FT2025
SWEDISH AEROSPACE AT THE CROSSROADS
The 12th Swedish Aerospace Technology Congress
FT2025 in Stockholm
October 14-15, 2025

Improving System Safety in Aviation: Supporting STPA with AI Models

Luiz Eduardo Galvão Martins[1], Ana Estela Antunes da Silva[2], Gabriel Nogueira Pacheco[1], Andrey Toshiro Okamura[2], Niklas Lavesson[3], Tony Gorschek[3]

[1]*Department of Science and Technology, Federal University of São Paulo, São José dos Campos, Brazil*
E-mail: legmartins@unifesp.br, gpacheco@unifesp.br
[2]*Faculty of Technology, State University of Campinas, Limeira, Brazil*
E-mail: aeasilva@unicamp.br, a213119@dac.unicamp.br
[3]*Department of Software Engineering, Blekinge Institute of Technology, Karlskrona, Sweden*
E-mail: niklas.lavesson@bth.se, tony.gorschek@bth.se

Abstract

Background: System safety in aeronautics is critical, as it directly affects aircraft reliability, efficiency, safety, and security. Given the complexity of modern aviation systems and the potential consequences of failures, a structured and proactive safety approach is essential. System-Theoretic Process Analysis (STPA) is a modern hazard analysis method designed to identify and mitigate risks. Unlike traditional methods that focus primarily on component failures, STPA accounts for both failures and unsafe interactions among system elements, including human operators, software, and organizational factors. Problem: Despite its effectiveness, STPA poses challenges in practical application. The process is time-consuming and requires extensive expertise in system safety, control theory, and system dynamics. Analysts must heavily rely on expert judgment to define losses, hazards, safety constraints, and unsafe control actions. Additionally, training in STPA is resource-intensive, making automation an appealing solution to streamline the process. Goal: To address these challenges, we developed two AI-driven pipelines to automate the initial steps of STPA, reducing reliance on expert knowledge and enhancing efficiency. Method: The first pipeline leverages a fine-tuned Llama3.1-8B model to extract losses, hazards, and constraints from ConOps documents. The second pipeline, BERT Error Detection for STPA (BEDS), improves accuracy by classifying, verifying, detecting errors, and suggesting potential corrections for the extracted elements. Results: The first pipeline was trained using 134 ConOps documents paired with corresponding STPA safety analysis elements. The dataset comprised 35 authentic documents from the CORDIS repository and 99 AI-generated examples. The model achieved a mean precision of 79.73%, recall of 81.09%, and an F1-score of 80.22%. For the second pipeline, 1,084 sentences were extracted from values identified during the first step of STPA. Three classifiers were developed: the sentence identifier achieved a mean accuracy of 95.20%, the incorrect sentence detector 88.61%, and the sentence error identifier 83.44%. While the pipelines were designed to work together, they can also be used independently. Conclusion: This study tackles the challenges of applying STPA in aeronautics by introducing two automated pipelines to streamline the initial process steps. The first pipeline, powered by a fine-tuned Llama3.1-8B model, extracts losses, hazards, and constraints from ConOps documents. The second pipeline, BEDS, verifies and corrects these elements with high accuracy. The results demonstrate strong precision and recall scores, highlighting the potential to reduce both the time and expertise required for STPA analysis in complex aviation systems.

Keywords: System Safety, STPA, AI-driven Pipeline, ConOps, Loss, Hazard

1 Introduction

In the context of safety-critical systems, system safety refers to a disciplined approach to identifying, analyzing, and mitigating hazards that could lead to accidents, injuries, or loss of life. System safety is the application of engineering and management principles, criteria, and techniques to achieve acceptable risk levels throughout the life cycle of a system whose failure could result in catastrophic consequences [1, 2].

Due to the complexity of modern aviation systems and the potential consequences of failures, a structured and proactive safety approach is necessary. System-Theoretic Process Analysis (STPA) is a hazard analysis method designed to identify and mitigate risks. Unlike traditional methods that concentrate primarily on component failures, STPA considers both failures and unsafe interactions among system elements, including human operators, software, and organizational factors [3].

Although effective, STPA presents several challenges in its practical application. It is a time-intensive process that requires substantial expertise in system safety, control theory, and system dynamics. Analysts must rely significantly on expert judgment to identify losses, hazards, safety constraints, and unsafe control actions. Furthermore, training in STPA demands considerable resources, which makes automation an attractive solution to streamline the procedure [4, 5, 6].

To address these challenges, we have developed two AI-driven pipelines to automate the first step of STPA, reducing reliance on expert knowledge and enhancing efficiency. The first pipeline leverages a fine-tuned Llama3.1-8B model to extract losses, hazards, and constraints from ConOps documents. The second pipeline, BERT Error Detection for STPA (BEDS), improves accuracy by classifying, verifying, detecting errors, and suggesting potential corrections for the extracted elements. The purpose of this work is to present and analyze the potential of the initial results obtained from the two AI-driven pipelines developed for automating the first step of STPA.

The remainder of this paper is organized as follows: in Section 2 we present the related work found in the literature; Section 3 explains the architecture, the dataset and the fine-tuning performed in the first pipeline; Section 4 explains the architecture, the data processing and the fine-tuning performed in the second pipeline; in Section 5 we discuss the preliminary results obtained with the pipelines; and in Section 6 we present the conclusions and directions for future work.

2 Related Work

There are some works that apply machine learning associated with the STPA process. Acar Celik et al. [7] apply STPA in a pedestrian collision detection component in autonomous cars, which uses machine learning. There was a concern in applying STPA in this component, as standards in automotive vehicles such as ISO 26262 do not explicitly describe how to regulate deep learning algorithms.

The work of Ghosh et al. [8] deals with components of autonomous cars. In this work, the authors apply STPA to identify risks of cyberattacks on software components, and machine learning is applied in algorithms for capturing external information, such as radars, for example. The authors argue that, with the increase of autonomous cars on the road in recent years, there is also an increase in potential attacks on these vehicles. Although there are threat analysis guides for vehicles like the ISO 21434 standard mentioned by the authors, there are still no methods specifically designed for autonomous cars. Therefore, the authors conducted a comparison between the ISO 21434 standard and STPA-Sec, an adaptation of STPA for security. The advantages of applying STPA-Sec, raised by the authors, are the ability to model malicious interactions on the road, such as objects and other users, and to identify critical paths for the vehicle's operation.

There are some works that use Large Language Models (LLMs) to assist in STPA analysis. Diemert and Weber [9] investigate the integration of LLMs in the hazard analysis processes, which the authors call Co-Hazard Analysis (CoHA). In the CoHA method, the authors teach Chat-GPT 3.0 coding to understand the description of the system of interest and provide the model with various queries to identify the unsafe Control Actions of the system described. The LLM's response is then marked in three categories: the response is correct and useful, the response is correct but not useful, and the response is incorrect. The authors then compare the proportion of each marking at different levels of system complexity. The authors state that CoHA can be moderately useful for simpler systems. However, as the complexity of the system increases, the performance of the LLM declines.

Yi Qi et al. [10] examine the application of STPA to Automatic Emergency Brake (AEB) and Electricity Demand Side Management (DSM) systems, utilizing Chat Generative Pre-Trained Transformer (ChatGPT). They investigated the impact of prompt engineering on STPA results. According to the authors, comparative results indicate that using ChatGPT without human intervention may be inadequate due to reliability issues. However, with careful design, it has the potential to outperform human experts. However, authors also identify future challenges, such as concerns regarding the trustworthiness of LLMs and the need for standardization and regulation in this field.

Charalampidou et al. [11] investigate the usefulness of LLMs, such as GPT4 from OpenAI, in applying the STPA hazard analysis in socio-technical systems. They applied the process of STPA to the ROLFER search and rescue drone system. Firstly, the system was discussed in detail with the LLM, defining its purpose, goals, components, and operations. Then, the LLM was talked through the steps of STPA and prompted to generate their specifications. These outputs were compared with those from the human safety team that applied STPA to the same system. The team discovered that LLM can be beneficial in certain aspects, such as loss scenario and safety specification generation and helping the researchers to reach a better understanding of the system, but have drawbacks in others, such as the generation of Unsafe Control

Actions and being used as a verification tool of the STPA results of the human analysts. Authors concluded that by using LLM tools such as ChatGPT-4, the total time needed for a complete execution of an STPA analysis in a complex sociotechnical system can be greatly decreased due to the generation of questions that help with the understanding of the system, and the generation of the causal scenarios and safety specifications. ChatGPT-4 can also be used to check for duplicate or similar safety specifications and UCAs, a rather tedious and time-consuming procedure.

In general, the works described recommend not using LLMs as substitutes for human analysts, but rather as a tool to assist in reducing the time required to complete the analysis. We can understand that the application of machine learning in STPA hazard analysis tasks is a research topic yet to be explored.

3 The First Pipeline: (SHACO)-Llama

The manual execution of STPA, a cornerstone for ensuring safety in complex systems, presents considerable challenges. Particularly in its initial phases, which involve the meticulous analysis of Concept of Operations (ConOps) documents, STPA is recognized as a resource-intensive and time-consuming endeavor. The inherent complexity of modern systems, detailed within extensive ConOps documents that can range from 10 to 200 pages, demands careful examination by human analysts, introducing potential for subjectivity, inconsistencies, and limitations in scalability. To address these issues, the first pipeline, termed STPA Hazard Analysis from ConOps (SHACO)-Llama, was developed. This innovative approach leverages advanced Natural Language Processing (NLP) techniques to automate the extraction of fundamental STPA elements (specifically losses, system-level hazards, and safety constraints) directly from ConOps documents. The primary objective of SHACO-Llama is to streamline the initial STPA process, thereby reducing the reliance on extensive expert judgment during these early stages and enhancing the overall efficiency and consistency of safety analysis.

The development of this pipeline targets a critical bottleneck in the STPA workflow: the first step involving the identification of losses, hazards, and constraints. Automating this step can yield cascading benefits throughout the safety analysis lifecycle, potentially accelerating the entire safety assurance process. This automation signifies a broader shift towards employing sophisticated NLP and Large Language Model (LLM) capabilities for the analysis of complex engineering documentation. Such an approach moves beyond traditional rule-based systems or simpler machine learning models, indicating a maturation of NLP to a point where it can be effectively applied to the extraction of safety-critical information from technical texts, albeit with necessary human oversight.

3.1 Architecture of the SHACO-Llama Pipeline

The SHACO-Llama pipeline is architecturally designed to ingest ConOps documents as input and subsequently output the initial set of STPA elements: losses, system-level hazards, and safety constraints. At the core of this pipeline lies a fine-tuned Llama3.1-8B model, an LLM developed by Meta [12]. The selection of Llama3.1-8B was predicated on its favorable balance of high performance and computational efficiency for specialized tasks, coupled with its robust instruction-following capabilities and open-source accessibility, which are crucial for research and development. The 8B parameter variant, specifically, offers a practical equilibrium between analytical power and the resources required for fine-tuning and deployment.

ConOps documents, which serve as the primary input, are foundational artifacts in system development. These documents typically adhere to established standards, such as IEEE Std 1362-1998 [13] and IEEE/ISO/IEC 29148-2011 [14], and provide detailed descriptions of system objectives, operational scenarios, and stakeholder needs from which safety elements must be inferred. The outputs generated by the pipeline (losses, system-level hazards, and safety constraints) are fundamental to the STPA methodology as comprehensively detailed by Leveson and Thomas [3]. The pipeline's operation can be characterized as an abstractive summarization task; it does not merely extract verbatim text but rather generates these STPA elements based on its contextual understanding of the input ConOps document. This capability is particularly significant because critical safety elements like losses and hazards are often implicitly described rather than explicitly stated in ConOps documents, requiring a level of inference akin to that performed by human safety analysts. The model’s proficiency in instruction-following is vital, as the fine-tuning process essentially trains it to adhere to the complex "instruction" of transforming a narrative ConOps document into a structured set of STPA elements.

3.2 Dataset Curation: The SHACO Corpus

A pivotal component in the development of the SHACO-Llama model was the creation of a specialized dataset, the STPA Hazard Analysis from ConOps (SHACO) corpus. This dataset was meticulously curated due to the absence of existing, suitable datasets for fine-tuning an LLM for the specific task of STPA element extraction from ConOps documents. The SHACO dataset comprises 134 ConOps documents, each meticulously paired with its corresponding STPA elements (losses, hazards, and safety constraints).

A hybrid strategy was employed for the compilation of this dataset to ensure both authenticity and sufficient volume for model training:

Authentic Documents: This subset includes 35 real-world ConOps documents, primarily from the aviation domain. These were sourced from publicly accessible repositories such as the Community Research and Development Information Service (CORDIS) [15] and the NASA Technical Reports Server (NTRS) [16]. The corresponding STPA elements for these authentic documents were generated by human experts, adhering to the guidelines presented in the STPA Handbook [3].

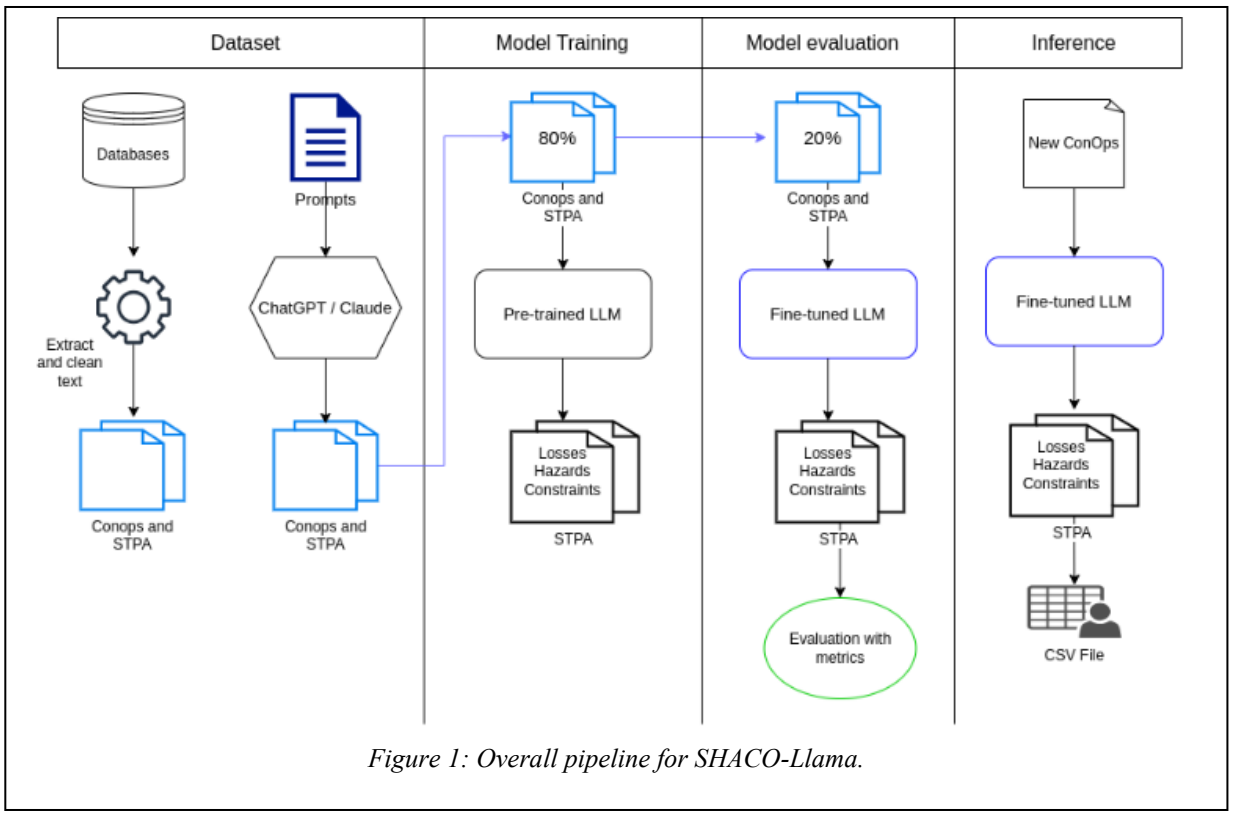

Figure 1: Overall pipeline for SHACO-Llama.

AI-Generated Documents: To significantly augment the dataset, an additional 99 ConOps documents along with their associated STPA analyses were generated. This was achieved through sophisticated prompt engineering techniques applied to state-of-the-art LLMs, including ChatGPT and Claude Sonnet 3.5. This generative approach was necessitated by the inherent difficulty in sourcing a large volume of publicly available, authentic ConOps documents with corresponding STPA analyses, often due to proprietary or sensitive information. The generation process involved carefully designed prompts and a structured pipeline to ensure the structural integrity and content validity of the synthetic documents, demonstrating an emerging application of LLMs as tools for creating training data for other specialized AI models.

The development of the SHACO dataset represents a significant undertaking, as the efficacy of fine-tuned LLMs is heavily contingent on the quality, relevance, and scale of the training data. This hybrid approach provided a pragmatic solution to the challenge of data scarcity in this specialized domain.

3.3 Model Adaptation: Fine-tuning Llama3.1-8B

The adaptation of the pre-trained Llama3.1-8B model for the specific task of generating STPA elements from ConOps documents was achieved through a supervised fine-tuning process. This methodology employed transfer learning, thereby building upon the extensive foundational knowledge already encoded within the Llama3.1-8B model while tailoring its capabilities to the nuances of STPA element extraction. The SHACO dataset was partitioned, with 80% of the data allocated for training the model and the remaining 20% reserved for testing and evaluation.

The fine-tuning workflow involved tokenizing the raw text from the SHACO dataset using the Llama3.1-8B model's native tokenizer, facilitated by the Hugging Face Transformers library [18]. The actual fine-tuning was orchestrated using the SFT Trainer module from the Hugging Face TRL (Transformer Reinforcement Learning) library. Given the substantial size of the Llama3.1-8B model, several advanced optimization techniques were crucial for enhancing training efficiency and managing computational resources:

Unsloth Framework: This framework was integrated to achieve approximately 2.4 times faster training speeds and a 58% reduction in memory utilization, making the fine-tuning process feasible on single-GPU infrastructures.

QLORA (Quantized Low-Rank Adaptation): This Parameter-Efficient Fine-Tuning (PEFT) technique was employed to minimize the number of trainable parameters while preserving model performance. QLORA implementation involved 4-bit NormalFloat (NF4) quantization of the base model, double quantization to further compression constants, and paged optimizers to manage memory fluctuations during training. Small, trainable Low-Rank Adaptation (LoRA) matrices (with rank r=16 and LoRA alpha of 16) were added to key weight matrices within the transformer architecture, specifically the query, key, value, output, and feed-forward network projection layers. The LoRA update is represented by $h=Wx+\Delta Wx=Wx+BAx$, where W are the frozen pre-trained weights and ΔW is factorized into low-rank matrices $B \in Rd \times r$ and $A \in Rr \times k$.

The model was fine-tuned for 10 epochs, utilizing a maximum sequence length of 8,192 tokens to accommodate comprehensive ConOps documents. An AdamW 8-bit optimizer with a linear learning rate scheduler was used for the training process. The application of PEFT techniques like QLORA was instrumental in making the fine-tuning of such a large model tractable on consumer-grade hardware, thereby democratizing access to advanced LLM capabilities for specialized research domains. The overall process, from initial data acquisition to the final deployment for inference, is visually captured in Figure 1.

See Section 5.1.1 for the SHACO-Llama evaluation methodology.

4 The Second Pipeline: BEDS

The first pipeline - just described in section 3 - aims to extract hazards, losses and constraints from CONOps documents applying the summarization task. Like all uses of LLMs, this task can be error prone [10,11]. Thus, to ensure more quality to the extracted sentences, the second pipeline begins with a LLM specifically trained to classify hazards, losses and constraints. In addition, the creation of this classifier allows users who already have pre-defined hazards, losses and constraints, can verify their adequation to the STPA process without having to use the first pipeline. This gives the possibility of individual use of each pipeline.

Therefore, a second pipeline, named BEDS (BERT Error Detection for STPA) was developed. BEDS is a tool to help identifying possible writing errors in sentences generated from the analysis and to offer suggestions to correct these errors. The pipeline uses a combination of classification models, in a way each sentence goes through a path of a specific combination of models according to its first classification into the three classes (hazard, loss and constraint).

The primary objective of the BEDS pipeline is to reduce the time and effort required to verify the first step of the STPA analysis and allow the analysts to allocate their time to more time-consuming activities in other steps. As a verification step for the STPA analysis, BEDS was developed to work in conjunction with the output obtained by SHACO-Llama. However, as previously mentioned, BEDS has a modular implementation that allows independent use to check for analysis inconsistencies from any source.

4.1 Architecture of the BEDS Pipeline

As the name suggests, BEDS uses fine-tuned BERT [15] models at its core. BERT was chosen for its wide application in the scientific community for NLP tasks, and due to its ability to adapt the models to downstream tasks. In this case, the task used was classification.

The BEDS pipeline is designed to ingest the sentences related to the initial set of STPA elements as input and subsequently output a report of the validity of the inserted elements. BEDS verifies the validity of input sentences based on the STPA Handbook [3].

This four-step pipeline consists of: (1) identification of the sentence's classification as losses, hazards, or constraints; (2) verification of the sentence's validity in accordance with the STPA Handbook guidelines; (3) identification of possible faults; and (4) provision of alternative writing suggestions for the sentence. The resultant report comprises the analyzed sentence, an indicator of its validity relative to the STPA guidelines, potential faults with associated probabilities for non-valid sentences, and a list of suggested revisions along with their similarity to the original sentence. Figure 2 illustrates the workflow of the BEDS pipeline.

In the STPA Handbook [3], several key instructions are provided for correctly defining analysis elements. Regarding losses, it is essential to consider stakeholders and their valued objects, with each object translated into a loss statement. For hazards, the system boundaries under analysis must be identified, and a hazard statement should describe a system state that, in conjunction with adverse environmental conditions, results in a loss. Vague descriptions such as "component failure" and "unsafe state" should be avoided. Finally, constraints are defined as conditions that must be satisfied to prevent hazards, typically by inverting the hazard condition in the statement.

In this pipeline, the capacity of the model to correctly classify valid and non-valid sentences is crucial, for this classification determines if the sentences are adequate or if the sentences need further analysis.

4.2 Dataset curation: STPA Step 1 Sentences

Also affected by the scarcity of STPA-related data, BEDS needed a suitable dataset to train the models for their intended purposes. The STPA Step 1 Sentences dataset is made up of sentences from various domains that are used as the main data for training.

This dataset was created by extracting fragments of STPA analyses found in presentations at the MIT STAMP Workshops *(https://psas.scripts.mit.edu/home/stamp-workshops/)*. This dataset contains nine columns, of which the first is the prediction data (the extracted sentence), three are target labels for classification (each for the first three pipeline steps), and the last five are metadata to improve traceability. The first target label is the class of the sentence, which can be one of the three STPA elements. The second target label is the validity of the sentence according to the STPA Handbook, which can be valid or non-valid. The last target label, given only to the invalid sentences, is the main fault observed in the sentence that differs from what is recommended by the handbook. The remaining columns are the metadata related to the sentence, such as domain, year, title, URL, and number of the presentation.

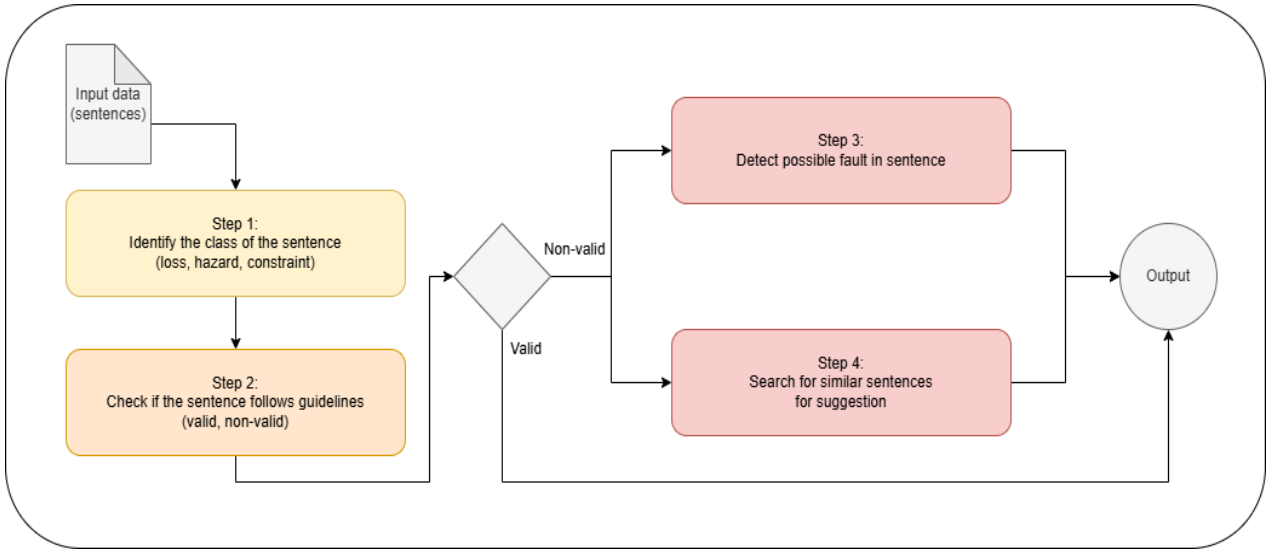

Figure 2: BEDS Pipeline Architecture

The resulting dataset contains 1,084 rows of varied classes and domains. 50 of those, however, were manually written (through paraphrase) and added to compensate for some class imbalance found in the dataset. As the dataset is an integral part of the pipeline development, the validity column was verified by specialists with three to four years of experience in STPA analysis. This contribution was necessary to ensure the valid sentences in this dataset correctly follow the STPA guidelines, and that the models can reliably classify the sentences.

4.3 Model Adaptation: Fine-tuning BERT

The BEDS pipeline uses specifically the "bert-base-uncased" model [17] throughout its classifiers. The fine-tuning process included converting categorical labels to numerical values, creating dataset subsets for each pipeline step, and encoding text with BERT's tokenizer. The training was facilitated by the Hugging Face Transformers library [18].

The classification models were evaluated using a 5-fold cross-validation technique, stratified on the class column. This allows the partitioning of 80% for training and 20% for testing for each iteration, where the testing data is not seen anywhere during training. As the final value for each step's performance, the mean between the results from the five iterations was used.

For the final step of BEDS, a sentence-transformer similarity model [19] using "all-mpnet-base-v2" is employed to suggest a sentence similar to the original. The sentence similarity model was trained using the "ContrastiveTensionLossInBatchNegatives" loss function.

5 Analysis and Results

In this section we present the results of preliminary evaluations of the two pipelines developed for the automation of the first step of STPA.

5.1 SHACO-Lhama Pipeline Evaluation

This subsection presents the evaluation results for the SHACO-Llama pipeline, which was developed to automate the initial step of the STPA by extracting losses, system-level hazards, and safety constraints directly from Concept of Operations (ConOps) documents.

5.1.1 Evaluation Methodology

The performance of the SHACO-Llama model was primarily assessed using the BERT-Score metric [20]. BERT-Score was selected over traditional metrics like BLEU or ROUGE because it evaluates semantic similarity rather than exact lexical matching, a critical distinction for LLM-generated STPA elements where conveying precise meaning matters more than reproducing identical tokens or phrases. The evaluation focused on three components of BERT-Score: Precision (semantic relevance of generated output to the reference), Recall (semantic representation of reference text in the output), and the F1-Score (the harmonic mean of Precision and Recall).

5.1.2 Quantitative Performance

The SHACO-Llama model, based on a fine-tuned Llama3.1-8B architecture, was evaluated on a reserved 20% test portion of the SHACO dataset. The quantitative performance, as measured by BERT-Score, is summarized in Table 1.

Table 1: BERT-Score results for SHACO-Llama.

Precision	Recall	F1-Score
0.7937 (~79%)	0.8109 (~81%)	0.8022 (~80%)

5.1.3 Discussion of Results

The F1-score of approximately 80% for SHACO-Llama indicates a substantial potential for such models to serve as assistive tools for safety analysts, potentially reducing the time and manual effort for the initial STPA steps. However, the F1-score also underscores that full automation without human oversight and verification is not yet advisable, particularly in safety-critical domains. While the BERT-Scores suggest good semantic alignment with expert-defined STPA elements, qualitative reviews by experienced STPA analysts revealed that some model-generated hazard formulations occasionally deviated from the strict principles outlined in the STPA handbook [3], highlighting the continued need for expert validation. The model development and evaluation faced constraints that likely influenced performance. The primary limitation was hardware resources, with training conducted on a single NVIDIA A100 GPU with 40 GB of VRAM. More extensive computational power could have enabled the exploration of larger models or more exhaustive hyperparameter tuning. Secondly, data availability posed challenges. While the SHACO dataset was curated for this research, a larger and more diverse set of authentic, expert-annotated ConOps and STPA analyses would be ideal for enhancing model generalization. The current reliance on a significant portion of AI-generated ConOps, though a pragmatic approach, might introduce certain biases. Given these constraints, an F1-score of about 80% indicates potential for enhancement with better resources and data.

5.2 BEDS Pipeline Evaluation

This subsection presents the evaluation results for the BEDS pipeline, which was developed to help verify the losses, system-level hazards, and system-level constraints from the initial step of the STPA.

5.2.1 Evaluation Methodology

The models trained in this pipeline were evaluated using traditional classification metrics. As a classification task, Accuracy is widely used to measure the general performance of the models. Meanwhile, F1-Score (the harmonic mean of Precision and Recall) is a robust alternative metric to better evaluate models in cases where the data used has a class imbalance. Closer results between the Accuracy and F1-Score mean a stable overall performance, while a gap in the metrics can show a difficulty of the model to generalize an imbalanced dataset.

5.2.2 Quantitative Performance

The BEDS pipeline is divided into three classification steps built on BERT models. The quantitative performance, calculated as the mean from 5-fold cross-validation, is summarized in Table 2.

Table 2: Classification results for BEDS.

Step	Accuracy (σ)	F1-Score (σ)
1 - Identify the class	95.20%	95.08%
2 - Determine the validity	88.61% (±7.24)	86.27% (±7.68)
3 - Classify possible faults	83.44% (±9.11)	73.16% (±16.44)

The pipeline organizes the input data through filters (the first filter is the sentence class, and the second filter is the validity of the sentence) so each sentence can be processed by a dedicated classifier. The BEDS first step consists of a single classifier, which achieved 95.20% of accuracy and 95.08% of F1-Score. The second step consists of three classifiers, one for each class (losses, hazards, and constraints) which achieved 90.37% (loss), 79.00% (hazard) and 96.47% (constraint) of accuracy, and 89.08% (loss), 75.78% (hazard), and 93.95% (constraint) of F1-Score, respectively. Similarly, the BEDS third step consists of three classifiers which achieved 74.50% (loss), 79.59% (hazard), and 96.25% (constraint) of accuracy, and 66.35% (loss), 57.32% (hazard), 95.83% (constraint) of F1-Score, respectively. The overall performance of each step, calculated as the mean between its classifiers, is summarized in Table 2.

BEDS showed decreasing values of the metrics as the pipeline advanced. It was also possible to observe an increase in the standard deviations over the three steps. As previously mentioned, for each successive step in the pipeline the input sentences are filtered and inserted into a dedicated classifier. This mechanism is also implemented during training, so each model can learn through a specific group of data. This causes a scarcity of data to be used both during training and testing, which may indirectly affect the results in later steps.

5.2.3 Discussion of Results

BEDS showed decreasing values of the metrics as the pipeline advanced. To give precise predictions, BEDS organizes the input data through filters (such as the class or validity of the sentence) so each sentence can be processed by a dedicated classifier. This mechanism is also implemented during training, so each model can learn through a specific group of data. This causes a scarcity of data to be used both during training and testing, which may indirectly affect the results in later steps.

Another phenomenon observed is the disparity among the results of the last classification step and the other two classification steps. Although the third step keeps a similar accuracy to the previous two steps, its F1-Score has a lower value. It is possible to interpret this problem as the classifier voting for the majority class in most of the sentences (hence the high accuracy), but it is not as efficient in detecting and classifying sentences in minority classes (resulting in an increased number of false positives and negatives, consequently decreasing the F1-Score). The small amount of

data mentioned previously, combined with a possible class imbalance found in the third step, could be the cause of the lower results.

5.2.4 Identified Limitations

Similarly to the SHACO-Llama pipeline, BEDS suffers from the small amount of data available in its dataset, both for training and testing the models. Data is a vital part of the research, as it enables the pipeline to learn from more examples and better generalize the classification predictions.

Another problem, specific to BEDS, is that it does not take into account the context of the sentence, nor the specialized human knowledge expected to interpret each sentence. Similarly, it does not take into account domain specificities such as certain practices or conventions known within an industry or company. Consequently, BEDS analyses each sentence in an isolated context and predicts only based on what it has learned through its training phase. Considering these limitations, BEDS also has potential for future improvement.

6 Conclusion

This work presented two AI-driven pipelines designed to automate and enhance the initial steps of System-Theoretic Process Analysis (STPA) in the aviation domain. The SHACO-Llama pipeline employs a fine-tuned Llama3.1-8B model to extract safety-relevant elements—losses, hazards, and constraints—from ConOps documents, achieving promising results in terms of precision, recall, and F1-score. Complementarily, the BEDS pipeline supports verification and correction of the extracted elements using fine-tuned BERT models, demonstrating high classification accuracy across various stages.

The preliminary results highlight the potential of these pipelines to significantly reduce the time, cost, and expertise required in early-stage STPA analysis while maintaining high levels of accuracy and interpretability. Nonetheless, certain limitations, such as reliance on synthetic training data and the need for domain-specific contextual understanding, underscore the importance of human oversight in safety-critical applications.

The first pipeline, SHACO-Llama, represents a significant innovation by leveraging a fine-tuned Llama3.1-8B language model to automatically extract critical STPA elements—losses, hazards, and safety constraints—directly from Concept of Operations (ConOps) documents. This contribution marks a notable advancement in applying large language models (LLMs) to interpret complex technical documentation and support safety analysis at scale, offering both high semantic fidelity and improved analyst productivity.

The second pipeline, BEDS (BERT Error Detection for STPA), complements SHACO-Llama by introducing an intelligent verification layer. It classifies the extracted elements, checks their validity against STPA guidelines, detects specific writing faults, and suggests corrected formulations. This pipeline not only improves the reliability and trustworthiness of automated STPA outputs but also reduces the cognitive load on human analysts by flagging inconsistencies and offering refined alternatives.

Together, these pipelines demonstrate a synergistic contribution: SHACO-Llama accelerates the extraction process, while BEDS ensures the quality and correctness of the outputs. Preliminary evaluation results indicate strong performance, with SHACO-Llama achieving an F1-score of ~80% and BEDS reaching up to 95% accuracy in classification tasks.

While challenges remain—particularly in data availability, contextual understanding, and full automation—the presented pipelines offer a foundational framework for integrating AI into safety-critical engineering workflows.

Future work will focus on expanding the dataset with additional real-world examples, refining the models' capabilities with larger architectures and improved training methodologies, and integrating both pipelines into a cohesive tool chain. Ultimately, this research contributes to the advancement of semi-automated safety analysis methods and lays the groundwork for broader adoption of AI in the field of aviation system safety. Moreover, the authors intend to: (i) increase the datasets used in order to improve the metrics obtained in both, pipeline 1 and pipeline 2; (ii) create sanity check modules in pipeline 1 in order to verify that there are no requirements documents (ConOps) with missing or invalid parts according to the STPA process; (iii) test document summarization and sentence classification approaches using LLMs directly and compare them to the fine-tuned models

Acknowledgment

The authors wish to thank São Paulo Research Foundation (FAPESP). This work was supported in part by a grant from 2023/03393-5. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] N. G. Leveson. Engineering a Safer World: Systems Thinking Applied to Safety. The MIT Press, 2012.

[2] N. G. Leveson. An Introduction to System Safety Engineering. The MIT Press, 2023.

[3] N. G. Leveson and J. P. Thomas. STPA Handbook, 2018. Available: https://psas.scripts.mit.edu/home/get_file.php?name=STPA_Handbook.pdf

[4] J. Chen, S. Zhang, Y. Lu and P. Tang, "STPA-based hazard analysis of a complex UAV system in take-off," 2015 International Conference on Transportation Information and Safety (ICTIS), Wuhan, China, 2015, pp. 774-779, doi: 10.1109/ICTIS.2015.7232133

[5] B. Olberts and Y. Dittjen, "Model Based STPA for Assisted Driving Functions," 2023 ACM/IEEE International Conference on Model Driven

Engineering Languages and Systems Companion (MODELS-C), Västerås, Sweden, 2023, pp. 85-86, doi: 10.1109/MODELS-C59198.2023.00027.

[6] A. Carniel, J. D. M. Bezerra and C. M. Hirata, "An Ontology-Based Approach to Aid STPA Analysis," in IEEE Access, vol. 11, pp. 12677-12697, 2023, doi: 10.1109/ACCESS.2023.3242642.

[7] E. A. Celik et al., "Application of STPA for the Elicitation of Safety Requirements for a Machine Learning-Based Perception Component in Automotive". In: TRAPP, M.; SAGLIETTI, F.; SPISLÄNDER, M.; BITSCH, F. (Ed.). Computer Safety, Reliability, and Security. Cham: Springer International Publishing, v. 13414. P. 319–332. 2022. Available: https://link.springer.com/10.1007/978-3-031-14835-4_21/.

[8] S. Ghosh, A. Zaboli, J. Hong, J. Kwon, "An Integrated Approach of Threat Analysis for Autonomous Vehicles Perception System". *IEEE Access*, v.11, p.14752–14777, 2023. Available: https://ieeexplore.ieee.org/document/10041909/.

[9] S. Diemert, J.H. Weber, "Can Large Language Models Assist in Hazard Analysis?" In: Guiochet, J., Tonetta, S., Schoitsch, E., Roy, M., Bitsch, F. (eds) *Computer Safety, Reliability, and Security.* SAFECOMP, Workshops. SAFECOMP 2023. Lecture Notes in Computer Science, vol 14182. Springer, 2023. Available: https://doi.org/10.1007/978-3-031-40953-0_35/.

[10] Yi Qi, Xingyu Zhao, Siddartha Khastgir, Xiaowei Huang, "Safety analysis in the era of large language models: A case study of STPA using ChatGPT", *Machine Learning with Applications*, Volume 19, 2025. Available: https://doi.org/10.1016/j.mlwa.2025.100622/.

[11] S. Charalampidou, A. Zeleskidis, I. M. Dokas, "Hazard analysis in the era of AI: Assessing the usefulness of ChatGPT4 in STPA hazard analysis", *Safety Science*, Volume 178, 2024. Available: https://doi.org/10.1016/j.ssci.2024.106608/.

[12] A. Grattafiori et al. "The Llama 3 Herd of Models". 2024. Available: https://arxiv.org/abs/2407.21783.

[13] IEEE. IEEE Guide for Information Technology - System Definition - Concept of Operations (ConOps) Document. IEEE Std 1362-1998, p. 1–24, 1998.

[14] ISO/IEC/IEEE. ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes –Requirements engineering. ISO/IEC/IEEE 29148:2011(E), p. 1–94, 2011.

[15] CORDIS. "Community Research and Development Information Service," 2025. [Online]. Available: https://cordis.europa.eu/. [Accessed: Nov. 5, 2024].

[16] NTRS. "NTRS - NASA Technical Reports Server," 2025. [Online]. Available: https://ntrs.nasa.gov/. [Accessed: Feb. 11, 2025].

[17] J. Devlin et al. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding". arXiv:1810.04805, arXiv, 24 May 2019. arXiv.org, https://doi.org/10.48550/arXiv.1810.04805.

[18] T. Wolf et al. "HuggingFace's Transformers: State-of-the-art Natural Language Processing". 2020. Available: https://arxiv.org/abs/1910.03771.

[19] Reimers, Nils, and Iryna Gurevych. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. arXiv, 2019. DOI.org (Datacite), https://doi.org/10.48550/ARXIV.1908.10084.

[20] T. Zhang et al. "BERTScore: Evaluating Text Generation with BERT". 2020. Available: https://arxiv.org/abs/1904.09675.