trained in the training set of the other real set and evaluated both in the test set of that real database and in the whole remaining real dataset (the same as in the first case).

A. Training on synthetic database

The right side of Table V shows the performance of the different approaches trained on the synthetic set both evaluated on the synthetic test set (upper tier) and the real database (lower tier). The prediction on the synthetic test set, with similar characteristics to the training set, presents a generally good F2 score, both overall and by classes. RF fed with intervals has good prediction capabilities, since the inputs are some steps closer to the output in the underlying model. However, the autoregressive LGBm (aLGBm) and the Transformer (which also include the previous class among the inputs) match or improve those results, even though they are fed directly with the CDMs. Attending to the score by class, there is a slightly better score in the more populated categories, although good prediction capabilities are obtained across the classes. In any case, the synthetic database does not present a sharp imbalance trend. Nevertheless, it seems that an equally distributed and enough populated database could level those scores.

However, the application to a real database did not provide good results. The aLGBm still provides the best results, also when applied to a database different than the one used for training, although with a score some points below the previous case. While the populated classes still score well, the scarcely populated categories are not well predicted. This pattern is repeated across all the methods, including also the RF with intervals, despite the less complex model required. This method scores especially well on labels 4 and 5, but very poorly on the others. Surprisingly, the transformer does

TABLE V: F1 and F2-scores of the best AI model, trained on the synthetic (left) or the real (right) sets, and evaluated in the test set (upper-tier) or the remaining real set (lower-tier). In bold, the model with the highest overall F2-score. Underlined, the best F2-score by class.

		Trained on synthetic set							**Trained on real set**						
		Overall	**Cl. 0**	**Cl. 1**	**Cl. 2**	**Cl. 3**	**Cl. 4**	**Cl. 5**	**Overall**	**Cl. 0**	**Cl. 1**	**Cl. 2**	**Cl. 3**	**Cl. 4**	**Cl. 5**
RF interv. (test set)	F1	.847	.889	.958	.891	.851	.752	.834	.632	.607	.000	.522	.998	.999	.997
	F2	.854	.902	.947	.868	.841	.805	.873	.663	.637	.000	.731	.999	.998	.997
RF CDMs (test set)	F1	.723	.785	.948	.849	.752	.703	.513	.484	.352	.000	.000	.905	.902	.907
	F2	.748	.780	.933	.820	.798	.803	.581	.509	.436	.000	.000	.954	.936	.924
LGBm CDMs (test set)	F1	.760	.760	.940	.810	.763	.711	.599	.463	.000	.000	.000	.896	.898	.949
	F2	.759	.788	.930	.766	.798	.774	.449	.448	.000	.000	.000	.950	.935	.922
aLGBm (test set)	F1	.871	.891	.959	.876	.864	.830	.854	.637	.770	.485	.000	.897	.893	.949
	F2	.879	.916	.948	.883	.859	.838	**.847**	.640	.835	.434	.000	.949	.926	.924
Transformer (test set)	F1	.899	.908	.967	.898	.871	.829	.850	.725	.828	.400	.000	.966	.949	.990
	F2	**.888**	**.896**	**.973**	**.891**	**.877**	.818	**.861**	**.732**	**.807**	**.294**	**.000**	**.961**	**.935**	**.981**
RF interv. (real)	F1	.536	.119	.039	.014	.881	.940	.944	.546	.475	.000	.000	.998	.999	.996
	F2	.523	.079	.024	.009	.823	.975	.958	.566	.578	.000	.000	.998	.999	.995
RF CDMs (real)	F1	.218	.024	<.001	.061	.551	.593	.385	.467	.169	.000	.211	.886	.885	.939
	F2	.270	.015	<.001	.043	.459	.766	.599	.489	.250	.000	.322	.944	.930	.908
LGBm CDMs (real)	F1	.394	.033	.004	.005	.831	.847	.777	.429	.000	.000	.000	.879	.884	.940
	F2	.403	.020	.003	.008	.830	.915	.806	.441	.000	.000	.000	.944	.931	.907
aLGBm (real)	F1	.612	.691	.039	.026	.908	.895	.952	.561	.762	.109	.000	.883	.880	.940
	F2	**.601**	**.672**	**.026**	**.016**	**.907**	**.903**	**.950**	.579	.824	.098	.000	.944	.923	.910
Transformer (real)	F1	.347	.224	.003	.137	.504	.555	.687	.609	.581	.823	.000	.927	.953	.970
	F2	.351	.220	.009	.236	.548	.595	.608	**.618**	**.549**	**.898**	**.000**	**.914**	**.949**	**.963**

not perform well when applied to a different database than the virtual set. Such an imbalanced database, with very few high-risk cases and so differently distributed from the training set is, at least partially, behind those poorer results. It is the belief of the authors, given the score in the synthetic database, that a more equally distributed database should provide better scores. Regarding the scarcity of real high-risk data, a potential approach is to evaluate the model during training in a validation set simulating the distribution on the real set, so it prioritises models scoring high on such databases.

From Fig. 3, the score is indifferent to the lag step. The window length has little influence on the performance, even though it adds more information to the model. For the RF (green) and LGBm (red) using the CDMs, the score both in the synthetic validation and the real sets is almost constant, with the LGBm approach performing slightly better. However, when attending to the aLGBm (blue), adding a 1-step lag window significantly increases the performance, but longer windows have no effect. This allows the conclusion that adding the previous class is what improves the model prediction capabilities.

B. Training on the real database

This case shows the analysis when training on the real set. On the left side of Table V, the performance of the models on the validation set of the real database (upper tier) and in the whole dataset of the other mission (lower tier) are shown.

The performance of the validation is greatly affected by the imbalance in the dataset. The overall F2 score is lower than when trained on the synthetic database, affected by the poor performance in the less populated categories, especially Class 1 and Class 2, where some methods are not able to predict any sample. Nevertheless, the aLGBm and the transformer are the better models, performing well along all the classes (including Class 0), except the least populated.

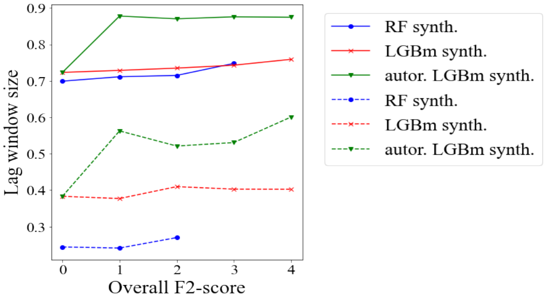

Fig. 3: Overall F2 score as a function of the lag window length. Solid: train and evaluated in synthetic set; dashed: train in synthetic evaluated on the real sets.

More interestingly, when comparing the performance of the models trained on the synthetic dataset and validated in the same real set, the performance is better for these two techniques, aLGBm and the transformer, especially the latest. Although not achieving the same scores as when validating in the synthetic dataset due to the imbalance character of the real set, they achieve almost similar results as when predicting on the test set in the previous scenario. Moreover, the performance of the transformer when applied to the other real set improves from when trained on the synthetic set and validated on the real set. It achieves good scores except for Class 2, not being able to predict any sample as the rest of the architectures. Again, a better-distributed database for training is likely to improve scores on the less populated categories.

C. Computational time analysis

Table VI shows a comparative of the computational time saved by the different AI approaches with respect to using the model-based approach for a case with 15 CDMs and 3 α-cuts. It can be seen that proposed approaches save time by skipping specific steps: the derivation of the p-boxes (except for the RF

TABLE VI: Computational time (in seconds) for robust CARA with and without using AI techniques.

	p-box	Pl/Bel	Class.	Total
No AI	75.4	2.37 ($\sim$150)	0.02	78.1 ($\sim$225)
RF inter.	75.4	-	0.06	75.8
RF CDMs	-	-	0.15	0.16
LGBm CDMs	-	-	0.03	0.03
aLGBm	-	-	0.03	0.03
Transf.	-	-	0.07	0.08

with intervals) and the computation of the Pl and Bel curves. The required time is one or two orders of magnitude smaller when using the ML approaches directly with the sequences. Note that, if increasing the number of α-cuts, the time required for the computation of Pl and Bel increases significantly (in the table, indicated in parenthesis), while the classification with AI-based approached remains indifferent.

VI. Conclusions

This work presented an AI-based approach to robustly classify space conjunctions provided the sequence of CDMs. Starting from an evidence-based methodology using DSt to model aleatory and epistemic uncertainty, the proposed approach was demonstrated to rapidly classify conjunction events with a good degree of reliability. In doing so it provides valuable decision support to operators. The gains in computational speed and the good levels of accuracy suggest that employing ML as surrogate models can be used to automate STM tasks.

The aLGBm technique, including the previous sample class as a feature, gave the best results across the different databases. The transformer, which employs a novel encoding of the time series information, showed some good results and good potential to be applied to a dataset from a mission different from the one used for training. However, more work to find the right set of hyperparameters is required. Using RF with the DSt intervals gets similar results in some cases as the other two techniques. However, it requires computing the DSt structures, which can be computationally expensive.

We used both synthetic and real databases. Although the performance in the synthetic dataset is excellent, the applicability to a very differently structured database was shown to be affected by the imbalance in the real datasets. Furthermore, the similarity between the time series of the synthetic and real database needs to be improved to allow one to train on the synthetic database and apply the model to a real dataset. Further work to improve performance on the real database is underway, including improving the techniques and models, obtaining more data, applying data augmentation and balancing techniques to the datasets, or combining the strengths of the different methods with an ensemble approach.

Acknowledgement

The authors would like to thank ESA's Space Debris Office for facilitating the real databases in the scope of the OSIP "Idea I-2019-01650: Artificial Intelligence for Space Traffic Management", and MIT's Aerospace Robotics, and Controls Lab for hosting the authors during the research of this work.

References

[1] European Space Agency - Space Debris Office, ESA/ESOC, "ESA's annual space environment report," September 2023.

[2] G. Peterson, M. Sorge, and W. Ailor, "Space Traffic Management in the age of New Space," *Center for Space Policy and Strategy*, 2018. The Aerospace Corporation.

[3] R. Serra, D. Arzelier, M. Joldes, J. Lasserre, A. Rondepierre, and B. Salvy, "Fast and accurate computation of orbital collision probability for short-term encounters," *Journal of Guidance, Control, and Dynamics*, vol. 39, pp. 1–13, 2016.

[4] A. B. Poore, J. M. Aristoff, J. T. Horwood, R. Armellin, W. T. Cerven, Y. Cheng, C. M. Cox, R. S. Erwin, and J. H. Frisbee, "Covariance and uncertainty realism in space surveillance and tracking," Technical report, Numerica Corporation Fort Collins United States, Washington, DC, USA, 2016.

[5] L. Sánchez and M. Vasile, "On the use of machine learning and evidence theory to improve collision risk management," *Acta Astronautica*, vol. 181, pp. 694–706, 2021.

[6] T. Flohrer, H. Krag, K. Merz, and S. Lemmens, "CREAM - ESA's Proposal for collision risk estimation and automated mitigation," in *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, Maui, Hawaii, US, 17-20 September 2019.

[7] E. Stevenson, V. Rodríguez-Fernández, E. Minisci, and D. Camacho, "A deep learning approach to solar radio flux forecasting," *Acta Astronautica*, vol. 193, pp. 595–606, 2022.

[8] G. Acciarini, F. Pinto, F. Letizia, J. Martínez-Heras, K. Merz, C. Bridges, and A. Güneş Baydin, "Kessler: a machine learning library for spacecraft collision avoidance," in *8^{th} European Conference on Space Debris*, ESA/ESOC, Darmstadt, Germany, 12-14 April 2021.

[9] K. Merz, V. Braun, B. Bastida Virgili, T. Flohrer, Q. Funke, H. Krag, and S. Lemmens, "Current collision avoidance service by ESA's Space Debris Office," in *7^{th} European Conference on Space Debris*, ESA/ESOC, Darmstadt, Germany, 18-21 April 2017.

[10] F. Laporte and M. Moury, "CAESAR, French probative publics service for in-orbit collision avoidance," in *6^{th} European Conference on Space Debris*, ESA/ESOC, Darmstadt, Germany, 22-25 April 2013.

[11] "Recommendation for space sata system standards. Conjunction Data Message," Recommended Standard, CCSDS, Washington, DC, USA, June 2013.

[12] L. Newman, M. Hejduk, R. Frigm, and M. Duncan, "Evolution and implementation of the NASA robotic conjunction assessment risk analysis concept of operations," in *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, Maui, Hawaii, US, 9-12 September 2014.

[13] L. Sánchez, M. Vasile, S. Sanvido, K. Merz, and C. Taillan, "Treatment of epistemic uncertainty in conjunction analysis with Dempster-Shafer theory." *Journal of Guidance, Control and Dynamics*, submitted.

[14] A. Dvoretzky, J. Kiefer, and J. Wolfowitz, "Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator," *Annals of Mathematical Statistics*, vol. 27, no. 3, pp. 642–669, 1956.

[15] G. Shafer, *A Mathematical theory of evidence*. Princeton, NJ: Princeton University Press, 1 ed., 1976.

[16] S. Ferson, V. Kreinovich, L. Ginzburg, K. Sentz, and D. Myers, "Constructing probability boxes and Dempster-Shafer structures," tech. rep., Sandia National Lab., Albuquerque, NM, United States, 2003.

[17] L. Breiman, "Random forest," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[18] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Liu, "Lightgbm: a highly efficient gradient boosting decision tree," in *31^{st} Conference in Neural Information Processing Systems (NIPS)*, vol. 30, p. 3149–3157, 2017.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, and A. Gomez, "Attention is all you need," in *31^{st} Conference in Neural Information Processing Systems (NIPS)*, vol. 30, p. 5998–6008, 2017.

[20] E. Stevenson, R. Martínez, V. Rodríguez-Fernández, and D. Camacho, "Predicting the effects of kinetic impactors on asteroid deflection using end-to-end deep learning," in *2022 IEEE Congress on Evolutionary Computation (CEC)*, Padova, Italy, 18-21 April 2022.

TABLE VII: F1 and F2-scores with ensemble, trained on the synthetic (left) or the real (right) sets, and evaluated in the test set (upper-tier) or the remaining real set (lower-tier). In bold, the model with the highest overall F2-score. Underlined, the best F2-score by class.

		Trained on synthetic set							**Trained on real set**						
		Overall	**Cl. 0**	**Cl. 1**	**Cl. 2**	**Cl. 3**	**Cl. 4**	**Cl. 5**	**Overall**	**Cl. 0**	**Cl. 1**	**Cl. 2**	**Cl. 3**	**Cl. 4**	**Cl. 5**
Ensemble (test set)	F1	.832	.839	.952	.851	.822	.794	.738	.514	.328	.000	.000	.902	.901	.950
	F2	.824	.840	.944	.843	.853	.822	.748	.490	.530	.000	.000	.000	.958	.923
Ensemble (real)	F1	.407	.028	.003	.007	.848	.861	.697	.474	.131	.000	.000	.887	.886	.940
	F2	.386	.018	.002	.004	.878	.921	.838	.456	.263	.000	.000	.951	.931	.908