Deep Reinforcement Learning for Airplane Components Failure Prognostics Full Cycle Automation

Baoqian Wang[1], Changzhou Wang[2], and Denis Osipychev[3]

[1]*Boeing AI,* [3]*Software Engineering, The Boeing Company, Huntsville, Alabama, 35808, USA*
baoqian.wang@boeing.com
denis.osipychev@boeing.com

[2]*Boeing Global Services, The Boeing Company, Seattle, Washington, 98124, USA*
changzhou.wang@boeing.com

ABSTRACT

As airplane components degrade over time, airplane service organization and airline maintenance team need to collaborate on performing component failure prognostics to improve operation efficiency. In particular, the airplane service organization analyzes flight/sensor data and sends alerts to the airline maintenance team upon identifying a potential failure. In response, the airline maintenance team conducts inspections and replaces the component if necessary. In this procedure, it is crucial to determine the timing for sending alerts: late alerts leave operators no time to avoid schedule interruptions with huge operation costs, while early alerts lead to unnecessary inspections and significant maintenance burden. Current solutions rely on heuristics and/or manual engineering reviews to make decisions on whether and when to send alerts, which is difficult to scale and adapt. To address this limitation**,** we developed a deep reinforcement learning-based approach to automate the prognostics procedure and enhance accuracy of alert timing. A case study on Air Cycle Machine (ACM) prognostics was conducted to demonstrate the feasibility and effectiveness of our approach.

1. INTRODUCTION

As airplane components degrade over time, airplane service organizations (e.g., Boeing Global Services (BGS)) and their airline customers need to collaborate on airplane components failure prognostics to replace/maintain components proactively to improve operation efficiency and reduce maintenance cost. In particular, airplane components are often mounted with sensors to monitor their operational states, the airline maintenance team sends flight records that include sensor information and service records that include maintenance history to the airplane service organization. By analyzing these flight and service records, the airplane service organization predicts possible component failures. Upon identifying an impending component failure, the airplane service organization promptly sends alerts to the airline maintenance team. In response, the airline maintenance team conducts inspections and maintenance on the component and replaces it if necessary. This airplane component failure prognostics procedure is illustrated in Figure 1.

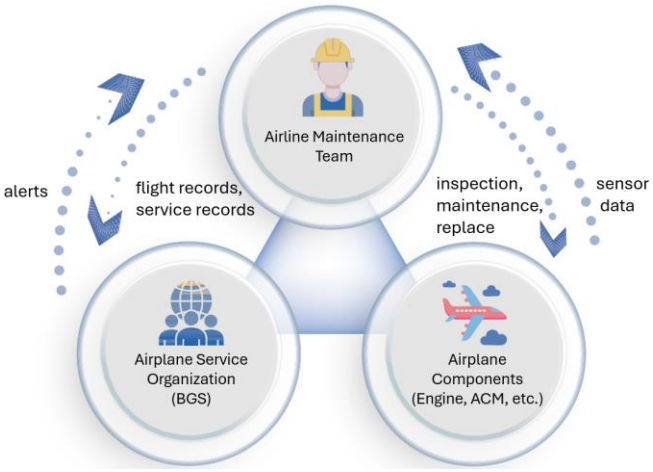

Figure 1. Airplane components failure prognostics.

In this airplane components failure prognostics procedure, machine learning or engineering-based models can be used to make a prediction of airplane component failure. However, it is crucial for the airplane service organization to determine when to send alerts to airlines given the failure predictions. Late alerts may cause schedule interruptions or even grounding of the airplane waiting for parts. Early alerts can

bring unnecessary inspections that lead to significant maintenance burden.

In the research literature, most airplane component failure prognostics methods predict the status of the airplane components based on the sensor or flight data using data driven model, e.g., machine learning or probabilistic models. Alerts will be sent upon a potential component failure prediction. These methods ignore the interactions between the airplane service organization and their airline customers such as historical alert timings, inspection results which include important information about airplane component operation states as well as the prognostics procedure status. The industry solutions rely on heuristic rules and engineering reviews, which require significant manual efforts and are difficult to scale or adapt.

To improve the efficiency of airplane components failure prognostics, we developed a deep reinforcement learning-based approach that automates the full cycle of prognostics procedure and enhances accuracy of alert timing. Specifically, we first built probabilistic models from real flight and service records to simulate airline responses to alerts and component state transitions, thereby enabling a full-cycle prognostics simulation. We then used a Long Short-Term Memory (LSTM) neural network model to represent the alert policy that outputs alerts decisions based on flight records and service records. By running the alert policy in simulated prognostics procedure, the policy is trained by policy learning algorithms based on feedback received from the simulation. We investigated two policy learning algorithms including Deep Q Network (DQN) algorithm and its variant Double Deep Q Network (DDQN) with Prioritized Replay Buffer (PRB) to learn the alert policy. Once trained, the policy can be deployed to automatically make alert decisions by processing incoming flight records and service records. Additionally, the alert policy parameters can be fine-tuned to adapt to new airplane component features and evolving airline operations. We conducted a case study on ACM prognostics using data from 17 airlines by comparing our approach to the currently deployed heuristic approach, which demonstrated the feasibility and effectiveness of our approach.

Our main contributions are: 1) Development of a deep reinforcement learning-based approach to automate full cycle prognostic procedure and enhance alert timing accuracy. Our approach leverages probabilistic data modeling derived from real flight and service records to simulate the prognostics process. The approach adopts an LSTM neural network to represent the alert policy, which effectively captures the temporal features in time-series records. 2) A case study on ACM prognostics that validates our approach by comparing our approach to the currently deployed heuristic prognostics approach. Using real data from 17 airlines, the study provides valuable insights into the practical deployment of the alert policy and guides parameter tuning to align with operational requirements.

In the rest of the paper, Section 2 reviews related work on airplane component prognostic and health management. Section 3 formulates the airplane components failure prognostic problem considered in this paper. A reinforcement learning formulation for prognostics procedure is provided in Section 4. Section 5 and 6 discuss alert policy learning algorithms, and alert policy deployment and evaluation, respectively. Section 7 presents the case study on ACM to evaluate our approach. Section 8 concludes the paper. All mathematical symbols used in this paper are defined in the Nomenclature.

2. RELATED WORK

The airplane component prognostic procedure typically consists of two stages. The first stage is to build a model to capture the state of airplane components and the second is to make failure predictions based on the model. The model used in the first stage can be categorized into three types including knowledge-based model, physical model, and data-driven model. The knowledge-based model uses the domain knowledge of the airplane component and empirical experience to get knowledge of airplane component operation state. This model requires expert knowledge or engineering experience and thus is inefficient and hard to scale.

In the physical model-based approach, a physical model is built through system identification and parameter estimation that includes dynamics of airplane components to capture the state transition of airplane components. For example, Kulkarni, Schumann, and Roychoudhury (2018) built an electrochemical battery model combined with Unscented Kalman Filter for onboard battery monitoring and prognostics in electric-propulsion aircraft. The physical model can provide physics interpretation of the airplane component state transitions. However, it is hard to develop high-fidelity physical models for complex airplane components such as engines.

Data-driven models are more favorable that allows modeling system transitions using data only without physics knowledge. Typical data-driven models include probabilistic models, machine learning models, reinforcement learning models. For example, Pidaparthi, Jacobs, Ghosh, Ravi, Amer, Luan, and Wang (2024) built a component level-probabilistic model to forecast damage growth for aircraft engines. Dangut, Skaf, and Jennions (2021) developed a hybrid machine learning approach combining natural language processing and ensemble learning for predicting aircraft component failures. Hu, Miao, Zhang, Liu, and Pan (2021) designed a reinforcement learning driven maintenance strategy for airplane maintenance decision optimization considering future mission requirements, repair cost, etc. In most existing data-driven models, the model is used to make

predictions on the failure of airplane components or maintenance decisions relying on airplane component data. But they ignore other context information of prognostics procedure such as prognostics service records that include historical alerts, inspection time and results. In practice, engineers and data analysts often rely on complete context information including component sensor data, flight records and service records to make decisions on sending alerts on potential airplane component failure. Unlike existing data-driven models, our deep reinforcement learning-based approach integrates the full-cycle context information for airplane component prognostics which leads to more accurate alert timings.

3. AIRPLANE COMPONENT FAILURE PROGNOSTICS PROBLEM FORMULATION

In this section, we describe the full cycle of airplane component failure prognostics and formulate the problem considered in this paper.

In airplane component failure prognostics procedure, there are sensors mounted on airplane components such as accelerometers, thermocouples and so on, to monitor the operational status of airplane components. In addition to the sensor data, there are flight logs associated with airplane components such as registry id, number of days since the last removal, etc. The sensor data and flight logs are concatenated into flight records denoted by $F(t)$, which are sent from airlines to airplane service organization. Moreover, airlines also share service records with the airplane service organization. The service records denoted by $S(t)$ include maintenance history in the life cycle of an airplane component such as alert times, inspection times and corresponding inspection results.

By analyzing flight records $F(t)$ and service records $S(t)$ at certain time step t, the airplane service organization determines whether to send an alert to its airline customers to remind them of impeding component failure. More formally, we aim to create a model that takes flight records $F(t)$ and service records $S(t)$ as inputs and outputs action $a(t) \in \{0, 1\}$, where 1 means sending an alert, represented as

$$a(t) = f(S(t), F(t)) \quad (1)$$

Upon receiving an alert, airlines may either ignore it or schedule an inspection. The probability of scheduling an inspection, given the current alert, flight and service records is modeled as $P(X|S(t), F(t), a(t) = 1)$, where X is a Bernoulli random variable to denote if an inspection is scheduled (X=1) or not (X=0). Due to the operational constraints, there is typically a delay between alert and inspection, represented by a continuous random variable Y. The inspection result can be Failed or Passed. A Failed result indicates that the airplane component requires maintenance or replacement, while Passed means it is functioning properly. We model the inspection result as a Bernoulli random variable Z with failure probability $P(Z|S(t), F(t))$.

Depending on airlines' responses upon receiving alerts and their following actions, there is a reward associated with the prognostic's procedure. Since the flight records and service records capture the whole contextual information, we designed a reward function captured by $r(S(t), F(t), a(t))$. Our goal is to obtain an alert model that maximizes the accumulated reward for the life cycle of an airplane component, i.e.,

$$max_f \ \sum_{t=0}^{T} r(S(t), F(t), a(t)) \quad (2)$$

where T is the terminal time of the life cycle of the airplane component.

4. REINFORCEMENT LEARNING FORMULATION

In this paper, we present a deep reinforcement learning-based approach to solving the formulated airplane component failure prognostics problem. We first provide a reinforcement learning formulation over the original problem, using a Markov Decision Process (MDP). In particular, MDP is mainly characterized by five components including state, action, environment transition, reward, policy, which are described as follows.

4.1. State and Action

State $s(t)$ is the contextual information of the failure prognostics procedure. In our case, we define the state to be a time sequence of flight records and service records, denoted by $s(t) = [S(t), F(t), \dots, S(t - N + 1), F(t - N + 1)]$, where N is number of time steps of time sequence records and is determined based on empirical experience. As discussed in problem formulation, *Action* $a(t) \in \{0, 1\}$ is whether to send an alert to airline customers at time step t. The airplane service organization (or *Agent*) uses an alert policy (or *Policy*) represented by $u(s(t))$ to output the action $a(t) = u(s(t))$.

4.2. Policy Model

The alert policy plays a critical role in effectively utilizing prognostic contextual information to generate alert decisions. The contextual information includes service records and flight records, which may contain out-of-sequence data. To address this complexity, we utilize an LSTM neural network model (Graves, Fernández, Gomez, & Schmidhuber, 2006) with fully connected layers to represent the policy. The LSTM is particularly good at capturing patterns within time-sequence data. It takes as input a state represented by a sequence of time-consecutive service and flight records. The outputs from the LSTM model are subsequently processed by fully connected layers, which allow for additional processing

and analysis of the information extracted by the LSTM model. By combining the capabilities of the LSTM model with the fully connected layers, we can generate accurate and informed alert decisions based on the contextual information provided. Moreover, LSTM allows variable sequence lengths for input simply by applying zero-padding techniques to make input length consistent. This capability enables the model to effectively handle out-of-sequence records by reordering and zero-padding the input as needed.

4.3. Environment Transition Model

Environment transition model simulates the prognostics procedure by defining the next state based on current state and action. By running the agent in the simulated prognostics procedure, we can collect a sequence of states, actions and calculate a sequence of rewards. These experiences enable the agent to optimize its decision-making behavior. Based on our airplane component failure prognostics problem formulation, we built environment transition model of failure prognostics by modeling airline's response to alerts, inspection delay, inspection result and transitions of airplane component states.

4.3.1. Airline Response Model

We estimate the probability of airlines performing inspection when they receive alerts using service records that contain interaction history between airlines and the airplane service organization. The service records include historical alert times, inspection times, and inspection results from each airline regarding an airplane component. Since each airline has its own operation policies and processes, their responses can vary. We estimate the probability of scheduling an inspection upon receiving an alert for each airline using frequentist approach. Particularly, we estimate the probability by calculating the proportion of alerts that have inspections scheduled among all alerts, which is represented by the following equation,

$$P(X|S(t),F(t),a(t)=1) \approx \frac{Number\ of\ alerts\ with\ inspections}{Number\ of\ alerts} \quad (3)$$

4.3.2. Inspection Delay Model

Once airlines decide to schedule an inspection, they will choose a date for inspection. Due to operation schedule, airlines won't inspect immediately when they decide to inspect component upon getting alerts. There is usually a delay from alert time to inspection time. We use an exponential distribution to fit the cumulative distribution function (CDF) of inspection delay *Y*. Particularly,

$$P(Y < y) = 1 - \exp(-\lambda y) \quad (4)$$

where λ is the parameter of exponential distribution and is estimated through Maximum Likelihood Estimation by using the sample mean of inspection delays,

$$\frac{1}{\lambda} \approx \frac{\sum delay\ of\ each\ inspection}{Number\ of\ inspections} \quad (5)$$

where delay of each inspection and number of inspections can be obtained from service records.

4.3.3. Inspection Result Model

In airplane component failure prognostics, we model the inspection result to depend on two factors including the number of days from the inspection time to the component's next removal time and inspection delay, which are available in service records. Note that this does not imply a causal relation from the two factors to the inspection result.

In particular, the closer the inspection time to the next removal time, the larger the probability for the inspection result being Failed, because airplane component is likely to malfunction as it gets closer to its removal time. We denote the number of days from inspection to the next removal to be a continuous random variable W. Moreover, the inspection result is also affected by inspection delay. The reason is that airplane components sometimes seem to be self-healed if only symptoms are considered. In particular, the symptom of degradation is obvious if inspected at the alert time but gradually disappears with time, e.g., when the mechanical parts grind off the touching portion of blades or cases. Therefore, when an airplane component is not functioning properly, the sooner the inspection is, the larger probability the inspection can detect that.

Based on Bayes' law, the inspection Failed probability $P(Z|W,Y)$ can be represented by

$$P(Z|W,Y) = P(Z|W) \cdot \frac{p(Y|Z,W)}{p(Y|W)} \quad (6)$$

where $p(\cdot)$ is probability density function. We then approximate $P(Z|W=w)$ with an exponential function $P(Z|W=w) = 1 - exp(-\lambda_2 w)$. We estimate parameter λ_2 using the data sample with $w = \mathrm{w}_0$. In particular, the corresponding $P(Z|W = \mathrm{w}_0)$ is approximated using

$$P(Z|W = w_0) \approx (Number\ of\ Failed\ inspections\ within\ \mathrm{w}_0\ days\ to\ removal)/(Number\ of\ inspections\ within\ \mathrm{w}_0\ days\ to\ removal) \quad (7)$$

We can then get $\lambda_2 = -\ln(1 - P(Z|W = \mathrm{w}_0))/\mathrm{w}_0$. Moreover, we assume the inspection delay is independent from the airplane component state, because when receiving

alerts, airlines mostly look at their operation schedules to schedule inspection as soon as possible. With this assumption, we approximate $p(Y|Z,W)/p(Y|W)$ with $p(Y|Z)/p(Y)$, which is further estimated by

$$p(Y=y|Z)/p(Y=y) = P(Z|Y=y)/P(Z) \approx (Number\ of\ Failed\ inspections\ with\ inspection\ delay \geq y)/(Number\ of\ Failed\ inspections) \quad (8)$$

using data samples from service records.

4.3.4. Airplane Component State Transition Model

The airplane component state transition model is used to capture the airplane component state change in response to time and actions from maintenance team. Given the complexity of these state changes, directly modeling their dynamics can be challenging. In this paper, we directly use actual sequential flight records to represent the transitions of airplane component states. Particularly, at each time step, the airplane component state transits to the next state specified by the flight record of the next time step.

When running the alert policy in simulated prognostics procedure, two special cases need to be handled regarding airplane component state transitions. The first case occurs when the simulated inspection result is Failed. In this case, we assume the airplane component is replaced, thereby terminating its current lifecycle, even if there are still actual flight records available after the simulated inspection time. The second case arises when the inspection result is Passed, but the real flight records run out for that airplane component lifecycle. In this case, the current airplane component lifecycle is also terminated.

4.4. Reward Model

The reward model provides feedback to alert policy based on alert decisions, airline responses as well as airplane component state. In our framework, the rewards associated with different operations are summarized as follows:

1) **Sending Alert:** Each time the service organization sends an alert, there is a small service cost, which is captured by a negative reward.

2) **Sending Unnecessary Alert:** As the alert policy generates alerts automatically based on context information, it can send an alert to airline even after the airline has responded to a previous alert on the same component. We give such an unnecessary alert a negative reward.

3) **Unnecessary Inspection:** Airlines schedule and perform inspections because of alerts. If an inspection is Passed indicating the component is working well, the inspection brings unnecessary costs to airlines. Therefore, we give a large negative reward to this unnecessary inspection.

4) **Successful Alert:** A large positive reward is given to a successful alert meaning either the inspection due to the alert successfully detected a malfunctioning component, or the alert is sent before component failure within a specified timeframe, e.g. 30 days, even if there is no inspection.

5) **Missing Component Failure:** We penalize missing component failure without alerts by giving it a large negative reward. The missing component failure refers to not sending alerts before component failure within a designated timeframe, e.g. 30 days.

With the above MDP definition, our objective is to get an alert policy that maximizes the expected sum of discounted reward values when running the alert policy in the simulated airplane component prognostics procedure.

5. POLICY LEARNING

With the formulated MDP for airplane components failure prognostic, the alert policy is learned by running the policy in the simulated environment to get feedback, based on which the policy optimizes its alert decision-making behaviors.

The alert policy is parameterized by weights of neural network including the LSTM module and the fully connected layers with parameters to be learned by reinforcement learning algorithms. As the input of the alert policy is continuous and its output is discrete (binary decision variable), DQN (Mnih, Kavukcuoglu, Silver, Rusu, Veness, Bellemare, Graves, Riedmiller, Fidjeland, Ostrovski, Petersen, Beattie, Sadik, Antonoglou, King, Kumaran, Wierstra, Legg, & Hassabis, 2015) or its variants including DDQN (Van Hasselt, Guez, & Silver, 2016), etc., can be used. The learning process is illustrated in Figure 2.

5.1. Deep Q-Network

In the DQN approach, we aim to learn the optimal Q-function, denoted as $Q(s,a)$, which represents the maximum expected accumulated discounted reward achieved by following the alert policy starting from state s and action a in the simulated prognostics environment. Once the optimal Q-function is obtained, the optimal alert policy can be derived by selecting the action that maximizes $Q(s,a)$, i.e.,

$$u(s) = argmax_a\ Q(s,a) \quad (9)$$

This alert policy $u(s)$ outputs the best action to take in each state to maximize the value function. We use a neural network to parameterize the Q function denoted by $Q(s,a;\theta)$. To learn these parameters, there is a replay buffer

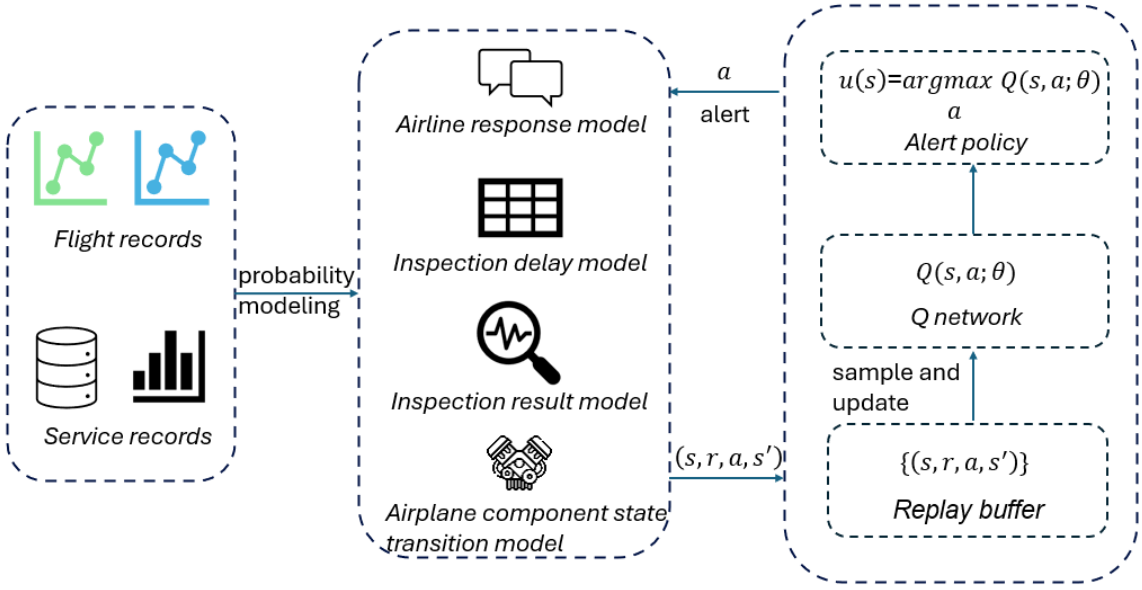

Figure 2. Illustration of deep reinforcement learning for airplane component failure prognostics.

that stores the training data in the form of data tuples $(s,\ r,\ a,\ s')$, corresponding to state, reward, action and next state, respectively. The neural network is trained iteratively. In each training iteration, the alert policy is executed in the simulated prognostics environment to generate new experience tuples, which are then added to the replay buffer. A data batch is randomly sampled from the replay buffer to update parameter θ through the gradient descent method to minimize the following temporal difference error

$$\delta = Q(s,a) - (r + \gamma max_a Q'(s',a)) \quad (10)$$

where γ is a discount factor and $Q'(s',a)$ is a separate target Q function that is a copy of Q function but are updated less frequently that helps to stabilize the learning procedure.

5.2. Double Deep Q Network with Prioritized Replay Buffer

In the airplane components failure prognostic framework, we only get a positive reward when there is a successful alert, which only happens once for the whole component lifecycle. Such a sparse reward setting makes it very challenging to learn the optimal decision-making policy. We used the prioritized replay buffer (Schaul, Quan, Antonoglou, & Silver, 2015) to address the reward sparsity issue. The prioritized replay buffer technique prioritizes the training data samples based on their importance quantified by temporal difference error. To further address the overestimation bias in DQN approach, we applied DDQN that decouples max operation of target Q function in Eq. (10) to the action selection and action evaluation procedure, captured by

$$\delta = Q(s,a) - (r + \gamma Q'(s', argmax_a Q(s',a))) \quad (11)$$

This DDQN prevents overestimation of the Q value function by using different Q functions for action selection and Q value evaluation. Specifically, the current Q-network is used to select the best action, while a separate target Q-network is used to evaluate the value of that action, thereby providing more accurate and stable value estimates.

6. POLICY DEPLOYMENT AND EVALUATION

After training the policy, we need to evaluate the performance of the policy. In addition to the reward value obtained by the policy running in the simulation environment, we use three other evaluation metrics to provide a more straightforward evaluation that is better aligned with business operation.

1) **Precision:** the ratio of alerts that lead to successful detection of airplane component failure to the total number of alerts generated by the alert policy. This metric reflects how often the alerts correspond to actual airplane component issues, indicating the effectiveness of the alert policy.

2) **Recall:** the ratio of the number of airplane component failures that are successfully alerted to the total number of failures. This value shows how often failures are successfully alerted.

3) **Inspection Redundancy:** the ratio of the number of unnecessary inspections that are triggered by false alerts on healthy components to the total number of inspections conducted by airline on the airplane component.

These three metrics capture the accuracy of alerts and the ability to detect all airplane component failures, and the associated inspection costs from the perspective of business operation. Ideally, an optimal policy has high precision and recall scores, and low inspection redundancy.

After the policy is learned in simulated environment, we deploy the policy to the actual airplane components failure prognostic procedure. Each day the service organization receives flight records $F(t)$ and service records $S(t)$ from airlines, those records as well as flight records from previous days, up to N records in total, where N depends on pre-defined service rules and past operation experience, are concatenated to form a state $S(t)$ and are given as input to the alert policy.

A key deployment challenge is to handle missing and out-of-sequence records, which disrupt the time order and result in variable-length inputs. To address this, we reorder the records into the correct temporal sequence and apply zero-padding to fill in any missing data, ensuring consistent input dimensions for the policy.

Moreover, as new flight and service records come in, we update the environment models with new data and perform continual training of policy in the new environment models to adapt to the new data. Additionally, we adjust reward parameters based on airline operation needs to balance various operational costs.

7. CASE STUDY

We conducted a case study on applying our deep reinforcement learning approach for ACM failure prognostics. In the following subsections, we describe the ACM failure prognostics procedure and experiment results.

7.1. Air Cycle Machine Failure Prognostics

ACM is a crucial component in a commercial airplane. Its main function is to provide cabin air conditioning and pressurization by utilizing the principles of thermodynamics. The ACM operates by taking compressed air from upstream components and then expanded through a turbine. This process cools the air, and the cooled air is then directed into the cabin and other compartments. In our case study of prognostics procedure for ACM, we have flight and service records on ACM from **17** airlines. Each airline provides sensor data for ACM, its lifecycle information, historical alerts and inspection records regarding an ACM. In total we have records for **257** ACM lifecycles. The source data column and data descriptions are summarized in Table 1. There are two ACMs in an airplane; we only show examples for the left side ACM here. Among these ACM data, the LLabel is Failure prediction label using a data-driven model, while LTruth is the ground truth label. In our study, we include LLabel into our state as it captures component features implicitly, while we don't include LTruth to avoid label leakage.

Table 1. Data from airlines on ACM.

Source data column	Data description
Tail	Airplane ID
Leg	Time of the flight record
LLabel	Failure prediction label using a pretrained model
LTruth	True failure label
LTT0Seconds	Duration of ACM from running to stopping
LDaysFromRemoval	Number of days from last removal
LDaysToRemoval	Number of days to next removal
AlertTime	Alert sent time
InstallTime	ACM installation time
RemovalTime	ACM removal time
InspectionTime	ACM inspection time
InspectionSummary	ACM inspection result

7.2. Flight Data Processing

Our deep reinforcement learning method is designed to effectively capture data characteristics of flight and service records, which usually contain out-of-sequence and noisy data. Moreover, sensor data are sometimes missing for certain time periods. To ensure good performance, we need to process the data to make it ready and clean for use.

The flight records, including sensor data and flight logs are shared by airlines. Each record includes a date, flight number, and basic information of the airplane component such as the airplane registry number. It also contains the number of days since the previous removal (except for the first installation on that airplane) and the number of days to the next removal (except for the last installation on that airplane, i.e., the component is still on-wing). Moreover, the flight record includes features extracted from the component's sensor data, along with a failure prediction label generated by a pre-trained model that uses these features as input. For example, in ACM, one of the features is TT0seconds, which indicates the number of seconds it takes for ACM from a consistent running state to full stop. Generally, the smaller the value of

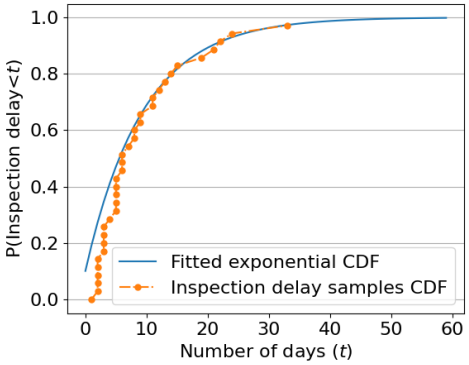

Figure 3. Fitting inspection delay model using exponential distribution.

TT0seconds, the larger probability of ACM malfunctioning, as degrading ACMs with higher frictions usually stop more quickly. Since flight records can miss entries, we apply a filter to get rid of flight records lacking important features such as number of days since its previous removal, number of days to its next removal, and sensor information.

In addition, we need to extract individual lifecycles of components. A complete lifecycle is obtained by checking each flight record in chronological order, if any of the following three conditions is satisfied, we reach the end of a component lifecycle, which means it is either replaced or there are no more available flight records.

- The registry number of the current flight record is different from the next one, which means flight record changes to a new airplane.
- The number of days since the last removal drops. For the same airplane component, the number of days from the last removal should continually increase.
- The number of days until the next removal jumps. For the same airplane component, the number of days to its next removal should continually decrease.

Another valuable dataset we utilize is the service records, which capture the interaction history between BGS and airlines. This includes historical alert times, inspection schedule times and inspection results, removal time of airplane component. Moreover, airlines sometimes perform inspections on airplane components proactively, which are also recorded in the service record. We only use records with alerts, which can be extracted based on entries with available alert time.

7.3. Model Fitting

With the ACM data from airlines, for each airline, we first fitted several models discussed in Section 4.3 including the airline response model, inspection delay model, inspection result model and airplane component state transition model to build an ACM prognostics simulation environment. The airline-specific inspection probability model is estimated based on the ratio of number of inspections to the total number of alerts. On average, the inspection probability for 17 airlines is 0.732. The inspection delay model, illustrated in Figure 3, demonstrates that the exponential distribution effectively captures the characteristics of inspection delays. To further validate the model fitting, we computed the one-sample KS statistic and CvM statistic (D'Agostino & Stephens, 2017) to measure the maximum deviation and average squared distance, respectively, between samples CDF and fitted exponential distribution CDF. The KS statistic value is 0.0842 and CvM statistic value is 0.0367. Moreover, the inspection result model, as shown in Figure 4, reveals that as the days to removal get closer and the inspection delay gets smaller, the inspection failure probability increases. This finding aligns with the discussions on factors that have an impact on inspection results in Section 4.3.3.

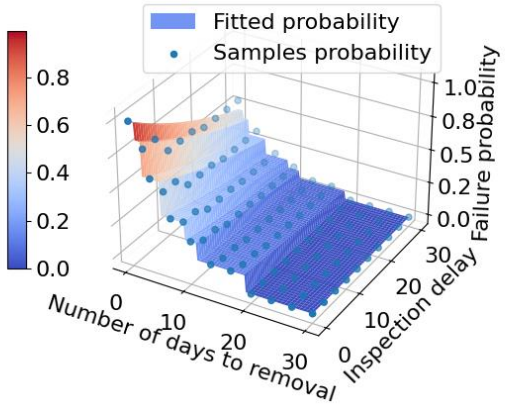

Figure 4. Fitted inspection result probability model.

Table 2. Reward for different operations.

Operation	Reward value
Send alert	-0.1
Unnecessary alert	-0.5
Unnecessary inspection	-30
Successful alert	100
Missing ACM failure	-100

7.4. Training Results

We trained our deep reinforcement learning approach on a Dell Laptop equipped with 12th i7-12800H, 2.4GHz, 32GB RAM using CPU-only computation. The dataset was split into 73% training data and 27% evaluation data by using a

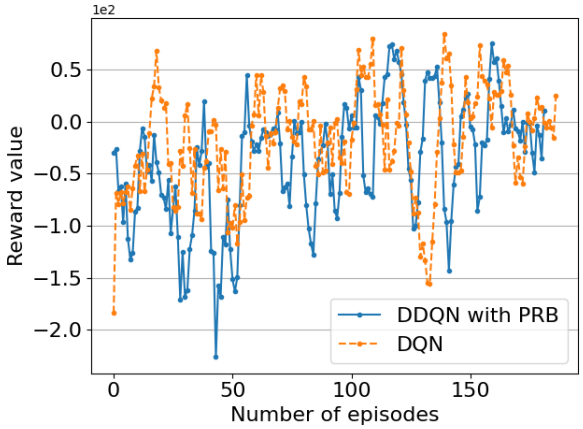

Figure 5. Training rewards with number of episodes for DQN and DDQN with PRB.

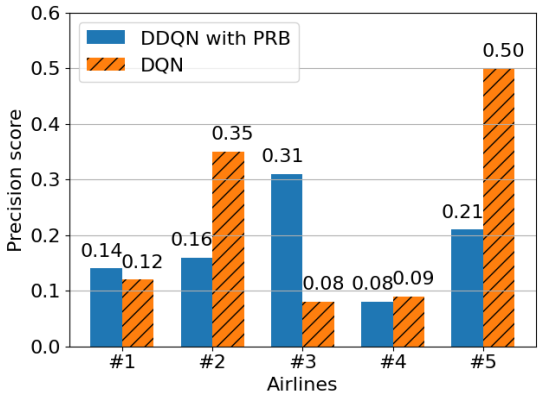

Figure 6. Precision scores of DQN and DDQN with PRB for different airlines ACM prognostics.

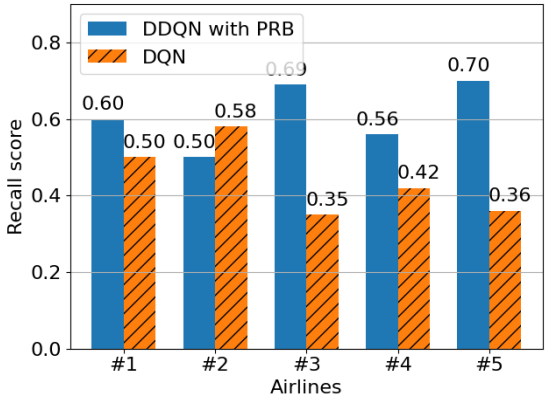

Figure 7. Recall scores of DQN and DDQN with PRB for different airlines ACM prognostics.

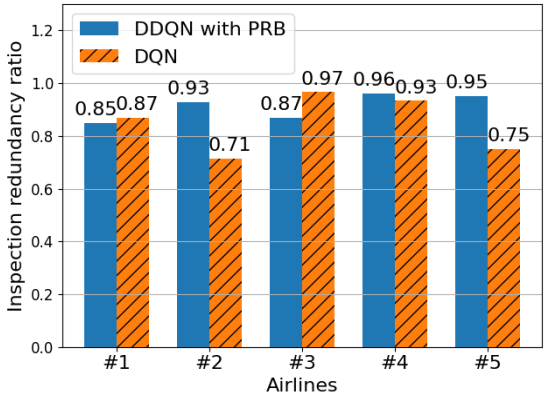

Figure 8. Inspection redundancy ratio of DQN and DDQN with PRB for different airlines ACM prognostics.

cutoff date of 05/01/2023 to separate records from each airline accordingly. And we set the reward for different operations as summarized in Table 2 based on general operational requirements. A large reward is given for issuing a successful alert, while an equally significant penalty is applied for missing an ACM failure. A smaller penalty value is given to unnecessary inspections. Minor penalties are given to sending alerts or unnecessary alerts.

We let the state $s(t)$ defined in Section 4.1 include time sequence vectors with length 100 and each vector contains four features including LDaysFromRemoval, LLabel, LTT0Seconds as described in Table 1 and a binary tag to indicate if there is a scheduled inspection to avoid sending unnecessary alerts. Both the Q Network and target Q Network are represented by an LSTM module followed by three fully connected layers with ReLU activation units (Nair & Hinton, 2010). Hidden size of the LSTM module is 64 and the three fully connected layers have 64, 64, 32 units. The learning rate for both DQN and DDQN with PRB are set to 0.001. The batch size is set to 64. The size of replay buffer and the parameter of prioritized sampling for PRB are set to 100000 and 0.1, respectively.

During training, both DQN and DDQN with PRB use ϵ-greedy selection strategy (Sutton & Barto, 2018), in which with probability ϵ the agent selects a random action to explore and with probability $1-\epsilon$ it selects the greedy action that maximizes the current Q-value function. DDQN with PRB adopts a ϵ power decay process with minimum ϵ set to 0.01 and power decay factor of 0.99. As learning progresses, the policy shifts from exploration toward exploitation. DQN uses a fixed epsilon with value 0.02. The discount factor γ is set to 0.99. The parameters of Q networks are updated at every transition step.

It takes around 43 minutes to train DQN algorithm and around 56 minutes to train DDQN with PRB. Figure 5 shows the training rewards over episodes for both algorithms. We can see from the figure that the training reward gradually increases with the number of episodes, indicating that both algorithms are optimizing the alert policy. We can also see that DQN achieved a similar reward level to DDQN with PRB.

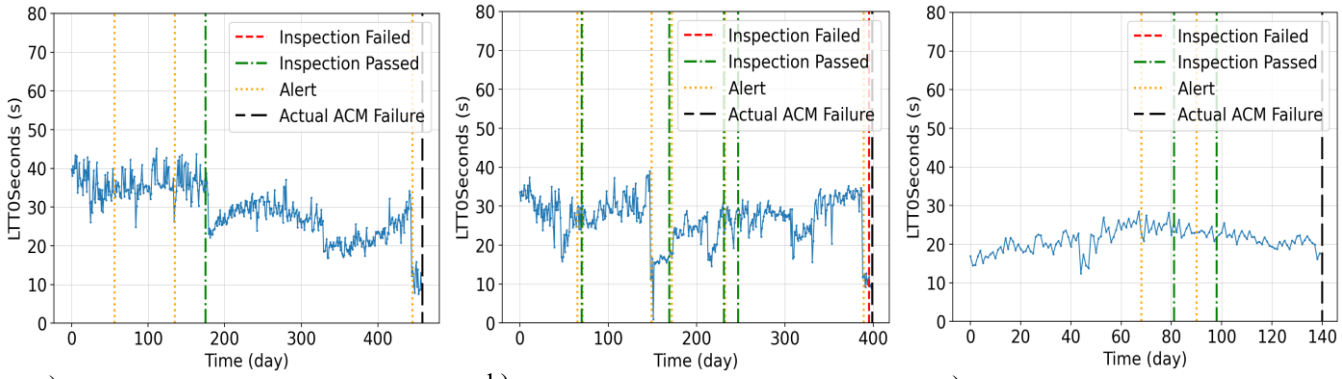

Figure 9. Examples of running DDQN with PRB for Airline #3 in simulated ACM prognostics procedure using real data.

7.5. Evaluation Results

After training the alert policies, we evaluated their performance by applying them to the prognostics procedure for five airlines using the evaluation dataset. To protect proprietary information of airlines, the airlines are anonymized and labeled as Airline #1 through #5. Moreover, as discussed in Section 6, we adopt precision, recall scores and inspection redundancy as key evaluation metrics. The evaluation results are summarized as follows.

Precision scores of DQN and DDQN with PRB for different airlines are shown in Figure 6, which shows that DQN achieves higher precision scores than DDQN with PRB for Airline #2, #4 and #5, but lower scores in Airline #1 and #3. As for recall scores shown in Figure 7, DDQN with PRB outperforms DQN in all airlines except Airline #2.

Comparing Figure 6 and Figure 7 reveals that both approaches have higher recall scores than precision scores for most airlines. This is because both approaches send many alerts, even though most of them are unnecessary ones. In other words, the alert policy is aggressive in sending alerts to capture any possible actual component failure even if there are many false alerts. This is validated by the inspection redundancy ratio shown in Figure 8. The figure shows that both DQN and DDQN with PRB have large inspection redundancy ratio. DQN achieves a relatively smaller redundancy ratio.

Furthermore, we also compared DQN and DDQN with PRB approaches with currently deployed heuristics approach for ACM prognostics. The average precision and recall scores of three approaches are summarized in Table 3. As shown in the table, the current heuristic approach has the highest precision score, however, DQN and DDQN with PRB approaches have much higher recall scores than heuristic approach.

Table 3: Precision and recall scores of three approaches.

	Precision	**Recall**
DQN	0.23	0.44
DDQN with PRB	0.18	0.61
Heuristic Approach	0.35	0.25

Note that in our deep reinforcement learning approach, we can adjust reward weights to balance the precision and recall scores which can potentially have both higher recall and precision scores than the heuristic approach. This flexibility also enables more adaptable business operation.

For illustration, in Figure 9, we show three representative prognostics outcomes from the DDQN with PRB alert policy for Airline #3, simulated using models derived from real data. Scenario 1 (Figure 9.a): an alert was issued within 30 days of an actual ACM failure, but no inspection returned “Failed”; the policy still receives a high reward because it correctly anticipated the failure. Scenario 2 (Figure 9.b): an alert triggered an inspection that returned “Failed,” confirming the fault and earning a high reward. Scenario 3 (Figure 9.c): no alert was issued within 30 days of failure, indicating a missed detection. Across these examples, failed ACMs typically exhibit smaller LTT0Seconds, i.e., shorter time from running to stop.

7.6. Impact of Reward Parameters

The performance of deep reinforcement learning approach for ACM prognostics is sensitive to reward parameters, particularly the weights assigned to different operational outcomes. These weights directly influence the precision and recall scores of the alert policy. To evaluate this impact, we compare the precision and recall scores of DQN trained alert policy under different reward weights for unnecessary inspections and successful alerts, which is shown in Figure 10. We can see that as the reward for unnecessary inspection

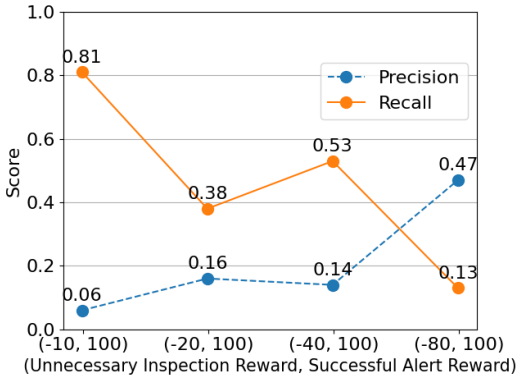

Figure 10. Precision and recall scores of DQN trained alert policy with different reward weights.

decreases from -10 to -80, the precision score has a trend to increase while the recall score tends to decrease. This occurs because smaller unnecessary inspection reward discourages the alert policy sending excessive alerts, thereby reducing unnecessary inspections. As a result, the precision score increases due to fewer false alerts, while the recall score decreases as the alert policy becomes more conservative, increasing the likelihood of missing actual component failures. Therefore, selecting appropriate reward weights is crucial to balancing the trade-off between precision and recall, ensuring the alert policy aligns with operational priorities.

8. CONCLUSION

In this paper, we developed a reinforcement learning-based approach to automate full cycle airplane component failure prognostics and enhance alert timing accuracy. Our approach leverages probabilistic data modeling derived from real flight and service records to simulate the prognostics process and adopts an LSTM neural network to represent the alert policy. We explored two deep reinforcement learning algorithms: DQN and DDQN with PRB to train the policy. A case study on ACM prognostics was conducted to evaluate our approach. The results demonstrated that our approach achieves much higher recall scores, but slightly lower precision scores compared to the heuristic approach currently deployed at BGS. Furthermore, we highlight the inherent trade-off between precision and recall, which can be balanced by tuning reward weights according to operational priorities and maintenance costs. In the future, we will explore other reinforcement learning algorithms such as Deep Deterministic Policy Gradient (Lillicrap, Hunt, Pritzel, Heess, Erez, Tassa, Silver, & Wierstra, 2015) to further enhance precision and recall scores and reduce inspection redundancy.

NOMENCLATURE

$F(t)$	Flight records at time t
$S(t)$	Service records at time t
$a(t)$	Alert action at time t
$f(\cdot)$	Alert model
$r(\cdot)$	Reward function
$P(\cdot)$	Probability mass or CDF
$p(\cdot)$	Probability density function
X	Bernoulli random variable to denote if an inspection is scheduled or not
Y	Continuous random variable to denote inspection delay
Z	Bernoulli random variable to denote if an inspection result is Failed or Passed
W	Continuous random variable to denote number of days between inspection and removal
$s(t)$	State at time t
s'	State at next time step
$u(\cdot)$	Alert policy
$Q(\cdot)$	Value function
$Q'(\cdot)$	Target value function
N	Number of records
T	Terminal time of airplane component life cycle

REFERENCES

Kulkarni, C., Schumann, J., & Roychoudhury, I. (2018). On-board battery monitoring and prognostics for electric-propulsion aircraft. *2018 AIAA/IEEE Electric Aircraft Technologies Symposium (EATS)* (pp. 1-12), July 9-11, Cincinnati, Ohio. doi: 10.2514/6.2018-5034

Pidaparthi, B., Jacobs, R., Ghosh, S., Ravi, S. K., Amer, A. W., Luan, L., ... & Wang, L. (2024). Proactive Aircraft Engine Removal Planning with Dynamic Bayesian Networks. *Annual Conference of the PHM Society* (Vol. 16, No. 1), November 9-14, Nashville, Tennessee. doi: https://doi.org/10.36001/phmconf.2024.v16i1.4148

Dangut, M. D., Skaf, Z., & Jennions, I. K. (2021). An integrated machine learning model for aircraft components rare failure prognostics with log-based dataset. *ISA transactions*, 113, 127-139. doi:10.1016/j.isatra.2020.05.001

Hu, Y., Miao, X., Zhang, J., Liu, J., & Pan, E. (2021). Reinforcement learning-driven maintenance strategy: A novel solution for long-term aircraft maintenance decision optimization. *Computers & industrial engineering*, 153. doi:10.1016/j.cje.2020.107056

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529-533. doi: https://doi.org/10.1038/nature14236

Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double q-learning. *AAAI conference on artificial intelligence* (Vol. 30, No. 1),

February 12-17, Phoenix, Arizona. doi: https://dl.acm.org/doi/10.5555/3016100.3016191

Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. *arXiv preprint*:1511.05952.

D'Agostino, R., & Stephens, M. (2017). *Goodness-of-fit-techniques*. Routledge.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.

Nair, V., & Hinton, G. E. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines. *27th International Conference on Machine Learning (ICML 2010)*, June 21-14, Haifa, Israel. doi: https://dl.acm.org/doi/10.5555/3104322.3104425

Graves, A., Fernández, S., Gomez, F., & Schmidhuber, J. (2006). "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks." *23rd International Conference on Machine Learning*, July 9-15, Pittsburgh, PA. doi: https://dl.acm.org/doi/10.1145/1143844.1143891

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver. D., & Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint*:1509.02971.

BIOGRAPHIES

Baoqian Wang is an Intelligent Systems Engineer at Boeing AI, where he focuses on developing advanced AI and robotics methods to improve airplane operation, manufacturing and maintenance efficiency, with interests in reinforcement learning, unmanned aircraft systems, multi-agent systems, software and hardware automation techniques. He received his Ph.D. in Electrical and Computer Engineering from University of California San Diego and San Diego State University in 2023 and his M.S. in Computer Science from Texas A&M University, Corpus Christi in 2019. He is passionate about advancing innovation of AI and robotics algorithms and their real-world applications.

Changzhou Wang received his Ph.D. in Information Technology from George Mason University, Fairfax, Virginia, USA in 2000. He joined Boeing in Sept 2000 and is now a Technical Fellow and data scientist in Boeing Global Services. He was PI, Co-PI and technical lead for government research contracts in data mining, information security and software quality of service. His current research interests include developing temporal data analytics technologies, adapting advanced artificial intelligence and machine learning methods and combining large-scale flight sensor data and deep engineering knowledge for prognostic modeling.

Denis Osipychev is an Associate Technical Fellow in AI/ML at Boeing Research & Technology and a PhD candidate in Computational Science and Engineering at the University of Illinois Urbana–Champaign, with an MS in Electrical and Computer Engineering from Oklahoma State University. His research and technical leadership focus on reinforcement learning, robust decision-making under uncertainty, and assurance for safety-critical autonomous systems; notable contributions include the development and patenting of a learning-based Collision Avoidance System (RL-CAS) and the prototype development of a high-performance air-combat agent for the DARPA AlphaDogfight Trials. He has advanced surrogate-simulation training and training-hardening methods, built reproducible MLOps and benchmarking pipelines, and focuses on translating rigorous research methods into deployable, verifiable assurance practices for trusted autonomous systems.