aerospace

MDPI

Article

An Information Integration Technology for Safety Assessment on Civil Airborne System

Xi Chen [1,2,†], Quan Zou [2,3,†], Jie Bai [1,2,*] and Lei Dong [1,2]

[1] Science and Technology Innovation Research Institute, Civil Aviation University of China, Tianjin 300300, China; chen_x@cauc.edu.cn (X.C.); l-dong@cauc.edu.cn (L.D.)
[2] Key Laboratory of Civil Aircraft Airworthiness Technology, Tianjin 300300, China; 2022092130@cauc.edu.cn
[3] School of Safety Science and Engineering, Civil Aviation University of China, Tianjin 300300, China
* Correspondence: jbai@cauc.edu.cn
† These authors contributed equally to this work.

Abstract: With the significant expansion of civil aviation, particularly in the low-altitude economy, there is a significant gap between the escalating demand for airworthiness certification of novel aircraft designs, such as electric vertical take-off and landing (eVTOL) vehicles, and the inefficiency of the current safety assessment process. This gap is partially attributed to safety assessors' limited exposure to these innovative aircraft models in the safety assessment process, necessitating extensive efforts in identifying precedents and their handling strategies. Complicating matters further, pertinent case studies are scattered across diverse, unstandardized digital formats, obliging assessors to navigate voluminous electronic records while concurrently establishing links among fragmented information scattered across multiple files. This study introduces an advanced information integration methodology, comprising a multi-level path-based architecture and a self-updating algorithm. The proposed method not only furnishes safety assessors with pertinent knowledge featuring explicative interconnectedness automatically, but also dynamically enriches this knowledge corpus through operational usage. Additionally, we devise a suite of evaluative criteria to validate the capacity of our method in processing and consolidating relevant safety datasets. Experimental analyses affirm the efficacy of our proposed approach in streamlining and refreshing safety assessment data. The automation of the retrieval of analogous cases, which relieves the reliance on expert knowledge, enhances the efficiency of the overall safety appraisal procedure. Consequently, this research contributes a solution to enhancing the velocity and accuracy of aircraft certification processes.

Keywords: knowledge graph; multi-level; path-based; safety assessment; civil aviation

Citation: Chen, X.; Zou, Q.; Bai, J.; Dong, L. An Information Integration Technology for Safety Assessment on Civil Airborne System. *Aerospace* **2024**, *11*, 459. https://doi.org/10.3390/aerospace11060459

Academic Editor: Sujin Bureerat

Received: 19 April 2024
Revised: 29 May 2024
Accepted: 2 June 2024
Published: 6 June 2024

1. Introduction

Safety assessment is vital in the design phase of civil aviation systems. Its primary objectives are to prevent accidents, mitigate their severity, and reduce operational risks, making it a complex and empirically rich endeavor. This assessment process is crucial for demonstrating compliance with airworthiness regulations. It involves evaluating system operations under various conditions, identifying potential accidents, assessing their probable impact, and recommending strategies to eliminate, mitigate, or manage hazards [1–3].

Consider the following scenario: a safety assessor is conducting an aircraft functional hazard assessment for Model A and is required to complete the item *Impact on Aircraft* of the failure condition *Uncommanded Ground Deceleration*. In order to fulfill the item, the assessor must refer to documented impacts under the same failure condition for Model B. In the event that this information is unavailable or that further examples are required, the assessor may need to consult Model C or even additional models. A lack of relevant experience can result in work stoppages as assessors must consult historical cases, which are often stored in disparate electronic files. It is crucial for assessors to identify the inherent connections among multiple cases, which can diminish the efficiency and quality of the

safety assessment process [4,5]. In recent years, there has been a substantial surge in the development of the low-altitude economy, accompanied by the completion of numerous novel aircraft designs, such as electric vertical take-off and landing (eVTOL) vehicles. A pivotal impediment to their widespread market adoption lies in the protracted duration of airworthiness certification processes. This challenge is deeply rooted in the lack of accumulated expertise among safety assessors concerning the peculiarities of these emerging technologies, necessitating extensive efforts in case study research and the consolidation of information gleaned therefrom.

Consequently, this study addresses two research questions:

Q1: What methodologies can be employed to enhance the accessibility of data dispersed across diverse documentation pertinent to the model undergoing safety assessment, and how might we effectively establish connections among these disparate data points?

Q2: How can continuously generated data be automatically updated?

This paper endeavors to address the aforementioned research questions through the following approaches: (1) The construction of a knowledge graph that integrates information from electronic documents and extracts coupling relationships, facilitating its use by safety assessors. (2) The establishment of an updating system capable of automatically importing new data, making the process imperceptible for users.

Addressing these existing issues, we propose an information integration method tailored to the safety assessment process on a civil airborne system. More specifically, we first propose a multi-level, path-based domain knowledge graph for safety assessment on civil airborne systems (SACAS) tailored to processes combined with safety assessment cases, in which we employ the multi-level path-based (MLP) architecture to construct it. The MLP architecture facilitates hierarchical storage of vast data and embeds process paths into the knowledge graph, making it the optimal solution for constructing knowledge graphs tailored to processes and cases. Upon the completion of the SACAS construction, we next introduce an updating and maintaining automatically (UMA) algorithm. The UMA algorithm sifts through the relevant portions of new cases from input databases and imports them into SACAS, facilitating interconnection of safety assessment cases. In particular, we design a Valid or Invalid (VORI) gate to filter valid data by computing semantic similarity. Figure 1 presents the flowchart of the information integration method.

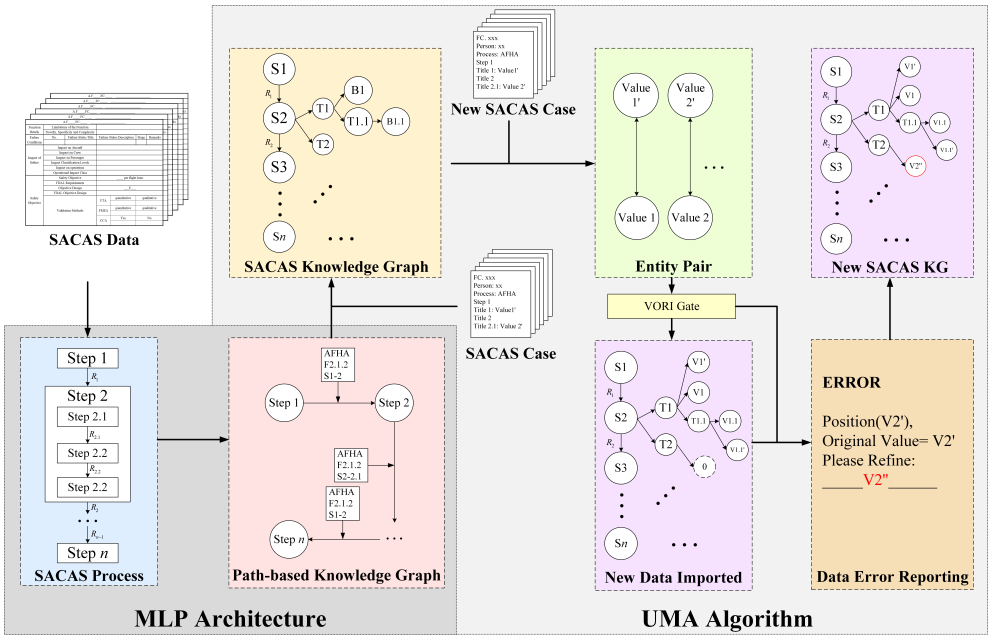

Figure 1. The flowchart of the information integration method.

The remainder of this paper is organized as follows. Section 2 briefly introduces related works of information integration techniques and knowledge graph construction methods. In this section, we clarify the research objectives and scope of this paper. Section 3 elaborates on the method of utilizing the MLP architecture to construct SACAS and provides detailed insights into the MLP architecture. Section 4 introduces the proposed UMA algorithm and its module functionalities, achieving the automatic updating and maintenance of SACAS. Section 5 presents the experimental results and analysis. Section 6 provides the conclusion.

2. Related Works

SAE ARP 4761A [3] outlines the safety assessment process applicable to civil aircraft, systems, and equipment. This standard provides guidelines for several key safety assessment activities, integral to ensuring the airworthiness and safety of new aircraft designs and modifications to existing designs. The safety assessment process described in *SAE ARP 4761A* highlights the universal steps and methodologies used across different aircraft models, emphasizing the shared characteristics of the safety assessment procedures. This approach underlines the comparability and transferability of safety assessment data among different aircraft models, simplifying the evaluation process by focusing on common procedural elements rather than model-specific peculiarities.

Moreover, *SAE ARP 4761* introduces a Model-Based Safety Assessment (MBSA) method in the appendix, which diverges from the universal safety assessment process by accounting for the unique system architectures of different aircraft models. Since it was introduced in *SAE ARP 4761*, MBSA has attracted considerable academic attention, reflecting a burgeoning research landscape dedicated to improving and refining these model-based safety assessment techniques [6–11]. The MBSA method is critical for aircraft models with distinct system configurations, as it tailors the safety assessment to reflect the specific interdependencies and architectural nuances of each model, ensuring a thorough evaluation of potential safety issues specific to each aircraft type.

This study focuses on the generic safety assessment processes outlined in *SAE ARP 4761A*. The structure of safety assessment data remains consistent across different aircraft models and their systems and components [3].

Some of the existing safety assessment data are stored as electronic files. It is difficult for a safety assessor to make full use of past case data in practical use because of the lack of substantial interactivity of these case data. In addition, assessors need to follow certain writing conventions in completing the materials. Although the case data stored in electronic files can provide some reference for them, the process is still time-consuming and error-prone.

There is currently no common attempt to solve this problem. Therefore, this makes full use of the correlation between the case data to provide valuable and accurate references for safety assessors, which reduces their work difficulty and improves their work efficiency.

In general, most information integration techniques are implemented through "centralized" storage. This kind of approaches uses databases and local servers to store information. However, these approaches has obvious drawbacks in terms of data ownership and data tampering. Xu [12] applied blockchain technology to revolutionize scientific marine data integration and sharing. By implementing decentralized storage management, he ensures traceability and data integrity in projects of the National Marine Data Center's marine science and technology programs.

Like blockchain technology, a knowledge graph is a decentralized way of storing knowledge. The implementation and development of this approach also have significant long-term implications. A knowledge graph is a graph-based semantic network, which represents relationships between entities as edges in a graph and entities as nodes in a graph. This structure enables knowledge graphs to effectively capture the correlations and semantic information between knowledge, providing powerful support for linking and querying between data [13–15].

The applications of knowledge graphs cover a wide range of fields, including information retrieval [16,17], question and answer [18–20], knowledge reasoning [21–23], and intelligent recommender systems [24,25]. Knowledge graphs first appeared in 2012, but it is only in recent years, with the rapid development of technologies such as deep learning and natural language processing, that knowledge graphs have gradually become one of the most active areas of research [26–28]. Nowadays, more and more scholars and research institutes have started to devote themselves to the construction and application of knowledge graphs, which promotes the continuous progress and innovation in this field. Wu [29] proposed an intelligent genealogy knowledge graph construction method by combining the AI big model, which realized the knowledge reasoning of the genealogy knowledge graph. Zha [30] implemented M2ConceptBase using a context-aware multi-modal symbol grounding approach to align concepts with images and descriptions. This knowledge base enriched large multi-modal models' (LMMs) cross-modal alignment, improving concept understanding and model performance. Jin [31] proposed an opinion summarization framework based on multi-modal knowledge graphs (MKGOpinSum) to utilize structural knowledge in multi-modal data for opinion summarization. Jin proposed a novel common-attention-based multi-modal embedding framework named CamE for the multi-modal knowledge graph complementation task. The method can capture the textual semantic relations and improve the completeness of the knowledge graph.

Although existing databases and information management systems are widely used across various fields, they face specific challenges in safety assessment. These systems often struggle to flexibly handle highly complex and dynamically changing safety data, especially when they involve cross-database relational queries and real-time data updates. Moreover, traditional relational databases have limitations in representing complex relationships and providing support for semantic querying. In contrast, knowledge graphs, with their unique graphical structure, can more naturally represent the myriad relationships between entities and support complex queries and analyses, which are crucial for safety assessment. Hence, we targeted to propose an information integration technique based on a knowledge graph. It makes use of our proposed MLP architecture and UMA algorithm to construct a multi-level, path-finding SACAS, and realizes automatic updating and maintenance of the knowledge graph. We use Neo4j to construct the SACAS for subsequent related applications.

In civil airborne system safety assessment, the application of a knowledge graph provides a more intuitive and efficient data interaction platform for the staff. By constructing SACAS, staff can easily access relevant case data and conduct deep mining and analysis through the correlations in the graph, thus providing an accurate and reliable reference basis for safety assessment. As a powerful knowledge representation and storage method, knowledge graphs provide a brand-new solution for civil airborne system safety assessment. With the deepening of related research and the continuous development of technology, it is believed that the applications of knowledge graphs in this field will expand, achieving a more extensive and far-reaching impact in the future.

3. Construction of Safety Assessment on Civil Airborne System Knowledge Graph

The safety assessment on civil airborne system (SACAS) knowledge graph is a multi-level path-based knowledge graph for workflow-oriented tasks such as safety assessment. It organizes case data with the relationships between them. This section is mainly about the construction of SACAS which is divided into two parts. Ontology modeling comes first. It can clarify the types of entity and relationship. Then, MLP is employed to realize the multi-level aspect and insert path information into SACAS. For our task, MLP is expected to ensure process integrity while facilitating connectivity and interaction among diverse data sources, offering considerable convenience to safety assessment practitioners. For the method and framework of this paper, the knowledge processing techniques involved in each stage of safety assessment are similar. Therefore, we take the aircraft functional hazard

assessment (AFHA) in the safety assessment process as an example for the construction of SACAS as well as for the subsequent experiments in this paper.

3.1. Ontology Modeling

The construction of a knowledge graph can be categorized into two approaches: top-down and bottom-up. The former one is usually performed in specialized domains with well-defined knowledge scopes, while the latter one is used to cover a wide range of general knowledge. For our process-complete and case-lack task, the top-down approach is more appropriate. The top-down construction of a knowledge graph involves determining the entities, relationship types, and architectural structure of the knowledge graph before populating it with data. Figure 2 illustrates the process of constructing the top-down knowledge graph.

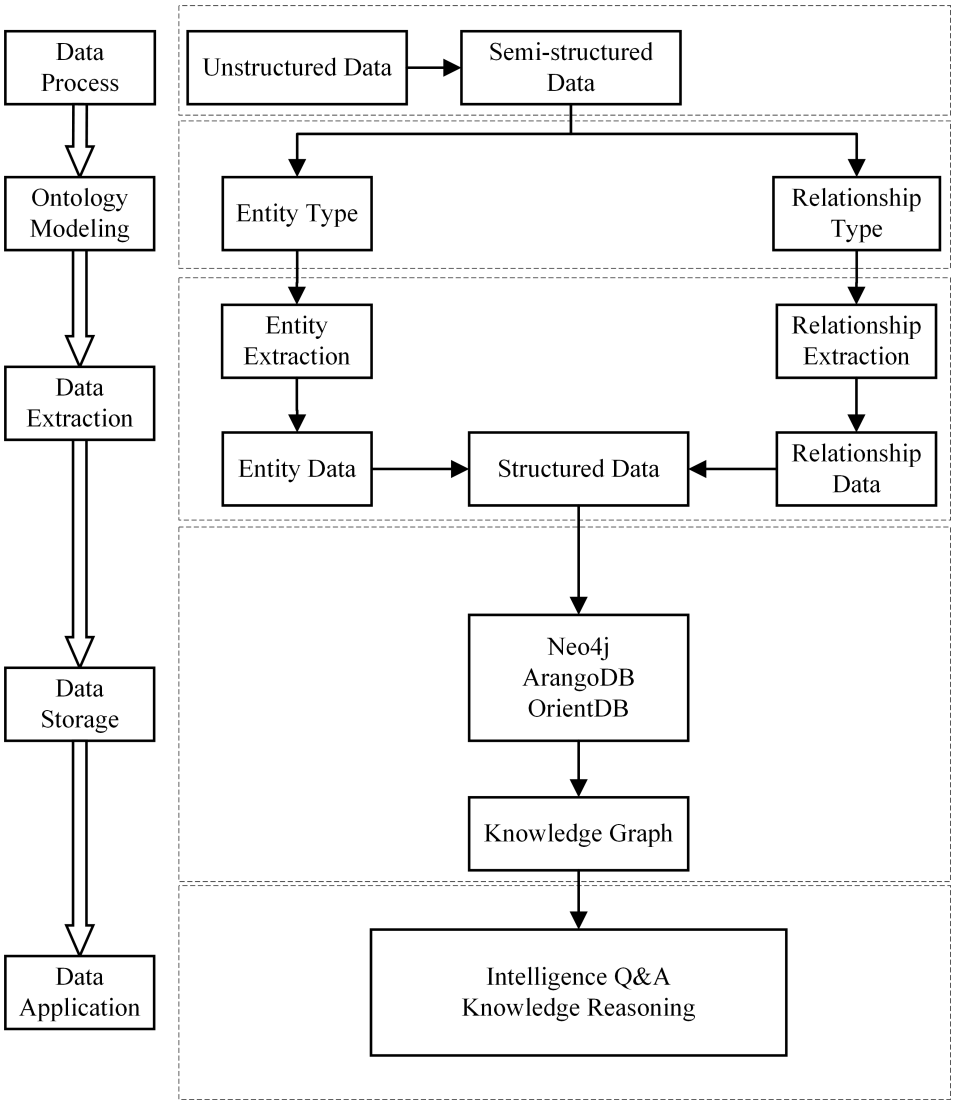

Figure 2. The process of top-down knowledge graph construction.

The constructed graph transforms textual data into a structured form of triplets for storage, thereby better representing textual semantics and facilitating data circulation and correlation.

1. **Ontology Modeling**: determining entity types, attributes of different entities, relationship types, and corresponding relationships between entity types. This step is essential for structuring textual data storage based on the results of ontology modeling.

2. **Knowledge Extraction**: entity extraction and relationship extraction, aiming to extract various entities and relationships from textual data. Machine learning techniques are commonly used for automatic or semi-automatic extraction from datasets.
3. **Knowledge Storage**: using open-source tools such as Neo4j or OrientDB, enabling knowledge storage.

SACAS focuses on its architecture to achieve its multi-level and path-based characteristics. Moreover, due to the scarcity of safety assessment cases, automated extraction methods for obtaining entities and relationships are not currently employed in this research. The results of the ontology modeling and the architecture of the knowledge graph are the main points of focus in this section.

3.1.1. Key Issues in Ontology Modelling

Researchers at Stanford University proposed several key issues to consider when ontology modeling in 2001 [32]. This subsection appropriately integrates and adapts these issues to the specific context of this study. We determined six key issues to build the SACAS ontology.

Step I: Clarifying the Domain and Scope of the Ontology

We will clarify the domain and scope of the ontology based on the following questions: What domain will the ontology cover? What will we use these ontologies for? What questions will the information in the ontology address? Who will use and maintain the ontology?

It is clear that the SACAS ontologies will cover the domain of safety assessment of civil airborne systems. We utilize these ontologies in order to build SACAS to enable the integration of information for assessment cases. The focus of this paper is on determining how to achieve information integration of civil airborne system safety assessment cases, i.e., to construct a multi-level, path-finding SACAS and to achieve automatic updating and maintenance of the knowledge graph. In future work, we will use the recommendation technology according to the SACAS and the characteristics of civil airborne system safety assessment work, which will provide some assistance to the relevant staff for the assessment work. Database technicians will maintain and use the ontology after new data input.

Step II: Listing Important Terms within the Domain

This step is crucial. Once important terms within the domain are defined, it prevents confusion with other meanings outside the domain.

This is why the terms are listed in Section 3.1. For example, the word "failure" has different meanings in different contexts, and it is necessary to define the use of these terms by isolating the terms within the domain that are prone to ambiguity and determining their meanings. One such term, failure, is clearly defined by *SAE ARP 4754B*:

> *Failure: An occurrence which affects the operation of a component, part or element such that it can no longer function as intended, (this includes both loss of function and malfunction)* [33].

Step III: Determining Classes and Class Hierarchies

This process involves three methods: top-down, bottom-up, or a combination of both. Top-down entails first determining the most general classes, then further classifying them.

This will be shown in Section 3.1.3, where the construction of classes is carried out in a logical order. This ensures the completeness of the classes and improves the efficiency of ontology modelling. For example, the top-level class could be "primary function", followed by "secondary function", and so forth.

Step IV: Identifying Class Properties

Even after determining the classes, the information provided by these classes is not enough. It is necessary to determine the internal relationship of these classes, i.e., attributes.

Attributes are divided into internal attributes of the class, external attributes, and relationships with other classes. It is worth noting that the next level of class inherits all the attributes of the previous level of class.

This step is to build a semantic web by linking the classes just established and adding features to each class to highlight the differentiation. Specific features and the relationship between classes should be analysed according to the specific content. In Section 3.1.3, we will undertake a detailed analysis.

Step V: Determining Constraints on Properties

Constraints on the attributes of different classes include the following: for example, an attribute can only have one value, a class must have a minimum of N attributes, a minimum or maximum of n common attributes between two classes.

Here, again, the relationship between attributes needs to be analysed on a content-specific basis. For example, if the attribute of a class is "model", then the attribute should have a constraint of "the value of this attribute is unique".

Step VI: Creating Instances

This step, a significant outcome of ontology modeling, involves entity types. This process requires:

Step VI.1: *Selecting a class*;
Step VI.2: *Creating a separate instance of that class*;
Step VI.3: *Filling in the attributes of that instance*.

This step is built on the basis of the completion of the first five steps, and is the most critical step in ontology modelling. In this step, we still have to analyze the specific situation and construct the appropriate entities and relationship types guided by the class and inter-class relationships.

By following these six steps, we can complete ontology modeling.

3.1.2. Aircraft Functional Hazard Assessment

Aircraft functional hazard assessment (AFHA) assesses the functional hazards at the aircraft level. It constitutes the top-level segment of the safety assessment process on a civil airborne system and stands as its most crucial component. AFHA comprises four steps [3,34]:

Step 1: ***Individual Failure States***
Step 2: ***Combined Failure States***
Step 3: ***Composite Failure States***
Step 4: ***Summary of Results***

Among these, the assessment of individual failure states is the most crucial step. It involves analyzing one or more failure states for each function based on the classification of aircraft-level functions. Once the analysis of failure states for each function is completed, this step is essentially fulfilled. Figure 3 presents the assessment table of individual failure conditions for aircraft-level functions:

Blanks (1) to (27) are for the assessment of individual failure conditions.

The analysis of failure conditions for a particular function comprises three components: detailed function description, addition of failure conditions, and the impact of failure and safety objectives during different flight phases.

Before constructing the domain knowledge graph, it is crucial to establish some important terms in the domain. The specific reasons will be elucidated in Section 3.1.1. Currently, we only list some important terms related to the safety assessment process: aircraft-level function, failure state, failure impact, flight phase, safety objective, etc.

A.F__(1)__.FC.__(2)__ _____(3)_____					
Function Details	Limitations of the Function		(4)		
	Novelty, Specificity and Complexity		(5)		
Failure Conditions	No.	Failure Condition Title	Failure Condition Description	Phase	Remarks
	(6)	(7)	(8)	(9)	(10)
Impact of failure	Impact on Aircraft		(11)		
	Impact on Crew		(12)		
	Impact on Passenger		(13)		
	Impact Classification Levels		(14)		
	Impact on operations		(15)		
	Operational Impact Class		(16)		
Safety Objective	Safety Objective		__(17)__ per flight hour		
	FDAL Requirements		(18)		
	Objective Design		__(19)__E__(20)__		
	FDAL Objective Design		(21)		
	Validation Methods	FTA	quantitative	qualitative	
			(22)	(23)	
		FMEA	quantitative	qualitative	
			(24)	(25)	
		CCA	Yes	No	
			(26)	(27)	

Figure 3. The assessment table of individual failure conditions for aircraft-level functions.

3.1.3. Results of Ontology Modeling

We used the open-source tool Protégé for ontology modelling and the results are shown in Figure 4.

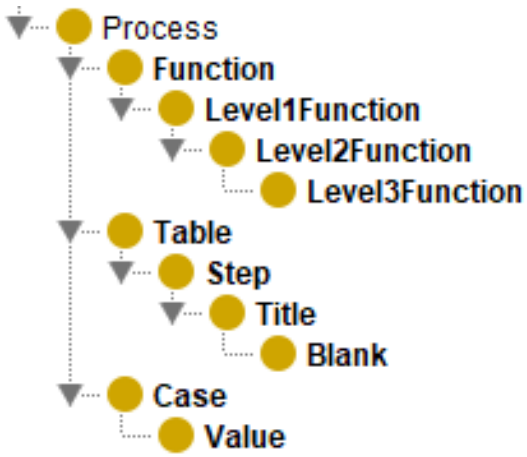

Figure 4. The result of "Class" construction.

From Figure 5, it can be observed that we constructed multi-level classes in a top-down manner: (1) The top-level class is "Process", representing the civil aircraft onboard system safety assessment process, such as AFHA and SFHA. (2) Secondary classes include Function, Table, and Case. (3) Classes at the third level and below are subordinate to the content of each secondary class.

Determining entity types, relationship types, and constraints are crucial steps in ontology modeling for safety assessment. Initially, we start at the engineering level by extracting relevant data from expert knowledge and safety assessment databases to identify key entities and relationships. Entity types such as Process, Level1Function, and Level2Function are determined based on class categorization; Failure Conditions, Impact of failure, and Safety Objectives are established according to the standards and practical engineering requirements in the safety assessment. Relationship types are defined in a path-oriented

manner, which is also based on process-oriented safety assessment, facilitating practical engineering applications. Furthermore, multiple constraints are set to ensure the accuracy and consistency of the knowledge graph. For instance, each Level2Function must be associated with only one Level1Function, ensuring the completeness and simplicity of the knowledge graph.

Through these methods, we have precisely modeled the ontology of the safety assessment knowledge graph, providing a solid foundation for subsequent graph construction and applications. Based on this framework, we added class properties, relationships between classes, and constraint conditions to complete the ontology modeling. Table 1 presents the results of the ontology modeling.

Table 1 contains various entity types and their relationship types, comprising a total of 31 entity types and 30 relationship types. These entity and relationship types form the basis for achieving the multi-level and path-based characteristics of the knowledge graph. With these entity types and relationship types, we can organize and summarize case data accordingly and embed attributes into various entity and relationship types, thereby completing the construction of the SACAS.

Table 1. The results of the ontology modeling.

Entity Type 1	Relationship	Entity Type 2
Process	process-function1	Level1Function
Level1Function	function 1_2	Level2Function
Level2Function	function 2_3	Level3Function
Level3Function	function-table	Table Number
Table Number	Step 1	Function Details
Table Number	Step 2	Failure Conditions
Table Number	Step 3	Impact of failure
Table Number	Step 4	Safety Objectives
Table Number	Step 4	Validation Methods
Function Details	Step 1.1	Limitations of the Function
Function Details	Step 1.2	Novelty, Specificity and Complexity
Failure Conditions	Step 2.1	No.
Failure Conditions	Step 2.2	Failure Status Title
Failure Conditions	Step 2.3	Failure Status Description
Failure Conditions	Step 2.4	Stage
Failure Conditions	Step 2.5	Remarks
Impact of failure	Step 3.1	Impact on Aircraft
Impact of failure	Step 3.2	Impact on Crew
Impact of failure	Step 3.3	Impact on Passenger
Impact of failure	Step 3.4	Impact Classification Levels
Impact of failure	Step 3.5	Impact on operations
Impact of failure	Step 3.6	Operational Impact Class
Safety Objectives	Step 4.1	Safety Objective
Safety Objectives	Step 4.2	FDAL Requirements
Safety Objectives	Step 4.3	Objective Design
Safety Objectives	Step 4.4	FDAL Objective Design
Safety Objectives	Step 4.5	Validation Methods
Validation Methods	Step 4.5.1	FTA
Validation Methods	Step 4.5.2	FMEA
Validation Methods	Step 4.5.3	CCA

3.2. MLP Architecture

Current methods for constructing knowledge graphs are primarily automated or semi-automated, largely relying on natural language processing (NLP) technology. This technology excels at text tokenization and semantic extraction. However, the safety assessment data in this study includes process-based path information and a multi-level structure, which significantly differs from typical textual semantic information. Before choosing the

multi-level path-based (MLP) architecture for building the SACAS, we attempted to use NLP techniques for automatic extraction and construction, but the results were unsatisfactory. This indicates that such methods are not well-suited for handling the complex data structures found in safety assessment. Consequently, tailored to the specific characteristics of safety assessment data and the needs of related work, we have developed a multi-level, path-based knowledge graph architecture.

The MLP architecture represents the optimal solution for constructing knowledge graphs tailored to processes and cases. It is designed to address the data hierarchies and relational complexities encountered in traditional knowledge graphs for safety assessment applications. The MLP architecture enables data from the process to the engineering level to be integrated and queried efficiently by explicitly modelling the different levels and paths of safety assessment data. The architecture diagram of MLP is illustrated in Figure 5.

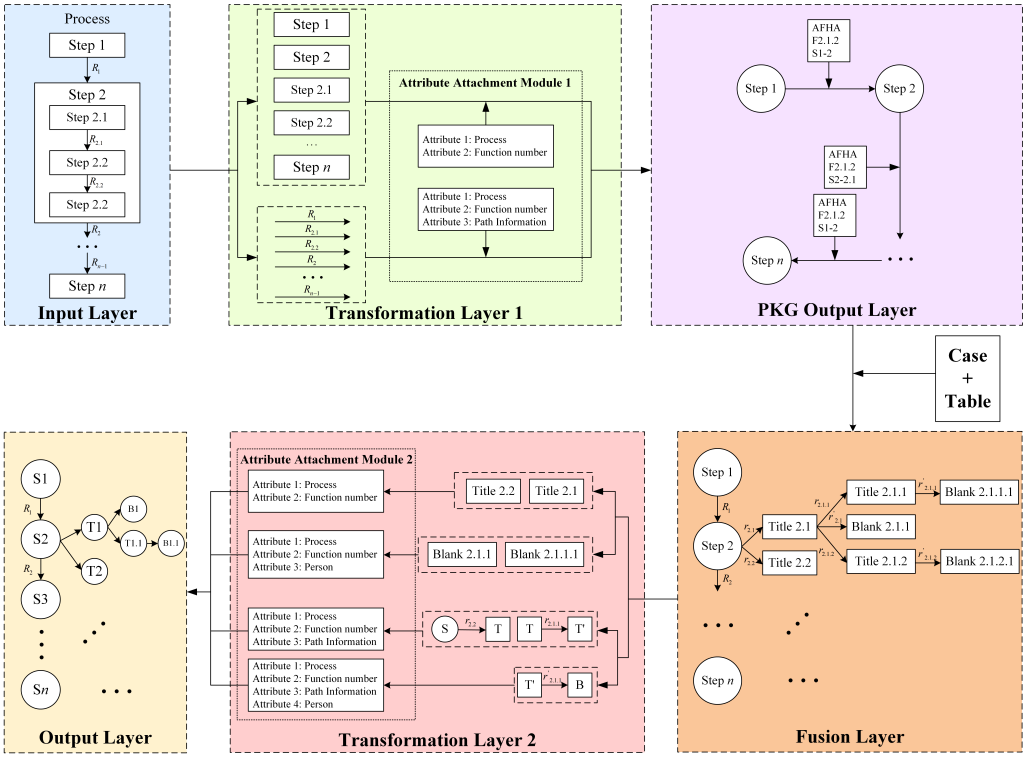

Figure 5. The architecture diagram of MLP.

The MLP architecture starts from an engineering perspective by constructing a knowledge graph based on the safety assessment process and then enriching the database based on relevant cases. From Figure 2, the MLP can be divided into the following aspects:

Input Layer: Extracts the assessment process into a process diagram based on evaluation forms and expert knowledge, which serves as input to the transformation layer. Step i stands for the safety assessment process and each step may contain several substeps. R_i represents the relationship among steps.

Transformation Layer 1 (including Attribute Attachment Module 1): Converts the steps and relationships in the process diagram into entities and relationships, respectively. This is because the next step is to attach different types of attributes to entities and relationships, respectively. Subsequently, Attribute Attachment Layer 1 adds attributes to the entities and relationships. For entities, attributes added include the type of assessment process (e.g., AFHA, SFHA), function number, and case name (usually named after the aircraft model, as different models may have different function classifications); for relationships, attributes added include the path position (e.g., Step 1→Step 2), the type of assessment process, function number, and case name.

Path-based Knowledge Graph Output Layer: Utilizes Neo4j to build a path-based knowledge graph with the entities and relationships output from the transformation layer. At this point, the path information in the security assessment process has been canonically stored.

Fusion Layer: Combines the path-based knowledge graph with safety assessment cases and tables (some functions may lack cases in existing data), forming a tree-like data structure. However, the tree-like structure is evidently different from a knowledge graph. This "tree-like" structure serves as an intermediate state of data input into the next layer. Case data can be likened to plugs being inserted into corresponding sockets (an already constructed PKG). Without this step, the subsequent layer would be unable to assign location information to the imported data. In other words, this step is indispensable, as its omission would lead to difficulties in attaching attributes or result in missing attributes.

Transformation Layer 2 (including Attribute Attachment Module 2): Initially separates the titles and blanks from the safety assessment cases and tables into entities and converts the connecting lines into relationships. Then, attributes are added to the entities and relationships as follows: (1) Titles and blanks from the cases and tables are categorized into titles and blanks (the "blank" in cases contains values, while the "blank" in tables does not). Attributes added to titles include the type of assessment process, function number, and case name; attributes added to blanks include the assessors' names, titles, length of service, units, etc., as well as the type of assessment process, function number, and case name. (2) The connecting lines are categorized into "step-title", "title1-title2", and "title-blank". Attributes added to the first two categories include the relationship position (related to which step and level of title), the type of assessment process, function number, and case name; attributes added to "title-blank" include the relationship position, assessors' names, titles, length of service, units, etc., as well as the type of assessment process, function number, and case name. Additionally, ontology modeling is required to obtain entity types and relationship types, which is demonstrated in Section 3.1.3.

Output Layer: Combines the entities and relationships output from Attribute Attachment Layer 2 with the path-based knowledge graph and completes the construction of the multi-level path-based knowledge graph.

4. UMA Algorithm

4.1. Basic Idea

Here, we establish a premise: Each new case to be input into the SACAS graph has highly consistent safety assessment objects and assessment methods with the existing cases in the graph. Only under such circumstances can we effectively compare and filter them; otherwise, it would be meaningless.

Prior to developing our updating and maintaining automatically (UMA) algorithm, we reviewed existing self-updating methods for knowledge graphs. These typically involve automatic extraction of new data, followed by comparison, selection, and integration with existing graphs to ensure updates. However, the unique challenges posed by safety assessment data, which often includes only partial updates in the form of specific data points rather than complete cases, preclude the direct application of these existing methods. Our UMA algorithm, therefore, is specifically designed to handle the incremental and partial data updates characteristic of safety assessment, ensuring that new information integrates seamlessly and meaningfully with the existing knowledge graph structure, delivering an effective reference for safety assessors.

Section 3 introduces the method of construction for the SACAS, successfully achieving its multi-level and path-based characteristics. When the SACAS is applied to the safety assessment process, it inevitably generates new cases and data. These additions can enrich the knowledge graph and assist relevant personnel in conducting safety assessment work more effectively. However, due to the low efficiency of manual updates and maintenance of the graph, we propose an algorithm called UMA for automatic updates and maintenance.

The algorithm consists of two parts: data cleaning and data importing, as shown in Figure 6. The following will explain each part separately.

As can be seen in Figure 6, the algorithm is divided into two modules: data cleaning and data import.

The data cleaning module combines and compares the SACAS with the cases to be imported, and the output is a number of aligned entity pairs. The two entities in each entity pair are the VALUE from the case and the VALUE' that the VALUE corresponds to in the SACAS. Next, the Valid or Invalid (VORI) gate makes a validity judgment of the value to be input based on the information of the entity pair. The VORI gate calculates the semantic similarity of the two entities in the entity pair. Values above the VORI threshold are output as entity pairs; values below the VORI threshold are individually output to the next step with a value of 0 on the one hand, and on the other hand the original value of the value is output to the missing data error reporting section.

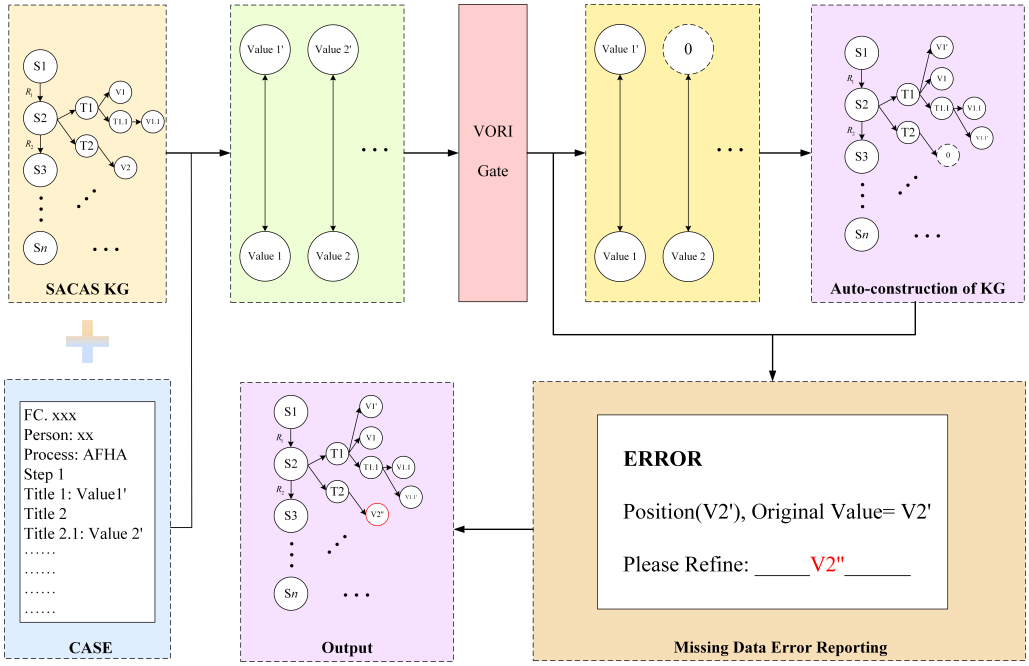

Figure 6. The workflow of the UMA algorithm.

The data import module is divided into two parts: automatic knowledge graph construction and missing data error reporting. The automatic construction part will split the entity pairs output from the VORI gate, and import the values into the knowledge graph according to their position in the knowledge graph to complete the automatic construction of the knowledge graph. Then the missing data error reporting section prints the output from the VORI gate and receives the correction data from the maintainer. After the maintainer determines that the corrected data is correct, the data undergoes the same automatic construction operation as in the previous step. In this way, a case can be accurately and completely entered into the SACAS.

4.2. Data Cleaning

To utilize the newly obtained data for updating the knowledge graph, we must address two questions:

1. **How to acquire new data?**
   With the SACAS already constructed and integrated into the safety assessment process, the new cases and data generated during safety assessment work possess attributes. Two points need to be clarified here: (1) Safety assessment forms must exist within the SACAS; (2) the new cases and data generated during safety assessment work refer to

"blanks". Therefore, we can obtain a set of blanks with attributes and their positions in the form or graph without the need for any specific method.

2. **How to ensure the validity of the new data?**
   Safety assessment work is highly rigorous. Therefore, when determining the validity of new data, we still need to consider two points: (1) The relevance of the data, i.e., whether the new data are from safety assessment work (as there may be invalid data such as 123456, abcdef, etc.); (2) the accuracy and reliability of the data, i.e., whether the assessment of a certain function or component is correct, which requires professional safety assessment personnel to verify, but is not within the scope of this study.

4.2.1. Data Extraction

After completing the safety assessment work, we obtain a set of triples consisting of "title-blank". This set includes the source of the cases (such as personnel information, aircraft model, etc.) as well as attributes of entities and relationships. In other words, each entity and relationship here can correspond to the existing SACAS. At this point, we have completed the data acquisition process. The MLP architecture mentioned in Section 3 makes it more convenient for us to obtain and input data. This step combines and compares the SACAS with the cases to be input. The algorithm obtains the process, function number and the entity pair consisting of both title and value in the case (Table). Then process, function, number and title information is used to find out value' at the same position as value in the SACAS. Finally, the output will be the entity pair of value and value' (which has the relationship of value or rather value' with title). In short, here, the case data is transformed into a number of entity pairs for effective data filtering by the VORI gate and automatic construction of the knowledge graph later.

4.2.2. Valid or Invalid Gate

The Valid or Invalid (VORI) gate makes a validity judgment of the value to be input based on the information of the entity pair. The VORI gate calculates the semantic similarity of the two entities in the entity pair. Algorithm 1 describes the specific steps of the VORI gate. The VORI gate receives a number of entity pairs. The entity pair consists of the value in the case, the value' that corresponds to it in the SACAS, and the relationship between value' in the knowledge graph and the corresponding title. The VORI gate first calculates between each entity pair (value and value') the the semantic similarity between each entity pair (value and value'). If the similarity is higher than the VORI threshold, the VORI gate outputs the entity pair; if it is lower than the VORI threshold, the VORI gate inputs the entity pair into the missing data error reporting module and assigns a value of 0 to the value in the entity pair and inputs it into the next step.

When new case data is acquired, the UMA framework will examine each "value" in the case individually. Based on the attributes and relationships of this "value", the framework can accurately identify the corresponding "value2" in the existing SACAS graph. The semantic similarity between them is calculated to determine their relevance, thereby determining the validity of the input.

The "values" in the case have many data types, some of which we cannot determine the validity or correctness by calculating the semantic similarity. For example, the "Impact Classification Levels" are classified as "1" (Catastrophic), "2" (Dangerous), "3" (catastrophic), "2" (dangerous), "3" (major), "4" (minor) and "5" (no impact on safety). The validity of inputs in this category has to be judged by professionals based on the specific content, so we will not discuss it. We will only discuss inputs whose data type is text.

Algorithm 1: VORI gate

Input: entity pair P;
Output: output pair P, output pair P_0, missing entity pair P'
1: **for** each $P = (V, V')$
2: $V = value$
3: $V' = value'$
4: score = calculate similarity(V,V');
5: **if** score >= VORIthreshold **then**
6: output P to auto-construction module;
7: **else**
8: $P' = (V, V')$;
9: output P' to missing data error reporting module;
10: V'= 0;
11: $P_0 = (V, 0)$
12: output P_0 to auto-construction module;
13: **end if**
14: **end for**

Since the values of the entities in SACAS are formatted, we can compute the semantic similarity between two entities based on the Vector Space Model (VSM). VSM has a very wide range of applications for tasks such as information retrieval oriented. Under VSM, the value of each entity is viewed as consisting of a number of mutually independent words, so the value can instead be represented as a vector. This simplifies the complex semantic relationships within the values and also makes the similarity of the values computable. Since the values in the case are paradigmatic and relatively structured, here, we use Jaccard similarity to compute the semantic similarity of the entity pairs. We use the N-Gram model to cut the values of each entity. For example, if the value of an entity is $V_0 = \{w_1, w_2, \ldots, w_n\}$, $w_i (i = 1, 2, \ldots, n)$ are the words that make up the value. We cut the V_0 with the 2-gram model and the result we obtain is $(w_1, w_2), (w_2, w_3), \ldots, (w_{n-1}, w_n)$. Based on this, we can calculate the semantic similarity of the entity pairs. The calculation formula is as follows:

$$similarity_n(V, V') = \frac{|V \cap V'|_n}{|V \cup V'|_n}, \quad (1)$$

where n is the N in the N-Gram model, i.e., the length of the cut segments. $V \cap V'$ is the number of the same segments in the cut segments of V and V', and $V \cup V'$ is the total number of all different cut segments of V and V'.

The entity pairs obtained from the combination of the case and the SACAS atlas are passed through the VORI gate, which will perform similarity calculations on the entity pairs, resulting in two directions of output: (1) valid data will be passed through the VORI gate in the form of entity pairs and outputted to the atlas building module; and (2) invalid data will not be passed through the VORI gate. The VORI gate will convert the value to 0 and output it in combination with the original entity pairs to the building module. At the same time, the VORI gate will output the invalid value as another output to the error reporting module. After further review by the safety assessment staff, the data for the new case is determined to be valid. At this point, data cleansing for the new case is complete.

4.3. Data Import

4.3.1. Automatic Construction of Knowledge Graph

When we have completed the cleaning of our input case data, we have obtained a set of data that has been filtered and organised. This means that we have eliminated duplicate entries, misinformation, and inconsistencies from the data, allowing us to ensure the quality of our data. The data is validated and can be confidently processed and analysed

subsequently. Once we have completed data cleansing, the next critical step is to transform this validated case data into the form of a knowledge graph. A knowledge graph is a data representation method that organizes entities, relationships and attributes to form a network structure, which can help us better understand the associations and connections between data.

By characterizing and embedding case data into a knowledge graph, we can better organize and manage this data, and discover the laws and patterns hidden behind it. The establishment of a knowledge graph can not only help us better understand the data, but also provide a richer and deeper information base for subsequent data analysis, mining and application. In addition, the establishment of knowledge graph also helps to improve our original database. By transforming the case data into the form of knowledge mapping, we can integrate it with our existing database, thus enriching and improving the content of the database. This will make our database more comprehensive, accurate, and of more applied and practical value.

Transforming case data into a knowledge graph is an important process, which can not only help us better understand and manage the data, but also provide a more powerful and flexible basis for our subsequent data analysis and application. Through this process, we can make better use of our data resources to provide more powerful support for business decision-making and innovation. In order to transform case data into a knowledge graph, we need to capture the entity, relationship and attribute information in the data with the help of algorithms. Entities are the data in the case, such as failure state descriptions, failure impacts, etc. Relationships are the connections between entities, describing their interactions or associations, and attributes are the information describing the characteristics or states of the entities, such as the personal information of the assessor, the type of assessment process, and the object of the assessment. Algorithm 2 describes how the cleaned data is imported into the existing SACAS.

Algorithm 2: Automatic construction of knowledge graph

Input: entity pair P;
Output: triplet table T
1: initial $T = \emptyset$;
2: **for** $P = (V, V')$
3: find the p and e of V;
4: combine p, e with V';
5: **if** a is connected to p by e_a **then**
6: combine a, e_a with V';
7: output (p, e, V') and (a, e_a, V') to T;
8: **else**
9: output (p, e, V') to T;
10: **end if**
11: **end for**

4.3.2. Missing Data Error Reporting

In the process of data cleansing, we often encounter data that does not pass validation, does not meet specifications, or has anomalies. These data may be due to input errors, system issues, or other reasons. In some cases, we receive data with a value of 0 from the auto-build module and a corresponding raw value from the VORI gate. In order to improve the completeness of the input cases and the referentiality of the database, our module needs to handle these cases and generate the corresponding textual errors so that the user can detect and correct them in time.

Firstly, we consider how to handle the case of data with a value of 0 from the auto-build module. This could mean that some data was not recognized or populated correctly during the auto-build process, or it could be due to missing data or other issues. For such

cases, the missing data error reporting module needs to detect these 0 values and pass them to the user along with the corresponding original values.

Secondly, for raw values from VORI gates, we need to make it clear that these data are not valid. VORI gates are used to validate and review data, and the values that they provide to this module are validated and confirmed to be considered invalid. However, sometimes, even if the data comes from the VORI gate, this part of the data may actually be valid but incorrectly determined by the VORI gate to be invalid due to its structural flexibility and versatility. Therefore, this module needs to be able to receive these raw values and pass them on to the user along with another value on the entity pair (the reference value) if required.

Once this detects the receipt of the input of the 0 value and the corresponding raw value, the next step is to generate the appropriate text error messages. These error messages should clearly indicate which data is in question, as well as the possible causes and solutions. Error messages can include, but are not limited to, the following:

1. The original data value.
2. The location in the SACAS of another value (reference value) on the entity pair.
3. The possible reasons why this may have occurred, e.g., input error, system failure, etc.
4. A suggested solution, such as re-entering the data, checking system settings, etc.

By generating such textual error reports, we can help users to identify and solve data problems in time, thus improving the integrity of the input cases and the reference of the database. At the same time, generating error reports also helps to improve the process of data cleansing and processing, and reduce the occurrence of similar problems.

5. Experimental Results

In this section, we will evaluate the proposed SACAS and the UMA algorithm to validate their effectiveness. The experiments use several AFHA cases from a specific aircraft model (due to trade secrets, specific model information will not be disclosed) as the dataset. The dataset is divided into a training set and a testing set in a 7:3 ratio. The training set will be used for constructing SACAS, while the testing set, after being processed for obfuscation, will be utilized to test the UMA algorithm.

All experiments are conducted on a PC with Intel i5-11260H processor running at 2.6 GHz, 16 GB RAM, and NVIDIA GeForce RTX 3050. Neo4j AuraDB was used as the knowledge graph construction tool. All evaluation and testing programs were compiled using Python 3.10.

5.1. Datasets

The beginning of Section 3 explains the scope of the dataset for this experiment. The following section specifically describes the datasets we used to construct SACAS. The assessment table of AFHA (see Figure 3) is the main part of the datasets for this experiment. There are a number of attributes (assessment objects, information of safety assessors, etc.) that are attached to each assessment case. Figure 7 shows the original datasets for constructing SACAS.

In Figure 7, we can see some values with "NA" that actually have no meaning. Due to the particularity of some functions or failure conditions in the task, there are some objectives the assessors do not have to design. As a result, there are these empty values. The documents we obtain from the case cannot be used directly as a dataset to construct SACAS. To build a knowledge graph, the dataset should consist of a large number of triples. The triples can be "entity1-relationship1-attribute1" or "entity1-relationship2-entity2". Part of the processed datasets are shown in Table 2.

With this format of datasets, we can easily construct SACAS in few steps. Then, the dataset is stored in ".csv" file format, and this experiment can use Python to transform the dataset into a knowledge graph.

A.F_3.2.3_.FC._5.1_ _Unannounced partial loss of aircraft ground deceleration capability_						
Function Details	Limitations of the Function		NA			
	Novelty, Specificity and Complexity		NA			
Failure Conditions	No.	Failure Condition Title	Failure Condition Description		Phase	Remarks
	1	Unannounced partial loss of aircraft ground deceleration capability	Unannounced partial loss of aircraft ground deceleration capability.		taxiing	NA
Impact of failure	Impact on Aircraft		Deceleration ability is reduced. The function is reduced during sliding.			
	Impact on Crew		The crew may not be aware of the condition of the aircraft before attempting to slow down. The crew may not be able to completely stop the aircraft, resulting in collisions or runway overruns during low-speed taxiing. To avoid these situations, the workload of the crew will increase significantly.			
	Impact on Passenger		When a collision occurs, unrestrained crew members may be injured.			
	Impact Classification Levels		III. Major			
	Impact on operations		NA			
	Operational Impact Class		II. Operation delay above 15 minutes			
Safety Objective	Safety Objective		__1.0E-5__ per flight hour			
	FDAL Requirements		NA			
	Objective Design		__1__E__-5__			
	FDAL Objective Design		A			
	Validation Methods		FTA	quantitative	qualitative	
				√	√	
			FMEA	quantitative	qualitative	
				√	√	
			CCA	Yes	No	
				\	\	

Figure 7. Original datasets for constructing SACAS.

Table 2. Part of the processed datasets.

Triples of SACAS
AFHA, Process_Function1, Provides aerodynamic performance
AFHA, Process_Function1, Provide a survival environment
AFHA, Process_Function1, Control aircraft trajectory
. . .
Provide a survival environment, Function1_2, Provide breathable atmosphere
Provide breathable atmosphere, Function2_3, Prevent toxic gases
Provide breathable atmosphere, Function2_3, Provide oxygen
. . .
Achieve ground deceleration, Function_Table, A.F.3.2.3.FC.1.1
Achieve ground deceleration, Function_Table, A.F.3.2.3.FC.1.2
Achieve ground deceleration, Function_Table, A.F.3.2.3.FC.1.3
Achieve ground deceleration, Function_Table, A.F.3.2.3.FC.2
Achieve ground deceleration, Function_Table, A.F.3.2.3.FC.3.1
Achieve ground deceleration, Function_Table, A.F.3.2.3.FC.3.2
. . .
A.F.3.2.3.FC.1.1, Step1, Function Details
Function Details, Step1_1, Limitations of the function
Limitations of the function, Step1_2, Novelty, Specificity and Complexity
Novelty, Specificity and Complexity, Step2, Failure Conditions
. . .

5.2. Evaluation Indicators

As introduced in Section 2, we would hardly obtain any instructive conclusions if the proposed method was applied in other domains, and vice versa. To quantitatively evaluate the performance for our task, the evaluation indicators should meet but are not limited to the following aspects:

1. For SACAS
   - **Knowledge Completeness**: Evaluate whether the SACAS contains all the key information required for the task. This includes information pertaining to various system components, technical specifications, safety standards, risk analyses, etc.
   - **Data Accuracy**: Evaluate that the information in the SACAS is accurate, including access to information from reliable data sources, timeliness of data updates, etc.
   - **Relevance and Linkage**: Evaluate the relevance and linkage between different data entities in the knowledge graph to ensure coherence and completeness of information, which helps system users to better understand the key relationships in the safety assessment process.
   - **Usability and scalability**: Evaluate the usability and scalability of the SACAS, including aspects such as the ability to adapt to different user requirements, and the ability to quickly integrate new information.
2. For the UMA algorithm
   - **Rationality**: Evaluate the rationality of the UMA algorithm in the process of safety assessment on civil airborne system data integration, including aspects such as the algorithm's computational efficiency, accuracy and scalability.
   - **Adaptability and flexibility**: Evaluate the adaptability and flexibility of the UMA algorithm in different data integration scenarios, including the ability to handle different data types, data formats and data sources.

In the evaluation process, a combination of quantitative and qualitative methods are used to make a comprehensive assessment using actual data and user feedback. The reviews of experts are combined to ensure the objectivity and accuracy of the assessment results.

5.2.1. Evaluation of SACAS

The primary function of constructing SACAS is to integrate safety assessment data. The performance of the knowledge graph lies in its relevance to the intended purpose. Evaluating the quality of SACAS involves assessing whether it: (1) maximizes the correct utilization of safety assessment data; (2) contains accurate and reliable knowledge and semantic information; (3) features an architecture that is both streamlined and maintainable. In the experiment, we evaluate whether the knowledge graph abides by the following aspects:

1. **Accuracy** (ACC_{KG}): it is the degree to which relevant knowledge in the domain is correctly represented. Accuracy is generally divided into syntactic accuracy, semantic accuracy and timeliness. Due to the specificity of SACAS and the UMA algorithm's filtering and screening of input data, we ignore the syntactic accuracy and timeliness. In our experiments, we designed a quantitative evaluation indicator ACC_{KG} for semantic accuracy, as shown in Equation (2).

$$ACC_{KG} = \frac{S_r(valid)}{S_r}, \tag{2}$$

   where S_r is the total number of paths contained in the knowledge graph and $S_r(valid)$ is the total number of valid paths of S_r. A critical task of this study is the construction of a multi-level, path-based knowledge graph. The role of this metric is to evaluate the effectiveness of the path information within the knowledge graph.
2. **Completeness** (Com_l): completeness refers to the degree of concentration of relevant information. It includes schema completeness, attribute completeness, overall

completeness and linkability. In general, for a fair comparison, either a knowledge graph is compared with knowledge graphs in the same domain; or the data of the knowledge graph is automatically obtained from the dataset, in which the recall of the extraction method judges the completeness. The first three types of completeness evaluate whether automatic methods (deep learning, etc.) have extracted information from the dataset completely. SACAS only uses the automatic method to self-update. So we calculate linkability as a way to evaluate the completeness of the knowledge graph. In our experiments, we designed a quantitative evaluation indicator Com_l for linkability, as shown in Equation (3).

$$Com_l = \frac{N_e}{N_p}, \tag{3}$$

where N_e and N_p are numbers of edges and nodes in a knowledge graph, respectively. The larger the Com_l is, the more linkable the knowledge graph is and the better its integrity. This demonstrates the connectivity within the knowledge graph. As new data are imported, the internal connections of the knowledge graph will inevitably increase, thereby raising the value of this metric. It serves not only to measure the construction quality of the knowledge graph but also to assess the practical effectiveness of the UMA algorithm.

3. **Consistency** (Con_{KG}): in ontology modelling, we set some constraint rules between entities and relationships. For example, a secondary function *reduce aircraft kinetic energy* can only correspond to a primary function *control aircraft kinetic energy*. If this constraint is violated, some incorrect nodes and edges may appear in the knowledge graph and lead to errors in the application process. For this reason, we need to quantize conflicts for each constraint as Equation (4).

$$Con_{KG} = \frac{1}{n}\sum_{i=1}^{n}\frac{R_i(conflict)}{R_i}, \tag{4}$$

where R_i represents the number of groups that should adhere to a particular constraint, $R_i(conflict)$ denotes the number of groups that have violated this constraint, and n is the total number of constraints. Here, a group refers to collections of items such as node-to-node, node-to-edge, or node-to-attribute that should follow certain constraints. For instance, the node *Failure State A* should be connected to the node *Level3Function B* and the relationship between them should be that *Level3Function B* points to *Failure State A*. We establish a constraint: **The *Level3Function* node, apart from connecting to the *Level2Function* node, should only connect to the *Failure State* node, and the relationship should be directed from the *Level3Function* to the *Failure State*.** In this case, the *Level3Function* node, the *Failure State* node, and their relationship should adhere to this rule, forming a group.

4. **Simplicity** (Sim_{KG}): it indicates whether the knowledge graph can correctly characterize the relevant content while avoiding information overload, i.e., the redundancy of the knowledge. Its essence is the presence of duplicate nodes within the knowledge graph (whose duplication is not necessary). Equation (5) allows for the calculation of the simplicity of the knowledge graph:

$$Sim_{KG} = 1 - \frac{1}{n}\sum_{i=1}^{n}\frac{p_i - 1}{p_i}, \tag{5}$$

where p_i is the number of each kind of node and n is number of all different nodes within the knowledge graph.

5.2.2. Evaluation of UMA Algorithm

We evaluate the proposed algorithm both qualitatively and quantitatively. Table 3 shows the qualitative evaluation table of the UMA algorithm.

Table 3. The qualitative evaluation table of the UMA algorithm.

Evaluation Indicators	Evaluation Content	Remarks
Correctness	Input	(1)
Correctness	Desired output	(2)
Correctness	Derived output	(3)
Readability	Presence of undefined variables	(4)
Readability	Conformity to identifier naming conventions	(5)
Readability	Presence of duplicate or missing statements	(6)
Robustness	Output with unintended inputs	(7)

Blanks (1) to (7) are what we need to fill in when evaluating the UMA with this table.

The qualitative evaluation includes the evaluation of the correctness, readability and robustness of the algorithm: (1) Correctness refers to the ability of the algorithm to fulfill the requirements for writing the algorithm. A correct algorithm meets the requirements of the specific problem, the programme runs correctly, has no syntax errors, and can pass typical software tests to meet the expected requirement specifications. (2) Readability refers to the ease with which the algorithm can be read. Algorithms with high readability should follow the naming rules of identifiers, be concise, easy to understand, and have an appropriate amount of comment statements, which are easy to read for yourself and others, and easy to debug and modify in the later stages. (3) Robustness refers to the algorithm's ability to react to various inputs and its ability to handle them, which is also known as fault-tolerance. An algorithm with high robustness should be able to perform its task correctly and output clear information in a variety of situations.

We perform quantitative evaluation in terms of time complexity and space complexity. For time complexity, it is often necessary to analyze the number of essential operations in the algorithm and the frequency with which these operations are performed. The order of magnitude of time complexity can be determined by calculating the number of times the key operations in the algorithm are executed. In general, time complexity is calculated by using the worst-case algorithm, i.e., the measure of time required for the execution of the algorithm in the most unfavorable case. The quantification of time complexity is denoted by O, where n denotes the size of the input data. Equation (6) allows one to compute the worst-case time complexity of the algorithm:

$$T(n) = O(f(n)), \tag{6}$$

where $T(n)$ denotes the time complexity of the algorithm and $f(n)$ denotes the function of input scale n.

For space complexity, the additional memory space required by the algorithm during execution needs to be analysed. Similarly, the worst-case space complexity of an algorithm is used to evaluate the memory usage of the algorithm. Equation (7) allows for the calculation of the worst-case space complexity of the algorithm:

$$S(n) = O(f(n)), \tag{7}$$

where $S(n)$ denotes the space complexity of the algorithm and $f(n)$ denotes the function of input scale n.

5.3. Results and Analysis

We evaluate the quality of the SACAS and the UMA algorithm based on the evaluation indicators presented in Section 5.1.

5.3.1. Results of the SACAS Construction

In our experiments, we adopt the MLP framework based on safety assessment data to construct the SACAS, which is an important result with multi-level and path-based information representation. Table 4 shows the scale of SACAS. Figure 8 shows part of the construction results of the SACAS, while Figure 9 presents its path-finding information representation.

Table 4. The scale of SACAS.

label	Lv1Func	Lv2Func	Lv3Func	Process	Step	TableNum	Value	Total
number	8	12	20	1	125	4	110	280
edge	Func1_2	Func2_3	FuncTab	P_Func1	Stepm_n	Valuep_q		Total
number	12	20	20	8	596	110		766

In Table 4, Lv1Func, Lv2Func, Lv3Func refer to Level1Function, Level2Function and Level3Function, respectively. TableNum refers to TableNumber. Func1_2 and Func2_3 refer to Function1_2 and Function 2_3, respectively. FuncTab refers to FunctionTable. Stepm_n refers to all step relations of the ontology modeling results in Table 1. Valuep_q refers to all the values.

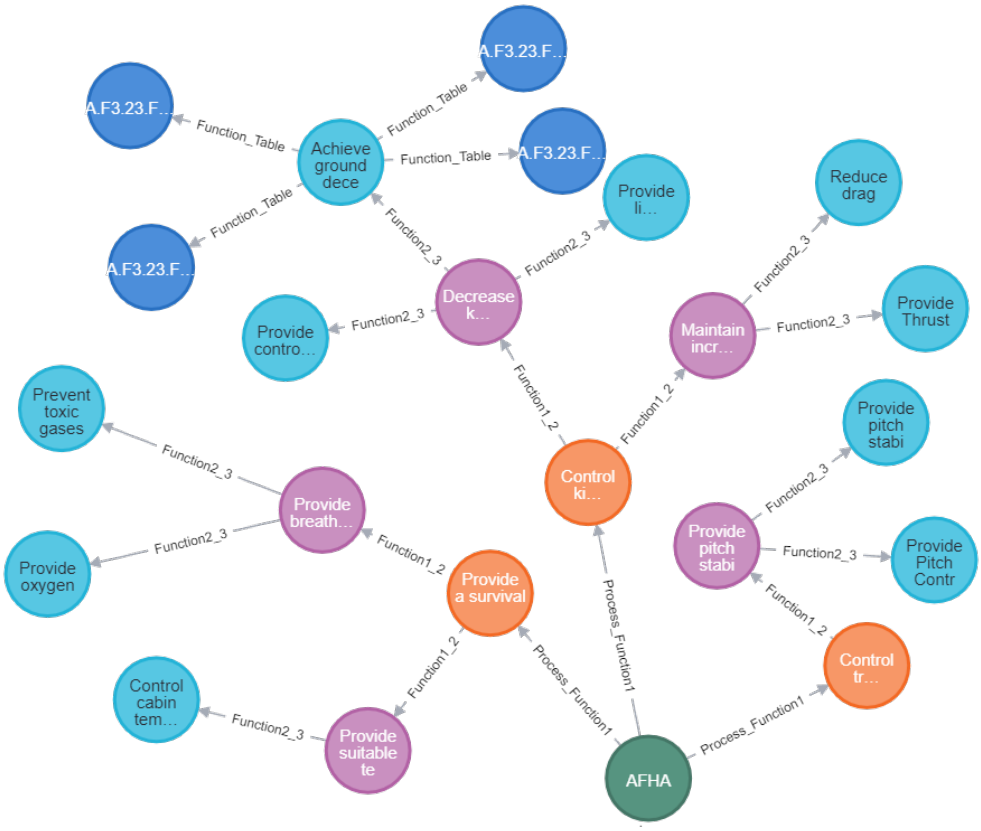

Figure 8. Part of the construction results of the SACAS.

We can see that the MLP architecture facilitates the achievement of multi-level and path-finding properties. This will visualize semantic information and make the structure of the data more stereoscopic. Hence, a knowledge graph will be more flexible and convenient for subsequent application work.

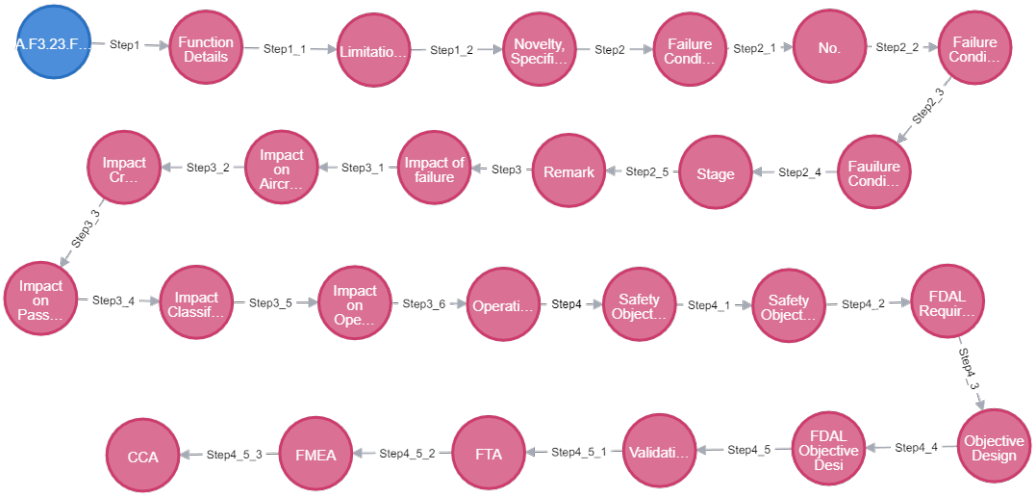

Figure 9. The path-finding information representation of SACAS.

According to Table 4, the knowledge graph comprises 280 nodes and 766 edges. All have been verified as correct. Using Equation (2), the ACC_{KG} is calculated to be 100%. According to Equation (3), the Com_l is calculated to be 273.57%. As indicated in Table 4, there are 58 constraints that must be adhered to during the construction of the graph. Upon verification, the total number of groups that violated these constraints is 94. Using Equation (4), the Con_{KG} is calculated to be 0.03%. Upon further verification, there are 180 different kinds of nodes, of which 25 kinds consist of 5 identical nodes each. Using Equation (5), the Sim_{KG} is calculated to be 92.23%.

Table 5 shows the quality evaluation results of the knowledge graph.

Table 5. The quality evaluation results of the knowledge graph.

Evaluation Indicators	Value (%)
ACC_{KG}	100
Com_l	273.57
Con_{KG}	0.03
Sim_{KG}	92.23

The results show that the proposed SACAS is able to accurately and fully utilize the safety assessment data. From the results, the construction process of this knowledge graph strictly follows many constraints set during ontology modelling. The architecture is concise with no repetitive parts. We further analysed the strengths and weaknesses of the construction process of the SACAS. The results given in Table 5 show that the application of the MLP framework allows the multi-layered and path-finding properties of the knowledge graph to be fully represented; however, there are some challenges, such as the possible complexity in dealing with complex relationships and connections between entities. Therefore, we can further optimize the algorithms and models to improve the representation capability and application effect of the knowledge graph.

The results demonstrate that the proposed method achieved satisfactory results and provides a solid foundation for subsequent safety assessment and application. With the in-depth research and continuous optimisation of knowledge graph quality assessment, we believe that SACAS will play an important role in related fields and provide more reliable support and guidance for solving practical problems.

5.3.2. UMA Performance

In the practical application, we tested the UMA algorithm and observed its performance in different scenarios. By collecting and analyzing the experimental data, we were able to obtain a comprehensive picture of the performance of the UMA algorithm in differ-

ent tasks, so as to assess its effectiveness and practicality. Table 6 shows the scale of new SACAS and comparison with SACAS in Section 5.3.1. Table 7 shows the quality assessment of new data under the UMA algorithm after importing it into the SACAS.

Table 6. The scale of new SACAS and comparison.

	Nodes	Edges
old SACAS	280	766
imported part	21	215
new SACAS	301	981

Table 7. The quality evaluation results of the new knowledge graph.

Evaluation Indicators	Value (%)
ACC_{KG}	100
Com_l	325.91
Con_{KG}	0.05
Sim_{KG}	91.61

We can see that the UMA algorithm achieves automatic update and maintenance of the SACAS, and through the two modules of data cleansing and data import, it realizes the validity judgment and automatic construction of the new case data, so as to provide efficient and accurate support for the safety assessment work. The automation characteristics of the UMA algorithm can greatly improve the efficiency of data processing, and at the same time, ensure the accuracy and completeness of the data, providing a solid foundation for the safety assessment work.

In the qualitative evaluation, we mainly examined the correctness, readability and robustness of the UMA algorithm. We analyzed whether the algorithm met expectations by looking at its outputs and assessing its adaptability and stability to different inputs. Table 8 shows the results of the qualitative evaluation of the UMA algorithm according to the evaluation table in Section 5.3.2.

Table 8. The results of the qualitative evaluation of the UMA algorithm.

Evaluation Content	Remarks
Input	entity pair P
Desired output	triplet table T
Derived output	triplet table T
Presence of undefined variables	none
Conformity to identifier naming conventions	no conflict
Presence of duplicate or missing statements	none
Output with unintended inputs	reporting error

Table 8 shows that the output of the UMA algorithm is in line with expectations and is able to correctly fulfill the purpose for which we wrote the algorithm. The algorithm is highly robust and is able to operate robustly in complex and changing environments, eliminating most invalid or interfering inputs as a way to complete the task. At the same time, the algorithm is concise and readable. The readability of an algorithm is crucial for engineering practice, and it directly affects the maintenance and further optimisation of the algorithm. Our evaluation shows that the content of the UMA algorithm is concise and clear, easy to understand and modify, which provides a good basis for future improvement work.

For quantitative evaluation, we focus on the time complexity and space complexity of the UMA algorithm. By testing and counting the running time and memory occupation of the algorithm on datasets of different sizes, we are able to evaluate the efficiency and resource utilization of the algorithm. After calculation, Table 9 shows the time and space complexity of the UMA algorithm (including the two algorithms).

Table 9. The time and space complexity of the UMA algorithm.

Algorithm	Time Complexity	Space Complexity
VORI Gate	$O(n)$	$O(1)$
Auto-construction	$O(n^2)$	$O(n)$
UMA(Total)	$O(n^2)$	$O(n)$

Table 9 shows that UMA algorithm shows excellent performance in terms of time complexity and space complexity. Assuming the input size is denoted as n, where n represents the number of nodes or relationships in the knowledge graph. The analysis of time complexity is based on the assumption that each node in the algorithm is accessed at least once, since the VORI gate performs a comparison (constant operations) for each input node, resulting in a worst-case time complexity of $O(n)$. The auto-construction algorithm, however, needs to compare the position information of the input nodes with the existing knowledge graph, involving a traversal operation, hence the time complexity is $O(n^2)$. The analysis of space complexity considers the temporary storage of nodes and their relationships during the algorithm's execution, which compared to the time complexity (bilateral data) is a measure of unilateral data. In other words, the algorithm only needs to store data from one side; thus, the space complexities of the VORI gate and the auto-construction algorithm are $O(1)$ and $O(n)$, respectively. These analyses assume that the basic operations of the data structure, such as adding or removing nodes, incur constant time and space costs. Its execution efficiency is high and resource utilization is reasonable, which can meet the requirements of practical applications.

The UMA algorithm shows excellent performance and reliability in practical applications. By comprehensively evaluating it, we gain a deeper understanding of the strengths and weaknesses of the algorithm and provide guidance and suggestions for its future improvement and optimisation. The successful application of the UMA algorithm not only provides an effective solution for the updating and maintenance of SACASs, but also provides a useful reference for the research and practical work in related fields.

6. Conclusions

With the continuous development of new aircraft, the lack of qualified safety assessors has seriously limited the market application of these new products. The current safety assessment on civil airborne system requires sufficient assessors with a wealth of experience and a wide range of knowledge, which is a wide gap from the current state of affairs in terms of the quality and quantity of practitioners.

To address these challenges, a comprehensive technological solution is required, comprising the following aspects: an information integration technology tailored for the safety assessment process, a method to address the issue of data sparsity in safety assessment cases, a recommendation system based on sparse data from safety assessment, and a self-updating framework for the safety assessment database. This technology, by integrating and updating safety assessment data, significantly enhances the efficiency and accuracy of the task, mitigating the impacts of experience deficiency or data sparsity on the assessment.

In this paper, we proposed a knowledge graph-based information integration technique for safety assessment on civil airborne system. Firstly, we constructed a multi-level path-based knowledge graph called SACAS for our task with the proposed MLP architecture. We designed a self-updating algorithm for knowledge graphs called UMA, which can effectively incorporate new knowledge from the usage process into the knowledge graph. In our experiments, we defined a set of evaluation metrics to evaluate the proposed methods. The results have shown that SACAS can accurately and adequately utilize safety assessment data. Its architecture is concise and has no repetitive parts. The application of MLP architecture enables the multi-layered and path-finding characteristics of the knowledge graph. Meanwhile, the proposed UMA algorithm has high correctness, readability and robustness, and has shown excellent performance in time complexity and space complexity.

Our work utilized the correlation of a large amount of case data, thus making the assessment work simpler, which in turn makes it less demanding on the safety assessors' experience. Moreover, assessors can learn the relevant knowledge quickly through the proposed method, thus increasing the efficiency of the whole safety assessment work.

The focus of this study is on the implementation of information integration technology and the construction of a database self-updating framework, which are the core components of this technology. Future research will follow this technological roadway, addressing the data sparsity issue in safety assessments and developing a recommendation system based on the information integration technology to further enhance the decision-making capabilities of inexperienced personnel. The potential applications of this technology extend beyond traditional commercial and cargo aviation to emerging low-altitude domains, effectively addressing the shortage of safety assessors in these areas and promoting the development of the low-altitude economy.

While the current study provides valuable insights, it is important to acknowledge the limitations in conducting quantitative comparisons with other methods within the scope of this paper. Due to the specific application area, the performance of the method described herein has not been tested in other scenarios. To enhance the efficiency of safety assessments for civil aircraft, further in-depth research will focus on the management and interaction of data in this domain.

Author Contributions: Conceptualization, X.C. and Q.Z.; methodology, X.C. and Q.Z.; software, Q.Z.; validation, X.C., L.D. and Q.Z.; formal analysis, Q.Z.; investigation, J.B.; resources, Q.Z.; data curation, Q.Z.; writing—original draft preparation, Q.Z.; writing—review and editing, X.C. and Q.Z.; visualization, Q.Z.; supervision, J.B. and L.D.; project administration, J.B.; funding acquisition, X.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program of China Grant 2023YFB4302902 and the Fundamental Research Funds for the Central Universities under Grant 3122022QD07.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

eVTOL	Electric Vertical Take-off and Landing
SACAS	Safety Assessment on Civil Airborne System
AFHA	Aircraft Functional Hazard Analysis
SFHA	System Functional Hazard Analysis
FHA	Functional Hazard Analysis
PSSA	Preliminary System Safety Assessment
SSA	System Safety Assessment
MLP	Multi-Level Path-based
UMA	Updating and Maintaining Automatically
VORI	Valid or Invalid
MBSA	Model-Based Safety Assessment

References

1. Federal Aviation Administration. Advisory Circular 20-174: Development of Civil Aircraft and Systems. 2011. Available online: https://www.faa.gov/documentLibrary/media/Advisory_Circular/AC_20-174.pdf (accessed on 30 September 2011).
2. Federal Aviation Administration. Advisory Circular 23.1309-1E: System Safety Analysis and Assessment for Part 23 Airplanes. 2011. Available online: https://www.faa.gov/documentLibrary/media/Advisory_Circular/AC%2023.1309-1E.pdf (accessed on 17 November 2011).

3. S-18 Aircraft and Systems Development and Safety Assessment Committee. *Aerospace Recomended Practice 4761A: Guidelines for Conducting the Safety Assessment Process on Civil Aircraft, Systems, and Equipment*; SAE International: Warrendale, PA, USA, 2023. [CrossRef]
4. Zhang, G.; Hu, G. Advances in Civil Aircraft System Security and Application Strategies. In Proceedings of the 2012 9th Yangtze River Delta Science and Technology Forum, Lianyungang, China, 17–19 November 2012; pp. 169–173.
5. Huang, M.; Jie, Y.; Song, Z.; Zhang, P. Research on Key Technologies for Airworthiness Review of Safety Assessment on Civil Aircraft System. *Aeronaut. Stand. Qual.* **2023**, *5*, 18–22. [CrossRef]
6. Gan, C.; Ding, S.; Qiu, T.; Liu, P.; Ma, Q. Model-based safety analysis with time resolution (MBSA-TR) method for complex aerothermal–mechanical systems of aero-engines. *Reliab. Eng. Syst. Saf.* **2024**, *243*, 109864. [CrossRef]
7. Li, G.; Teng, Y.; Ding, S. Complex physical-model based dynamic system safety analysis of Aviation Piston Engine considering hybrid uncertainty of fault. *Eng. Fail. Anal.* **2023**, *152*, 107515. [CrossRef]
8. Sharvia, S.; Papadopoulos, Y. Integrating model checking with HiP-HOPS in model-based safety analysis. *Reliab. Eng. Syst. Saf.* **2015**, *135*, 64–80. [CrossRef]
9. Li, Y.; Gong, Q.; Su, D. Model-based System Safety Assessment of Aircraft Power Plant. *Procedia Eng.* **2014**, *80*, 85–92. [CrossRef]
10. Sun, M.; Gautham, S.; Ge, Q.; Elks, C.; Fleming, C. Defining and characterizing model-based safety assessment: A review. *Saf. Sci.* **2024**, *172*, 106425. [CrossRef]
11. S-18 Aircraft and Systems Development and Safety Assessment Committee. *Aerospace Recomended Practice 4761: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment*; SAE International: Warrendale, PA, USA, 1996. [CrossRef]
12. Xu, D.; Zhou, J.; Xu, T.; Xia, Y.; Liu, J.; Chen, E.; Dou, D. Multimodal Biological Knowledge Graph Completion via Triple Co-Attention Mechanism. In Proceedings of the 2023 IEEE 39th International Conference on Data Engineering (ICDE), Anaheim, CA, USA, 3–7 April 2023; pp. 3928–3941. [CrossRef]
13. Cai, H.; Zheng, V.W.; Chang, K.C.C. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 1616–1637. [CrossRef]
14. Hu, S.; Zou, L.; Yu, J.X.; Wang, H.; Zhao, D. Answering Natural Language Questions by Subgraph Matching over Knowledge Graphs. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 824–837. [CrossRef]
15. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Yu, P.S. A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 494–514. [CrossRef] [PubMed]
16. Yadav, D.; Para, H.; Selvakumar, P. Unleashing the Power of Large Language Model, Textual Embeddings, and Knowledge Graphs for Advanced Information Retrieval. In Proceedings of the 2023 International Conference on Electrical, Computer and Energy Technologies (ICECET), Cape Town, South Africa, 16–17 November 2023; pp. 1–5. [CrossRef]
17. Nadeem, M.; Zenkert, J.; Weber, C.; Fathi, M.; Hamza, M. Smart UX-design for Rescue Operations Wearable—A Knowledge Graph Informed Visualization Approach for Information Retrieval in Emergency Situations. In Proceedings of the 2023 IEEE International Conference on Electro Information Technology (eIT), Romeoville, IL, USA, 18–20 May 2023; pp. 180–185. [CrossRef]
18. Yan, N.; Li, J.; Chen, J.; Liu, Q.; Li, A.; Wang, K. Implementation of Intelligent Q&A System for Electric Power Knowledge Based on Knowledge Graph. In Proceedings of the 2023 5th International Conference on Electrical Engineering and Control Technologies (CEECT), Chengdu, China, 15–17 December 2023; pp. 605–609. [CrossRef]
19. Li, Z.; Xu, Q.; Zhang, W.; Zhang, T. An Approach and Implementation for Knowledge Graph Construction and Q&A System. In Proceedings of the 2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Taiyuan, China, 3–5 December 2021; pp. 425–429. [CrossRef]
20. Xu, X.; Xie, X.; Yi, J.; Tang, X. Research on Knowledge Graph-Based Q&A for Aero-Engine Fault Diagnosis. In Proceedings of the 2023 5th International Conference on Applied Machine Learning (ICAML), Dalian, China, 21–23 July 2023; pp. 180–185. [CrossRef]
21. Wang, J.; Wang, W.; Meng, F.; Zhou, L.; Guo, S. A Review of Knowledge Reasoning Based on Neural Network. In Proceedings of the 2023 IEEE 12th International Conference on Communication Systems and Network Technologies (CSNT), Bhopal, India, 8–9 April 2023; pp. 580–584. [CrossRef]
22. Tian, L.; Zhou, X.; Wu, Y.P.; Zhou, W.T.; Zhang, J.H.; Zhang, T.S. Knowledge graph and knowledge reasoning: A systematic review. *J. Electron. Sci. Technol.* **2022**, *20*, 100159. [CrossRef]
23. Chen, X.; Jia, S.; Xiang, Y. A review: Knowledge reasoning over knowledge graph. *Expert Syst. Appl.* **2020**, *141*, 112948. [CrossRef]
24. Guo, Q.; Zhuang, F.; Qin, C.; Zhu, H.; Xie, X.; Xiong, H.; He, Q. A Survey on Knowledge Graph-Based Recommender Systems. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 3549–3568. [CrossRef]
25. Liu, J.; Duan, L. A Survey on Knowledge Graph-Based Recommender Systems. In Proceedings of the 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 12–14 March 2021; pp. 2450–2453. [CrossRef]
26. Wei, X.S.; Song, Y.Z.; Aodha, O.M.; Wu, J.; Peng, Y.; Tang, J.; Yang, J.; Belongie, S. Fine-Grained Image Analysis with Deep Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 8927–8948. [CrossRef] [PubMed]
27. Zhu, Z.; Lin, K.; Jain, A.K.; Zhou, J. Transfer Learning in Deep Reinforcement Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 13344–13362. [CrossRef] [PubMed]

28. Ye, M.; Shen, J.; Lin, G.; Xiang, T.; Shao, L.; Hoi, S.C.H. Deep Learning for Person Re-Identification: A Survey and Outlook. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 2872–2893. [CrossRef] [PubMed]
29. Wu, X.; Jiang, T.; Zhu, Y.; Bu, C. Knowledge Graph for China's Genealogy. In Proceedings of the 2020 IEEE International Conference on Knowledge Graph (ICKG), Nanjing, China, 9–11 August 2020; pp. 529–535. [CrossRef]
30. Zha, Z.; Wang, J.; Li, Z.; Zhu, X.; Song, W.; Xiao, Y. M2ConceptBase: A Fine-grained Aligned Multi-modal Conceptual Knowledge Base. *arXiv* **2023**, arXiv:2312.10417. https://doi.org/10.48550/ARXIV.2312.10417.
31. Jin, L.; Chen, J. Self-supervised opinion summarization with multi-modal knowledge graph. *J. Intell. Inf. Syst.* **2023**, *62*, 191–208. [CrossRef]
32. Noy, N.; Mcguinness, D. Ontology Development 101: A Guide to Creating Your First Ontology. *Knowl. Syst. Lab.* **2001**, *32*, 1–8.
33. S-18 Aircraft and Systems Development and Safety Assessment Committee. *Aerospace Recomended Practice 4754B: Guidelines for Development of Civil Aircraft and Systems*; SAE International: Warrendale, PA, USA, 2023. [CrossRef]
34. Xiu, Z. *System Safety Design & Assessment in Civil Aircraft*; Shanghai Jiao Tong University Press: Shanghai, China, 2018.