Learning Real Trajectory Data to Enhance Conflict Detection Accuracy in Closest Point of Approach

Zhengyi Wang, Man Liang, Daniel Delahaye, Weilu Wu

Thirteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2019)

Learning Real Trajectory Data to Enhance Conflict Detection Accuracy in Closest Point of Approach Problem

Zhengyi WANG *
Ecole Nationale
de l'Aviation Civile
Toulouse, France
zhengyi.wang@alumni.enac.fr

Man LIANG
University of
South Australia
Adelaide, Australia

Daniel DELAHAYE
Ecole Nationale
de l'Aviation Civile
Toulouse, France

Weilu WU
Ecole Nationale
de l'Aviation Civile
Toulouse, France

***Abstract*—Closest Point of Approach (CPA) is one of the main problems in aircraft Conflict Detection (CD). It aims to find out the minimum distance and the associated time between two aircraft on the same altitude with crossing traffic. Conventional CPA prediction model generally assumes that the speed and heading of the aircraft are constant. But the uncertainties in real operations lead to the inaccuracy of CPA prediction. In this paper, we introduce a novel CD framework with Machine Learning (ML) methods. It aims to improve the CPA prediction accuracy with the help of real trajectory data. The new model contributes to not only reduce the number of fault short-mid term conflict alert for air traffic controllers but also support the implementation of future free flight concept, so as to reduce fuel consumption and emission. In our study, we firstly propose a data processing method to generate a close-to-reality simulation data from Mode-S observations. Then, feature engineering is used to transform the raw data into suitable features, which will enable the ML models to make predictions with high-performance. Six prevailing ML methods (MLR, SVM, FFNNs, KNN, GBM, RF) are used to predict the CPA time and distance. Their prediction results are compared with the conventional CPA model (baseline). The simulation results demonstrate that the GBM is the best prediction model both in CPA prediction and conflict detection. However, the results also prove that not all ML models outperform the baseline CPA model. Suitable ML methods can greatly enhance the accuracy of conflict detection.**

***Keywords*—Air traffic management, Conflict detection, Closest Point of Approach, Machine Learning**

1. INTRODUCTION

Free flight concept is a crucial research topic in future Air Traffic Management (ATM). Aircrafts in free flight eliminate the needs for Air Traffic Control (ATC) by providing pilot with the freedom to select their path and speed in real time [1]. Nevertheless, the conflict between aircraft must be detected and resolved before achieving this goal. By definition, conflict occurs when the distance between two aircraft violates the minimum allowed separation. To ensure the safety and efficiency of free flight, automated aircraft Conflict Detection (CD) is of great significance. The implementation of effective CD contributes to not only reduce the number of fault short-mid term conflict alert for Air Traffic Controllers (ATCO) in reality, but also support the implementation of future free flight concept, so as to reduce the fuel consumption and CO_2 emission.

CD is generally studied at three different levels: long-term, mid-term and short-term [2]. In long-term, CD involves trajectory planning and airline scheduling, which are the first operations to avoid unnecessary conflicts and to ensure flight safety. Mid-Term Conflict Detection (MTCD) is usually carried out by ATCO with the help of semi-automated tools over a time horizon of tens of minutes. The frequently used tools include Center TRACON Automation System (CTAS) [3] and User Request Evaluation Tool (URET) [4], etc. In view of short-term CD, the time scale is in seconds or minutes. The detected conflict must be dealt with immediately, otherwise it will cause severe accident. To assist ATCO and pilots, Short Term Conflict Alert (STCA) and Traffic Collision Avoidance System (TCAS) [5] are developed and applied in short-term CD. In order to effectively improve the accuracy of CD and ensure enough time for conflict resolution, in this paper, we mainly focus on the MTCD with the lookahead time of 5-20 minutes.

Closest Point of Approach (CPA) is a key concept in algorithmic aspects for 2D MTCD. It aims to determine the minimum distance and the associated time between two aircraft at the same altitude with crossing traffic. CPA is firstly studied in maritime domain for vessel CD [6, 7]. Then, plenty of research has been carried out on using CPA for aircraft CD [8–15]. In practical use of MTCD for ATM, CPA concept is applied in Eurocontrol's MTCD tool, FAA AERA-2 tool, URET, etc. [14] However, CPA is problematic in the actual MTCD application. In conventional and theoretical CPA calculation, it is assumed that each aircraft flies in a straight trajectory with a constant velocity vector. Actually, the aircraft may change or intend to change the heading throughout the flight. Even in cruise phase, there are still some minor changes in heading. Besides, the ground speed is the sum of airspeed and wind speed vectors. These vector components both have high-level uncertainties, especially for

wind speed. Due to the stochastic climate change and the limitation of wind modelling strategy, the wind prediction bias is large and increases with time. Therefore, the traditional CPA calculation method frequently lacks accuracy in the real world with high rate of false alarms and missed detects [14].

In this paper, we will develop an novel MTCD framework with Machine Learning (ML) methods. It is able to enhance the conflict detection accuracy in CPA problem by learning from the real trajectory data. Firstly, we propose a simple data processing method to generate the close-to-reality dataset from easily accessible mode-S observations. The generated dataset ensure that all aircraft are flying freely with no further conflict resolution maneuvers. Unlike other fast-time and Monte Carlo simulated dataset, our dataset is generated based on real trajectories, so it is easy to extract the aircraft intents, and specify the airspace, aircraft type, flight phase, etc. Then, feature engineering is conducted to transform raw data into suitable features for prediction models. Finally, several prevailing ML models will be applied for CPA prediction and conflict detection. To the best of our knowledge, Most CD researches used non-actual data, such as simulation data and ideal data. This study will provide a novel insight of CD based on real trajectory data.

2. Conflict detection with Closest Point of Approach

CPA refers to the positions at which two dynamically moving objects reach their closest possible distance [16]. In the air, conventional CPA model assumes that aircraft fly at the same altitude with constant speeds and constant headings [14, 15]. Figure 1 illustrates the interaction of two aircraft. This scenario occurs from t_0 to t_3. Dotted lines indicate the distance between two aircraft at various timestamps. The timestamp is noted as t_{CPA}, when two aircraft reach their closest distance d_{CPA} labelled by red dotted line.

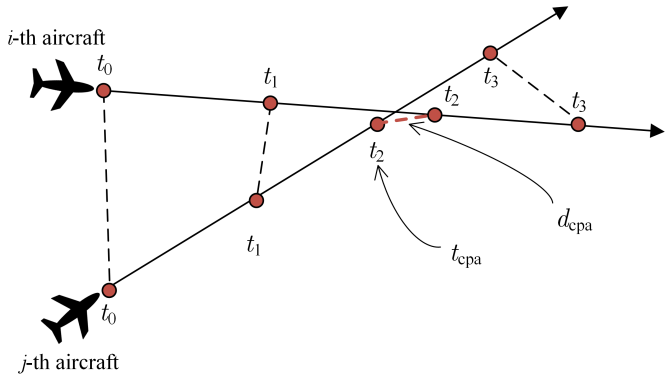

Figure 1: CPA illustration

Conventional model uses the following procedure to compute t_{CPA} and d_{CPA}. At first, it is assumed that two aircraft i, j fly at the same altitude with fixed heading and ground speed starting from common time t_0. Their speed vectors are $\mathbf{v}_i = (v_i\cos(\varphi_i), v_i\sin(\varphi_i))^T$ and $\mathbf{v}_j = (v_j\cos(\varphi_j), v_j\sin(\varphi_j))^T$. Their position vectors at initial time are $\mathbf{p}_i = (x_i, y_i)^T$ and $\mathbf{p}_j = (x_j, y_j)^T$. Thereinto, v_m, φ_m, x_m and y_m, $m \in \{i, j\}$ are speed, heading and X, Y coordinates of the m-th aircraft, respectively. Then, using basic calculation, t_{CPA} and d_{CPA} can be derived as follows:

$$t_{\mathrm{CPA}}(\mathbf{p}_i, \mathbf{p}_j, \mathbf{v}_i, \mathbf{v}_j) = \frac{-\left(\mathbf{p}_j - \mathbf{p}_i\right)\cdot\left(\mathbf{v}_j - \mathbf{v}_i\right)}{\left|\mathbf{v}_j - \mathbf{v}_i\right|^2} \tag{1}$$

$$= \frac{-X_{\mathrm{diff}}V_{\mathrm{sdiff}} - Y_{\mathrm{diff}}V_{\mathrm{cdiff}}}{{V_{\mathrm{sdiff}}}^2 + {V_{\mathrm{cdiff}}}^2} \tag{2}$$

$$d_{\mathrm{CPA}}(\mathbf{p}_i, \mathbf{p}_j, \mathbf{v}_i, \mathbf{v}_j) = \left|t_{\mathrm{CPA}}\cdot\left(\mathbf{v}_j - \mathbf{v}_i\right) + \mathbf{p}_j - \mathbf{p}_i\right| \tag{3}$$

$$= \sqrt{\left(t_{\mathrm{CPA}}V_{\mathrm{sdiff}} + X_{\mathrm{diff}}\right)^2 + \left(t_{\mathrm{CPA}}V_{\mathrm{cdiff}} + Y_{\mathrm{diff}}\right)^2} \tag{4}$$

where

$$\begin{aligned} X_{\mathrm{diff}} &= x_j - x_i \\ Y_{\mathrm{diff}} &= y_j - y_i \\ V_{\mathrm{sdiff}} &= v_j\sin(\varphi_j) - v_i\sin(\varphi_i) \\ V_{\mathrm{cdiff}} &= v_j\cos(\varphi_j) - v_i\cos(\varphi_i) \end{aligned}$$

Note that, if $\left|\mathbf{v}_j - \mathbf{v}_i\right| = 0$, two aircraft are flying parallelly at the same speed. In this case, their initial positions are considered as the CPA, and t_{CPA} is noted as 0. If the calculated t_{CPA} is inferior to 0, it means that the CPA has been passed and two aircraft are diverging.

In this study, for convenience, aircraft is considered as a rigid body. We will perform MTCD on aircraft pairs, with which the t_{CPA} is within the lookahead interval $[5, 20]$ min. Then, if d_{CPA} is equal or less than the distance threshold d_s, a potential conflict is declared. According to ICAO Doc 4444 [17], the minimum horizontal separation between two aircraft in the en-route radar control airspace is 5 Nm. This method is referred to as fixed threshold CD, which is used in Eurocontrol's MTCD tool [14]. Remark that, we consider this conventional CD approach as the CPA baseline method.

3. Raw Data Cleaning and Processing

The data source for this study is the mode-S observations covering France on January 20, 2012. This raw dataset contains 21,314 trajectories with 11,214,216 records. The following information is contained in each record:

- Flight number
- Coordinated Universal Time (UTC) timestamp
- Position (longitude, latitude, altitude)
- Ground speed
- Vertical speed
- Heading
- Wind direction and speed

We will generate the close-to-reality simulated dataset with aircraft intent information by following steps:

A. Coordinate system transformation

Conflict detection focuses on trajectory pairs, which have relatively small scales. Therefore, we can convert Geographic Coordinate System (GCS) into Projection Coordinate System

(PCS). Specifically, we choose the spatial reference EPSG 2154[1] as the PCS. The area of use for this PCS is France. The projection method of this coordinate system is Lambert Conformal Conic (LCC), which is widely used for aeronautical charts. Thus, longitude, latitude and altitude are converted into X, Y and Z coordinates.

B. Missing point estimation

Normally, the radar makes a full scan every 4 seconds. Nevertheless, sometimes the update frequency can be 8, 16, 32 seconds, which may cause missing values in the data. Since the trajectories consisted of discrete points are not smooth, we use piecewise linear interpolation to estimate the missing information.

C. Recalculation and transformation of heading

The heading in the raw dataset is magnetic heading, which is in relation to magnetic north. After coordinate transformation, the heading φ (in degrees) in the current coordinate system need to be recalculated. $\varphi_{i,t}$ of the i-th aircraft at time t can be estimated by position changes:

$$\varphi_{i,t} = \begin{cases} \hat{\varphi}_{i,t}, & \hat{\varphi}_{i,t} \geq 0 \\ 360^\circ + \hat{\varphi}_{i,t}, & \hat{\varphi}_{i,t} < 0 \end{cases} \tag{5}$$

where

$$\hat{\varphi}_{i,t} = \arctan2\left((x_{i,t+1} - x_{i,t-1}), (y_{i,t+1} - y_{i,t-1})\right) \tag{6}$$

The output of arctan2 is in degrees. To illustrate the heading calculation, some specific values of heading and the output of $\hat{\varphi}$ are plotted on a circle, see Figure 2. $t = 2, ..., M_i - 1$, where M_i is the total number of points of the i-th trajectory. x_i and y_i are respectively X, Y coordinates of the i-th aircraft.

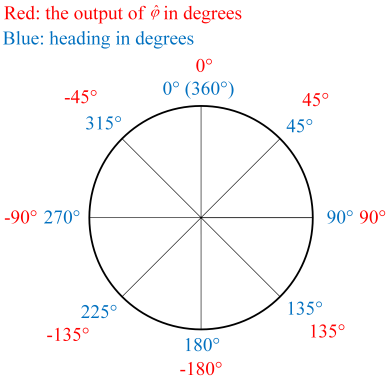

Figure 2: Illustration of heading and the output of $\hat{\varphi}$

It is noteworthy that heading is cyclical. For example, heading 0° is equivalent to heading 360°, so that heading 359° and heading 0° should be 1° apart. However, if we leave this feature unencoded, the distance between heading 359° and heading 0° will be calculated as 359° by subtraction, which is undesirable.

[1]Detailed information can be found at https://epsg.io/2154

To solve this problem, we encode the heading using the following transformation:

$$\varphi \mapsto \big(\cos(\varphi), \sin(\varphi)\big) \tag{7}$$

Because sine and cosine functions are both uniformly continuous on $\mathbb{R}$, the newly constructed features of heading become cyclical.

D. Data filtering

A research on sensitivity analysis of CPA [12] demonstrated that the heading has a much greater impact on the CPA calculation than the ground speed. In order to ensure the usage of CPA calculation while considering the uncertainty in the real-world, we leave the ground speed as it is and allow minor changes in heading. More specifically, we will check each trajectory and filter out the trajectories which don't meet the criteria below:

- The maximum change of heading should be less than 4°.
- The consecutive change of heading over 1° should occur less than 4 times.
- The total change of heading should be less than 10 times.

Figure 3 shows some violation examples of these criteria.

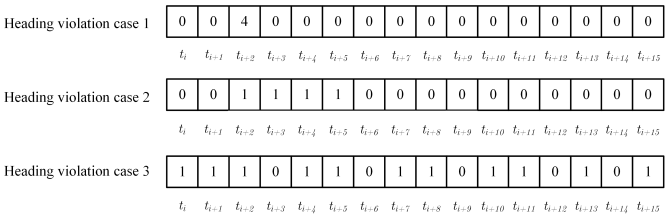

Figure 3: Some violation cases in terms of heading

E. Time alignment

In reality, there are very few accidents. To generate initial cases with more conflicts, we align the initial timestamps of all trajectories to 0. Each generated trajectory pair is designed to ensure that two aircraft are flying freely at their speed and heading without conflict resolution maneuvers, which is very close to the real aircraft intent. In addition, after time alignment, the traffics become more complicated due to large number of potential conflicts. This fact increases the difficulty of the CD problem and is a great challenge to CD approaches.

F. Trajectory pairs matching

For simplicity, we only consider spatially intersecting trajectory pairs flying horizontally at the same optimal altitude, although few disjoint trajectories may also contain dangerous conditions. In this way, the trajectories for CD is in 2D space. Because the heading of each trajectory changes slightly, we can approximate trajectories as line segments defined by the start and end points. A widely used algorithm is applied to determine whether two line segments intersect [18]. Then, we will check all trajectory pairs and keep the intersecting trajectory pairs in the dataset. Next, we will calculate the actual d_{CPA} and t_{CPA} of each intersecting trajectory pair by selecting

the minimum distance between all points of two trajectories and the corresponding index. Note that, in this paper, we only focus on aircraft pairs with t_{CPA} between 5 to 20 minutes. Hereby, trajectory pairs with 300s $\leq t_{\text{CPA}} \leq$ 1200s will be kept. Finally, the dataset contains 88,217 trajectory pairs. The remaining trajectories in the dataset were illustrated in Figure 4. It can be seen that most trajectories are over the French border. The reason is that aircraft are in the cruise phase when they are above the border, and only the aircraft flying levelly at the optimal altitude were kept in the dataset.

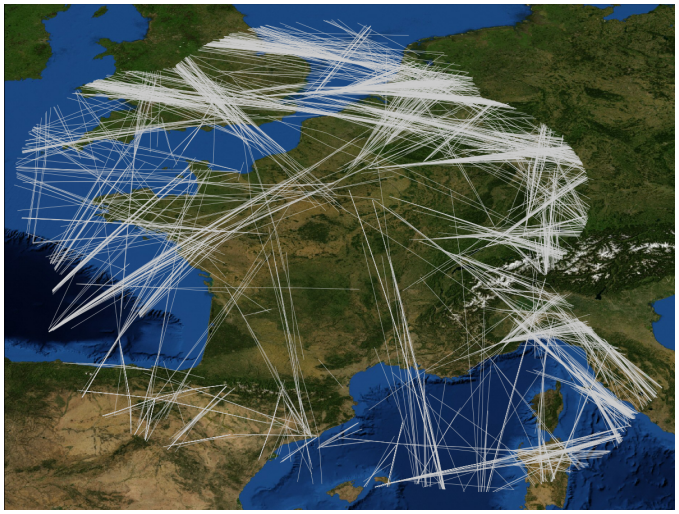

Figure 4: Illustration of 2D trajectories in the dataset

4. Machine Learning Approach to Predict Closest Point of Approach and Detect Conflict

A. Feature extraction

Feature engineering refers to the process of extracting features from raw data using domain knowledge and transforming them into formats which are suitable for ML models [19]. In this study, we adopt basic features contained in the raw dataset and the enriched additional features to build more accurate prediction models. Here, 18 features used in this study are presented in Table I.

TABLE I: The complete list of features

Type	Feature	Description
Basic	x_i, y_i	Position of i-th aircraft
	x_j, y_j	Position of j-th aircraft
	$\cos(\varphi_i)$, $\sin(\varphi_i)$	Cyclical transformation for heading of i-th aircraft
	$\cos(\varphi_j)$, $\sin(\varphi_j)$	Cyclical transformation for heading of j-th aircraft
	v_i, ψ_i, w_i	Ground speed, wind direction and wind speed of i-th aircraft
	v_j, ψ_j, w_j	Ground speed, wind direction and wind speed of j-th aircraft
Enriched	$x_j - x_i$	Components of CPA formula
	$y_j - y_i$	
	$v_j \sin(\varphi_j) - v_i \sin(\varphi_i)$	
	$v_j \cos(\varphi_j) - v_i \cos(\varphi_i)$	

B. Machine learning models

In the real world, there are minor changes in the heading when aircraft are leveling, but the ground speed varies greatly. These changes are stochastic and irregular, depending on both environmental factors and human factors. They are in violation of the assumption of the CPA formulas and bring some difficulties to the CPA baseline model prediction.

In order to enhance the baseline CD model applied in real conditions, we introduce ML techniques to predict t_{CPA} and d_{CPA} based on real trajectory dataset generated in section 3. Note that, for each ML algorithm, we build 2 single-target models, each trained on training set $\mathcal{S}_i : (\mathbf{X}, \mathbf{t}^{(i)}) = \{(\mathbf{x}_1, t_1^{(i)}), ..., (\mathbf{x}_N, t_N^{(i)})\}, i = 1, 2$, where 2 target variables $\mathbf{t}^{(1)}$ and $\mathbf{t}^{(2)}$ correspond to t_{CPA} and d_{CPA}, respectively.

The prediction of t_{CPA} and d_{CPA} belongs to regression problem. The idea is to approximate the mapping function from input vector $\mathbf{x} \in \mathbb{R}^D$ in the input space $\mathcal{X}$ to the corresponding output vector $t \in \mathbb{R}$ in the output space $\mathcal{Y}$:

$$t = h(\mathbf{x}) + \epsilon \tag{8}$$

where ϵ is random noise.

To select the appropriate h from hypothesis space $\mathcal{H}$, Empirical Risk Minimization (ERM) [20] is introduced by means of selecting h^* that minimizes the empirical risk:

$$h^* = \arg\min_{h \in \mathcal{H}} R_{\text{emp}}(h) \tag{9}$$

where R_{emp} is the empirical risk, defined as the average of the loss function on the training set:

$$R_{\text{emp}}(h) = \frac{1}{n} \sum_{i=1}^{N} L(h(x_i), t_i) \tag{10}$$

According to No Free Lunch (NFL) theorem [21], if an algorithm performs well on a certain class of problems then it necessarily pays for that with degraded performance on the set of all remaining problems. Therefore, in order to get the best possible prediction performance for specific problem, we propose several prevailing ML models and then conduct a comparative study.

The ML methods applied to this study are presented in Table II. Multiple Linear Regression (MLR) is one of the most classical ML models. It aims to estimate the mapping function under linear assumption. Support Vector Machine (SVM) is a popular supervised learning model, firstly developed for classification [22]. The learning strategy of SVM is to construct a hyperplane in a high-dimensional space. SVM can also be used for regression, namely Support Vector Regression (SVR), by introducing a margin of tolerance [23]. Feed-Forward Neural Networks (FFNNs) are the first and simplest type of Artificial Neural Network. It is said to be universal functional approximators [24]. The feed-forward term means that the architecture doesn't have closed directed cycles, which ensures that the outputs are deterministic functions of the inputs [25]. K-Nearest Neighbors (KNN) [26] is a lazy learning algorithm, in which the prediction is made locally on the delayed dataset.

In the regression analysis, KNN approximate the target by local interpolation of the nearest neighbors. Gradient Boosting Machine (GBM) is a famous ensemble learning method and can be viewed as iterative functional gradient descent algorithms [27]. Random Forests (RF) [28] are popular tree-based ensemble learning method. They are combinations of tree predictors such that each tree in the forest depends on the values of a random vector sampled independently and with the same distribution. With the combination of weak learners, a stronger learner will be generated. As an ideal candidate for bootstrap aggregating (bagging) algorithm, the idea in RF is to improve the variance reduction of bagging by reducing the correlation between the trees, without increasing the variance too much. In addition, because the law of large numbers, overfitting is seldom seen in RF with sufficient number of data.

TABLE II: ML methods used in this study

Type	Algorithm
Linear	Multiple Linear Regression (MLR)
Non-Linear	Support Vector Machine (SVM) Feed-Forward Neural Networks (FFNNs) K-Nearest Neighbors (KNN)
Ensemble	Gradient Boosting Machine (GBM) Random Forests (RF)

Then, CD can be achieved by comparing the minimum separation threshold d_s with the predicted d_{CPA}, which is a binary classification problem. It can be represented by a function $g : \mathbb{R}^D \to \{0, 1\}$ defined as:

$$g(x) = \begin{cases} 1, & d_{\mathrm{CPA}}(x) \leq d_s \\ 0, & d_{\mathrm{CPA}}(x) > d_s \end{cases} \tag{11}$$

where $g(x) = 1$ indicates that an alert have to be issued and a conflict occurs, vice versa.

5. Model Validation and Results Discussion

A. Experimental setup and model selection

The experiment was run on a laptop with Intel core i7-8750H CPU @ 2.20GHz, 16GB RAM and NVIDIA GeForce GTX 1070 GPU. All algorithms were implemented in Python 3.6.2.

The ERM mentioned in Section 4 is used for selecting the best model from a training set. However, the model performance on new dataset may be disappointing. It is known as overfitting, which should be avoided. Another possible case is that the model is not enough to capture the underlying relations between inputs and outputs. This fact is known as underfitting, which is also undesirable. These conflicts are referred to as bias-variance trade-off. To handle this problem, we propose nested Cross Validation (CV). It consists of outer loops and inner loops. A K_1-fold CV splits the dataset S into K_1 subsets S_i, $i = 1, ..., K_1$. For each outer loop i, S_i is the test set and the remaining $K_1 - 1$ folds $S_{-i} = S \backslash S_i$ act as the training set. Then, there is another K_2-fold CV, which will further split the training sets S_{-i} into K_2 subsets $S_{-i,j}$, $j = 1, ..., K_2$. For each inner loop j, $S_{-i,j}$ act as the validation set and the remaining $K_2 - 1$ folds $S_{-i} \backslash S_{-i,j}$ act as the training set. The purpose of the inner loop is to select the hyperparameters and the outer loop aims to assess the model performance. Let $K_1 = 5$, $K_2 = 5$, then the proportion of training sets, validation sets and test sets is respectively $64\%/16\%/20\%$.

We use random search algorithm for hyperparameter optimization in the inner CV, considering that it is much more efficient than classical grid search algorithm in high-dimensional search space [29]. The strategy of random search is based on independent draws from a probability distribution in the grid or range of hyperparameters. The pseudocode is presented in Algorithm 1. The random search will be conducted with 64 trials in terms of each model. The hyperparameters need to be tuned according to their probability distributions are presented in Table III. Other hyperparameters not in the table are set to default values.

Algorithm 1 Random search on model $\mathcal{A}$

Input:
$\mathbf{V} = \{\mathbf{v}_1, ..., \mathbf{v}_N\}$: Range or grid of N hyperparameters
$\mathbf{F} = \{F_1(\cdot), ..., F_N(\cdot)\}$: Cumulative Distribution Function (CDF) of N hyperparameter values
S_V: Validation set
K: Numbers of sampling iterations
Output:
λ^*: Selected hyperparameters

procedure RandomSearchCV($\mathbf{V}$, $\mathbf{F}$, S_V, K)
  for i = 1 to K **do**
    $(r_1, .., r_N) \leftarrow N$ uniformly distributed random numbers generated between 0 and 1
    $\Lambda_i \leftarrow (F_1^{-1}(r_1), ..., F_N^{-1}(r_N))$ ▷ Inverse CDF
  end for
  $\lambda^* \leftarrow \arg\min_{\lambda \in \Lambda} Err(\mathcal{A}, \lambda, S_V)$
  return λ^*
end procedure

SVM, FFNNs and GBM were trained for 10000 epochs. For FFNNs and GBM, reducing the learning rate as the training progresses is useful to improve the learning ability. To this end, we use the schedule of reducing the learning rate when a metric stopped improving, commonly known as ReduceLRonPlateau. In this paper, the metric is chosen as the Mean Absolute Error (MAE) on the validation set. The learning rate is starting from 0.1 with patience of 100 epochs and decay factor of 0.5. The learning rate can be adaptively adjusted by this algorithm and need not to be manually tuned.

B. Performance evaluation

We use Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to assess the prediction performance

TABLE III: Hyperparameters optimized in random search

Method	Hyperparameter	Range or grid	Distribution
MLR	λ	$[10^{-5}, 10]$	log-uniform
SVM	kernel	{'linear', 'poly', 'rbf', 'sigmoid'}	uniform
	degree [a]	{2,3,4,5}	uniform
	γ [b]	$[10^{-5}, 1]$	log-uniform
	C	$[1, 10^3]$	log-uniform
	ε	$[10^{-2}, 1]$	log-uniform
FFNNs	M	$\{2, 3, \ldots, 10\} \cup \{16, 32, 64, 128\}$	uniform
	loss function	{'MAE', 'MSE'}	uniform
	Φ	{'ReLU', 'tanh', 'sigmoid'}	uniform
KNN	K	{1,2,...,20}	uniform
	weight function	{'uniform', 'inv_distance'}	uniform
	p [c]	{1,2,3,4,5,6}	uniform
GBM	boosting type	{'GBDT', 'GOSS', 'DART'}	uniform
	max number of leaves	{10, 20, ..., 100}	uniform
	fraction of bagging	{0.5, 0.7, 0.8, 0.9}	uniform
	fraction of feature	{0.5, 0.7, 0.8, 0.9}	uniform
	loss function	{'MAE', 'MSE'}	uniform
RF	number of estimators	{10, 20, ..., 100}	uniform
	max_features	[1, 2, 4, 8, 16, 32, 64, 128]	uniform
	loss function	{'MAE', 'MSE'}	uniform

[a] Only used when kernel is 'poly'.
[b] Only used when kernel is 'rbf', 'poly' or 'sigmoid'.
[c] Only used when weight function is 'inv_distance'.

on d_{CPA} and t_{CPA}:

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|t_i - y_i| \tag{12}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(t_i - y_i)^2} \tag{13}$$

where y_i is the forecast value and t_i is the actual value.

To evaluate the classification accuracy of CD, we introduce the idea of confusion matrix [30]. The classification result contains four possible cases for the predicted class and the actual class: True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN). TP and TN are correct decisions made. FP and FN are also known as type I error and type II error, respectively. The illustration is shown in Figure 5. Furthermore, we will introduce four measures including True Positive Rate (TPR), True Negative Rate (TNR), False Negative Rate (FNR) and False Positive Rate (FPR), which are respectively defined as follows:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{14}$$

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}} \tag{15}$$

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{FN}+\mathrm{TP}} \tag{16}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}} \tag{17}$$

C. Prediction results of d_{CPA} and t_{CPA}

Table IV depicts the prediction performance of d_{CPA} and t_{CPA} for different algorithms. The best value of each metric is marked in bold. Though rigorous assumptions have to be made a priori while this is not always true in reality, the prediction performance of CPA baseline model on d_{CPA} is not too bad, considering the mean value of actual d_{CPA} is nearly 30 Nm. Nevertheless, the prediction results on t_{CPA} is not satisfying, especially in terms of RMSE. Note that, the mean value of actual t_{CPA} is only about 550s. This fact indicates that the CPA baseline model is unreliable to predict t_{CPA} in the real world and the results are very unstable. Compared to the CPA baseline model, there are 3 ML models that perform better on d_{CPA} and 4 ML models that perform better on t_{CPA}. The intersection of these well-performed models includes FFNNs, GBM and RF. GBM significantly outperforms other models both in terms of d_{CPA} and t_{CPA} prediction. The MAE of the CPA baseline model is reduced by 2.34 Nm (62.23%) and the RMSE is reduced by 5.18 Nm (64.67%) for d_{CPA} prediction. In view of t_{CPA} prediction, the performance improvement is much more obvious: the MAE of the CPA baseline model is reduced by 56.94 seconds (66.95%) and the RMSE is reduced by 720.65 seconds (91.69%). MLR is completely incapable of this problem in that the relationship between input and output is far from linear. KNN and SVM are also inferior to other approaches, which could be attributed to their insufficient learning capabilities.

With less prediction error on d_{CPA}, ATCO and pilots will obtain more precise information about the spatial severity of potential conflicts. Accurate t_{CPA} will bring more reliable temporal severity information about potential conflicts. Reliable spatial and temporal severities result in the increase of situation awareness. Hereby, effective and exact conflict resolution instructions will be delivered by ATCO, which can avoid accidents. These are the benefits of ML models to enhance the CD accuracy.

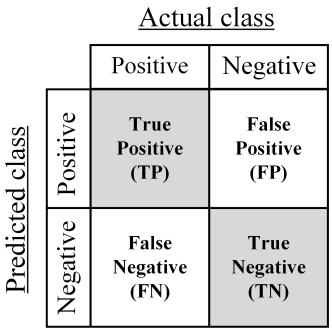

Figure 5: Confusion matrix visualization

TABLE IV: Prediction results of d_{CPA} and t_{CPA}

Models	d_{CPA}(Nm)		t_{CPA}(s)	
	MAE	RMSE	MAE	RMSE
Baseline	3.76	8.01	85.05	766.31
MLR	22.23	31.36	227.71	280.63
SVM	6.28	10.29	98.64	158.12
FFNNs	1.96	3.33	41.40	86.53
KNN	5.79	10.43	76.11	127.11
GBM	**1.42**	**2.83**	**28.11**	**63.66**
RF	2.14	4.20	34.26	72.28

D. Conflict detection accuracy

TABLE V: Confusion matrix

Models	TP		FN		TN		FP	
	Num	Rate	Num	Rate	Num	Rate	Num	Rate
Baseline	6359	92.15%	542	7.85%	10293	95.81%	450	4.19%
SVM	5323	84.78%	956	15.22%	9787	86.12%	1578	13.88%
FFNNs	6551	94.93%	350	5.07%	10377	96.59%	366	3.41%
KNN	5729	83.02%	1172	16.98%	9852	91.71%	891	8.29%
GBM	**6673**	**96.70%**	**228**	**3.30%**	**10517**	**97.90%**	**226**	**2.10%**
RF	6514	94.39%	387	5.61%	10418	96.97%	325	3.02%

To evaluate the effectiveness of our models for CD, we conduct CD by comparing the predicted d_{CPA} with the minimum horizontal separation $d_s = 5$ Nm. This assignment of d_s has a better understanding for ATC through raising the safety margin. The classification results are summarized in Table V. The best value of each metric is highlighted. As for the CPA baseline model, The TNP and FPR are acceptable. 95.81% TNR indicates that most negative classes can be correctly classified. However, TPR and FNR are not satisfying: 7.85% actual conflict cases are not correctly identified. It should be noted that FN is arguably more serious than FP, because FN incorrectly identifies conflict case as conflict-free, which may let ATCO underestimate the danger and ignore it rather than deliver conflict resolution instructions. In view of ML models, we do not mention MLR due to its low performance. It is observed that 3 ML models outperform the CPA baseline model. They are still FFNNs, GBM and RF. GBM is the best one among these models with 4.55% improvement in conflict case identification and 2.09% improvement in non-conflict case identification. Although results of GBM contain 3.30% FN that miss actual conflicts, note that, all conflicts occur in the dataset are assumed that there are no conflict resolution maneuvers made by aircraft. Besides, through analysis, we find that most FN cases have long t_{CPA}. The longer the t_{CPA}, the higher the uncertainty. Therefore, Such results are capable of proving the CD capability. Furthermore, GBM not only can improve the performance of the CPA baseline model but also can be used for collision detection in reality.

6. Conclusion and Perspectives

This paper presents a novel ML approach to enhance the CD performance in CPA problem by learning from real trajectory data. Unlike a vast majority of models which assume that aircraft fly horizontally with uniform speed and heading, the proposed model is capable of dealing with level flights in reality with minor change in heading and great variation in ground speed.

We firstly introduce a data processing method to generate the dataset from mode-S observations. This dataset represents all aircraft intents in reality. Then, feature engineering is conducted to transform raw data into suitable features for prediction models. To validate the effectiveness of the ML approach for CPA prediction, six ML models are used to predict t_{CPA} and d_{CPA} with comparing to baseline CPA model. The numerical experiment demonstrates that three ML models outperform the CPA baseline model in terms of accuracy and stability. In particular, GBM is the best model and greatly improve the CPA baseline model. Furthermore, we conduct CD by comparing the predicted d_{CPA} with the minimum horizontal separation. The classification result is presented using the idea of confusion matrix, which proves that GBM still achieve the highest accuracy in CD. However, the result also demonstrates that not all ML models outperform the CPA baseline model. Suitable ML methods are capable of enhancing the accuracy of MTCD.

In the future work, we will go deeper into feature engineering by extracting more meaningful features to improve the prediction performance. Additional factors (such as ATC and AOC data) will be involved, as they are influencing the trajectories. Furthermore, A confirmation that the results still stand with more complex and realistic trajectories will be performed. In addition, the probability of conflict is also important to the ATCO. Thus, we will calculate the conflict probability according to the d_{CPA}, t_{CPA} and uncertainties of error.

References

[1] Radio Technical Commission for Aeronautics, *Final Report of RTCA Task Force 3 Free Flight Implementation*. RTCA, 1995.

[2] M. Prandini, J. Hu, J. Lygeros, and S. Sastry, "A probabilistic approach to aircraft conflict detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 199–220, 2000.

[3] D. G. Denery and H. Erzberger, *The Center-Tracon Automation System: Simulation and Field Testing*, 1997.

[4] D. J. Brudnicki, K. S. Lindsay, and A. L. Mcfarland, "Assessment of field trials, algorithmic performance, and benefits of the user request evaluation tool (uret) conflict probe," in *IEEE Digital Avionics Systems Conference*, 1997.

[5] P. Brooker, "Stca, tcas, airproxes and collision risk," *The Journal of Navigation*, vol. 58, no. 3, pp. 389–404, 2005.

[6] M. D. Ciletti and A. W. Merz, "Collision avoidance maneuvers for ships," *Navigation*, vol. 23, no. 2, pp. 128–135, 1976.

[7] E. M. Goodwin, "A statistical study of ship domains," *Journal of Navigation*, vol. 28, no. 3, pp. 328–344, 1975.

[8] P. Bauer, A. Hiba, J. Bokor, and A. Zarandy, "Three dimensional intruder closest point of approach estimation based-on monocular image parameters in aircraft sense and avoid," *Journal of Intelligent & Robotic Systems*, pp. 1–16, 2018.

[9] J. Kucher and L. Yang, "Survey of conflict detection and resolution modeling methods," in *AIAA Guidance, Navigation, and Control Conference*, 1997, pp. 1388–1397.

[10] C. Munoz, A. Narkawicz, and J. Chamberlain, "A tcas-ii resolution advisory detection algorithm," in *Aiaa Guidance, Navigation, & Control*, 2013.

[11] L. C. Yang and J. K. Kuchar, "Prototype conflict alerting system for free flight," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 4, pp. 768–773, 1997.

[12] Y. Huo, D. Delahaye, and Y. Wang, "Sensitivity analysis of closest point of approach," in *ICRAT 2018, 8th International Conference for Research in Air Transportation*, 2018.

[13] M. Prandini, J. Hu, J. Lygeros, and S. Sastry, "A probabilistic approach to aircraft conflict detection," *IEEE Transactions on intelligent transportation systems*, vol. 1, no. 4, pp. 199–220, 2000.

[14] S. Alam, H. Abbass, C. Lokan, M. Ellejmi, and S. Kirby, "Computational red teaming to investigate failure patterns in

medium term conflict detection," in *8th Eurocontrol Innovation Research Workshop, Eurocontrol Experimental Center, Brtigny-sur-Orge, France*, 2009.
[15] Q. Yang, A. Lim, K. Casey, and R.-K. Neelisetti, "An enhanced cpa algorithm for real-time target tracking in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 5, no. 5, pp. 619–643, 2009.
[16] S. Arumugam and C. Jermaine, "Closest-point-of-approach join for moving object histories," in *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*. Citeseer, 2006, pp. 86–86.
[17] ICAO, *Doc 4444, Procedures for Air Navigation Services - Air Traffic Management (PANS-ATM)*, Sixteenth edition, 2016.
[18] E. D. Jou, "Determine whether two line segments intersect," in *Modeling in Computer Graphics*. Springer, 1991, pp. 265–274.
[19] A. Zheng and A. Casari, *Feature engineering for machine learning*. O'Reilly Media, 2018.
[20] V. Vapnik, "Principles of risk minimization for learning theory," in *Advances in neural information processing systems*, 1992, pp. 831–838.
[21] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.
[22] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152.
[23] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in neural information processing systems*, 1997, pp. 155–161.
[24] B. C. Csáji, "Approximation with artificial neural networks," *Faculty of Sciences, Etvs Lornd University, Hungary*, vol. 24, p. 48, 2001.
[25] C. M. Bishop, "Pattern recognition and machine learning (information science and statistics)," no. 4, p. 049901, 2006.
[26] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.
[27] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.
[28] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
[29] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.
[30] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

AUTHORS BIOGRAPHY

Zhengyi Wang received the B.E. degree in Aviation Engineering from Civil Aviation University of China (CAUC) in 2016, and the M.Sc. degree in Operational Research from École Nationale de l'Aviation Civile (ENAC) in 2018. He is currently studying for secondary M.Sc. and engineering degree in ATM and Airport Operations at CAUC. His research interests focus on solving ATM problems by combining real data with emerging data mining and ML techniques.

Man (Annie) Liang is currently a lecturer in School of Engineering, University of South Australia (UniSA). She received the B.E. degree in Transport and Transportation (Air Traffic Control) from CAUC, and then the Advanced Master degree in Air Transport Management and Air Navigation Operations from ENAC in 2006. She obtained her Ph.D in Applied Mathematics from University of Toulouse&ISAE-SUPAERO in Feb. 2018 and did a post-doc at ENAC. During 2006-2014, she worked at the department of Air Traffic Management at CAUC as a lecturer and researcher. She holds the Chinese air traffic controller license.

Daniel Delahaye is working as a Professor and Director in the OPTIM Laboratory of ENAC. He obtained his engineering degree from the ENAC school and a master of science in signal processing from the National Polytechnic Institute of Toulouse in 1991. He obtained his Ph.D in automatic control from the Aeronautic and Space National school in 1995 and did a post-doc at the Department of Aeronautics and Astronautics at MIT in 1996.

Weilu Wu received the B.E. degree in Aviation Engineering from CAUC in 2016, and the Advanced Master certificate in Air Navigation System Engineering and Operations from ENAC in 2018. She is currently studying for secondary M.Sc. and engineering degree in ATM and Airport Operations at CAUC.

APPENDIX

A. Speed change illustration

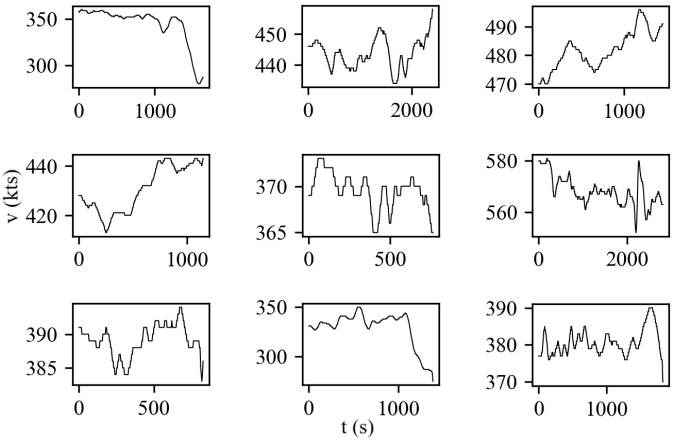

Figure I: Some examples of ground speed changes over time