Aviation-BERT: A Preliminary Aviation-Specific Natural Language Model

Chetan Chandra *, Xiao Jing *, Mayank V. Bendarkar †, Kshitij Sawant *, Lidya R. Elias *, Michelle Kirby ‡, Dimitri N. Mavris§

Aerospace Systems Design Laboratory, School of Aerospace Engineering
Georgia Institute of Technology, Atlanta, Georgia, 30332

Data-driven methods form the frontier of reactive aviation safety analysis. While analysis of quantitative data from flight operations is common, text narratives of accidents and incidents have not been sufficiently mined. Among the many use cases of aviation text-data mining, automatically extracting safety concepts is probably the most important. Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based large language model that is openly available and has been adapted to numerous domain-specific tasks. The present work provides a comprehensive methodology to develop domain-specific BERT model starting from the base model. A preliminary aviation domain-specific BERT model is developed in this work. This Aviation-BERT model is pre-trained from the BERT-Base model using accident and incident text narratives from the National Transportation Safety Board (NTSB) and Aviation Safety Reporting System (ASRS) using mixed-domain pre-training. Aviation-BERT is shown to outperform BERT when it comes to text-mining tasks on aviation text datasets. It is also expected to be of tremendous value in numerous downstream tasks in the analysis of aviation text corpora.

I. Introduction and Background

Commercial aviation safety continues to be of paramount importance as air transport is poised to overtake pre-COVID pandemic levels by the end of 2023 [1]. Safety risk management has two parts – a forward-looking proactive approach and a backward-looking reactive approach. With the advent of novel aircraft technologies and configurations to meet aggressive efficiency targets, the traditional proactive methods need to be updated. Advanced design-based methods for ensuring system safety in this rapidly changing paradigm have been proposed [2–4]. While these work for individual new aircraft designs, they cannot by themselves assess the safety risk of aviation operations at the level of the National Airspace System (NAS). Reactive approaches typically involve investigating incidents or accidents in order to determine probable causes and provide recommendations to avoid future failures. Reactive approaches focused on analysis of a handful of accidents have been traditionally used for rule-making to ensure similar accidents are not repeated. However, the hundreds of thousands of aviation accident and incident text reports can also be utilized to estimate the probabilities of different safety scenarios in the NAS. The Integrated Safety Assessment Model (ISAM) is one such tool under development by the FAA to provide a baseline safety risk assessment and model the risk of any proposed changes to the NAS [5].

While traditional accident analysis conducted manually still holds merit, data-driven methods now form the frontier of reactive aviation safety analysis. The widespread presence of advanced sensors to monitor flight parameters has generated large heterogeneous datasets which may be used to gain insights into the safety and quality of aircraft operations. The National Transportation Safety Board (NTSB) Report Database[1] and Aviation Safety Reporting System (ASRS)[1] are two openly available aviation databases that store accident and incident text narratives and are used in this work. Past quantification of the safety models within ISAM extensively utilized manual analysis of the NTSB and ASRS databases among other sources [6, 7]. Automating the quantification of ISAM models using text-based machine learning techniques to extract the safety facts from such historical accident and incident data form an important next
step. However, these text-based techniques present a challenge in their subjectivity because they must first convert the abstract and subjective language of accident and incident narratives into numerical representations which computers are capable of understanding and processing. Natural language processing (NLP) is a field that was developed for this purpose, so that large-scale text corpora may be searched, classified, and analyzed.

Some of the simplest text processing techniques involve text representations as a weighted method such as Term Frequency-Inverse Document Frequency (TF-IDF) [9]. Singh et al. [10] compared the clustering results for a dataset of standard newswire articles and found that the use of TF-IDF with stemming resulted in the most successful clustering. Miyamoto et al. [11] investigated the ASRS dataset filtered for flight delays using TF-IDF and found a unique safety perspective to delay cause identification. Pre-processing of all unnecessary characters, keywords, tags, and punctuations was found to be vital to the successful utilization of TF-IDF techniques by multiple studies [12][13]. This however has a tendency to lose contextual information stored in language. For example, root words like ‘break’ from the following sentences are considered the same in the TF-IDF approach - ‘ATC was on a break’ vs. ‘the structure reached its breaking point’. Capturing this lost context is vital in extracting safety events stored in the accident and incident reports. Without capturing context effectively, an automated NLP pipeline cannot expect to quantify a safety model like ISAM.

Transformer-based models that utilize self-attention form the current state-of-the-art in capturing context in NLP. A particular model - Bidirectional Encoder Representations from Transformers (BERT) broke many NLP leaderboard records when it was released [15] and is of interest in the present work. BERT has been trained on multiple domain-specific languages. For example, the BERT model was pre-trained on the biomedical text corpus to create BioBERT - a language model for biomedical text mining [16]. SciBERT is a similarly pre-trained from scratch BERT model that was trained on scientific text corpus [17]. LEGAL-BERT was pre-trained using both mixed-domain pre-training as well as pre-trained from scratch for use in the legal domain [18]. Similarly, FinBERT was created for financial text and sentiment analysis [19][20]. All these models provided better domain-specific language understanding than the BERT base model and provided guidance on how domain-specific language models may be developed.

The interest in using BERT for aviation text data mining is recent. Ref. [21] applied the BERT base model to determine the answer to a question - “When did the incident happen?” to a set of ASRS reports with reasonable success in getting relevant answers from the text narratives. In [22], the authors tested a combination of knowledge graphs along with BERT and GPT-3 (another transformer-based model) to extract answers to questions like “Which incidents were related to crosswind?” They found that combining the transformer models with an aviation knowledge graph provided better results than just using a transformer model. Ref. [23] pre-trained a compact model based on Robustly optimized BERT (RoBERTa) from scratch on the ASRS narratives and compared it to the RoBERTa model to find that it was able to perform comparable across multiple natural language understanding (NLU) tasks while being much compute-efficient. However, to the best of the authors’ knowledge, there is no openly available, pre-trained, aviation domain-specific language model in literature.

There are two broad approaches to pre-training large language models (LLMs) like BERT for domain-specific applications. The first one is called mixed domain pre-training and includes utilizing the English language pre-trained model and then further pre-training it on domain-specific text data. The second is called pre-training from scratch, where the LLM is pre-trained with no text data other than the domain-specific text. This latter approach generally results in better outcomes for downstream tasks like classification, question answering (QA), etc. but requires a large available dataset [24]. The present work only considers about 540,000 aviation-specific text narratives from NTSB and ASRS for pre-training, which is much smaller than what is usually used for pre-training from scratch. Hence, a mixed-domain pre-training method is selected.

The present work seeks to serve as a methodological guide for readers for domain-specific pre-training of BERT. The preliminary Aviation-BERT model described here will be improved upon in future work by incorporating additional aviation-specific text data. This Aviation-BERT model has also been used for multi-label classification tasks on ASRS data in a parallel effort [25] where it shows performance improvements over BERT-Base-Uncased across the board. The rest of this paper is organized as follows: Section II will explain the aviation text datasets, the steps taken to generate an aviation-specific version of BERT, and the performance metrics that will be utilized to measure its effectiveness. Section III will discuss results while Section IV concludes the present work and mentions opportunities for future work.
II. Method

An overview of the methodology is provided in Figure 1. The following subsections detail the steps taken to develop this preliminary Aviation-BERT model.

Fig. 1  Methodology overview for data collection, pre-processing, vocabulary modification, language model pre-training, leading up to evaluation

A. Datasets

1. The ASRS Corpus

The Aviation Safety Reporting System (ASRS) is a program developed by NASA that collects voluntary and de-identified reports of unsafe occurrences in the aviation industry [26]. The program’s main objective is to identify deficiencies in the safety of the national airspace system (NAS) so that corrective actions can be implemented. The ASRS system works on a preventative model – rather than waiting for accidents to occur, aviation stakeholders like pilots, air traffic controllers, and maintenance technicians submit reports that identify deficiencies without an accident occurring [27]. This information submitted when no accidents occur can fill in the gaps often left in accident investigations by identifying potential causal factors including the events leading up to the accident, factors that increased risk, how problems were detected, and attempts made to resolve the problems [27]. Over almost 40 years of operation, the ASRS database has received over 1.78 million reports that span a wide range of technical and human-centered aviation safety events [27]. The ASRS database contains journalistic narratives on a broad spectrum of events resulting from both human and technical factors. Each unique incident narrative is one row along with 91 columns (representing metadata about each corresponding incident). The metadata in the columns, generated by either experts or reporters, provides context to reports of the incidents. The columns provide a detailed summary of the incident conditions and variables, including information on the “type of operations, the type of aircraft, the reporter’s qualifications, the weather, the type of airspace, and a variety of other event-specific characteristics” [26]. A crucial part of the reported incident is contained in the narrative section where details are provided on the actual occurrences before, during, and after the incident. This semi-structured dataset can be assessed from the ASRS website [3] where one can get information on flight narratives and metadata using filters such as the “data of occurrence, aircraft type, flight environment, location, persons involved, and event assessment.” The following columns contain the narratives within ASRS used in the present work: Reporter 1 Narrative, Reporter 2 Narrative, Reporter 1 Callback, Reporter 2 Callback, and Reporter 1 Synopsis.

2. The NTSB Corpus

The National Transportation Safety Board (NTSB) is an independent federal agency that is responsible for the following: investigating aviation accidents and incidents, recording findings, and maintaining a publicly available accident and incident database. This database includes information regarding civil aviation accidents and the select few incidents that have occurred on US soil and international waterways since 1962 [26]. Preliminary reports are filed

[https://asrs.arc.nasa.gov/](https://asrs.arc.nasa.gov/)
3. Pre-processing

Textual data gathered from 223,576 ASRS and 84,281 NTSB reports under all relevant columns is merged into a single dataset and qualitatively refined in preparation for pre-training of the language model. The first step involves text cleanup, where new lines and extra spaces are removed, followed by replacing incorrectly placed special symbols with appropriate ASCII characters. The second step is to remove duplicate texts and those shorter than 70 characters. Both these steps are performed while continuously surveying the entire dataset. Figure 2 provides an illustration of the text cleanup process with examples. A total of 546,789 texts are obtained after pre-processing is complete.

To ensure future flexibility in utilizing this dataset with other models, the text case is left unchanged. Case specificity pertaining to the models is managed by the tokenizer prior to data consumption during pre-training.

Fig. 2 Cleanup process of raw ASRS and NTSB text with examples.

The refined dataset is then shuffled and split into training and testing datasets, with the latter containing 15% of all texts. After refinement and splitting, the training dataset has 464,771 texts and the testing dataset contains 82,018 texts.

B. Vocabulary and Tokenizer

BERT uses tokenization to process input text by replacing it with a sequence of tokens. WordPiece tokenization [30, 31], an adaptation of Byte-Pair Encoding (BPE) [32], is used to develop the vocabulary for pre-trained BERT-Base. This method breaks words down into smaller units, or subwords, enabling a more efficient representation of out-of-vocabulary words. The final vocabulary of pre-trained BERT-Base is approximately 30,000 tokens comprising the most
Table 1  Tokenization Examples

<table>
<thead>
<tr>
<th>Word</th>
<th>BERT-Base Tokenizer</th>
<th>Aviation-Specific Tokenizer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATC</td>
<td>at ##c</td>
<td>atc</td>
</tr>
<tr>
<td>TRACON</td>
<td>tr ##aco ##n</td>
<td>tracon</td>
</tr>
<tr>
<td>avionics</td>
<td>av ##ion ##ics</td>
<td>avionics</td>
</tr>
<tr>
<td>tailboom</td>
<td>tail ##bo ##om</td>
<td>tailboom</td>
</tr>
<tr>
<td>B747</td>
<td>b ##7 ##47</td>
<td>b747</td>
</tr>
<tr>
<td>RWYS</td>
<td>r ##wy ##s</td>
<td>rwys</td>
</tr>
<tr>
<td>Airworthiness</td>
<td>air ##worth ##iness</td>
<td>airworthiness</td>
</tr>
<tr>
<td>acft</td>
<td>ac ##ft</td>
<td>acft</td>
</tr>
<tr>
<td>AUTOTHROTTLES</td>
<td>auto ##th ##rot ##tile ##s</td>
<td>autothrottles</td>
</tr>
</tbody>
</table>

frequently occurring words and subwords in the English Wikipedia and Book corpus [15]. This BERT-Base-Uncased vocabulary falls short of containing many common aviation-specific terms. Due to this deficiency, the BERT-Base models are forced to divert parametrization capacity and training bandwidth to model domain-specific terms using fragmented subwords [24].

To obtain a more domain-specific vocabulary for aviation-related tasks, the WordPiece tokenization method is applied to the combined ASRS plus NTSB dataset. The selected algorithm for tokenization generates subwords based on their frequency of occurrence in the dataset. A minimum frequency threshold is set to determine the least number of times a subword must appear before being included in the final vocabulary. Additionally, a cut-off value is used to limit the maximum number of subwords to be included, thereby controlling the size of the vocabulary. The frequency and cut-off values are determined using Zipf’s law, which states that word frequency in any language is inversely proportional to its rank in a frequency table [33].

The generated vocabulary specific to ASRS and NTSB texts is integrated into the BERT-Base vocabulary by removing preexisting subwords. A tokenizer with an architecture identical to that of BERT-Base is then created, which incorporates the new vocabulary consisting of 1926 additional tokens. This enhances the tokenizer’s familiarity with frequently used out-of-vocabulary words. Examples demonstrating tokenization of a few commonly occurring aviation words are provided in Table 1. Section II.C provides additional details on the tokenization process and how the tokenizer handles upper and lower case words.

C. Language Model Pre-training

BERT [15] is a language model based on Transformers [14] that has advanced the state-of-the-art performances on Natural Language Understanding (NLU) tasks. NLU is a field of Natural Language Processing (NLP) that helps machines conduct tasks like document classification, question answering, or story comprehension by maximizing their understanding of language. BERT uses a transformer-based architecture, which enables it to learn extended dependencies in textual data by employing a self-attention mechanism consisting of multiple layers and heads. The Base version of BERT is used in the present work. It is distinct from other versions in that it has 12 encoder layers, 12 self-attention heads, a hidden embeddings size of 768, and a pre-trained vocabulary of approximately 30,000 words, resulting in 110 million parameters. Interested readers are referred to the work of Devlin et al. [15] for a more thorough explanation of the model architecture.

As mentioned in Section I using domain-specific data for pre-training has shown promising results on various text corpora, such as biomedical, scientific, finance, and legal. There are two primary approaches for pre-training BERT-Base for such applications: “pre-training from scratch” and “mixed-domain pre-training.” The former involves providing a massive domain-specific text corpus consisting of billions of words to an untrained BERT-Base model [17/24], resulting in better domain-specific models that may lack some basic language capabilities and is computationally expensive. In the latter approach, BERT-Base (already pre-trained on the English language) is further pre-trained on domain-specific language [16/18–20]. This requires much lesser training data and is computationally cheaper. The second strategy is utilized in the present work to pre-train Aviation-BERT.

BERT-Base [15] is offered with cased and uncased vocabularies [1]. In the cased version, the input text is left unchanged, preserving the original text case, i.e., the input has upper and lower case letters. In the uncased version,
the input text is first converted to lowercase before tokenization. The choice of using either variation ultimately depends on the characteristics of the dataset and the specific requirements of the downstream task. The ASRS corpus mentioned in Sec. II.A.1 has a considerable portion of the text entirely in uppercase (nearly 148,000 texts after pre-processing). The ability to clearly characterize case-related nuances is greatly diminished for the given ASRS plus NTSB dataset. Therefore, for simplicity, BERT-Base-Uncased is selected for the present work. A BERT-Base-Uncased model pre-trained on English Wikipedia and BookCorpus is accessible through the Hugging Face Transformers library.

BERT’s high level of understanding of contextual representations of words and sentences is attributed to extensive pre-training on two unsupervised learning tasks. These tasks are Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) [15]. MLM involves randomly masking a certain percentage of tokens in the input sentence and training the model to predict the original token. The pre-trained version of BERT-Base-Uncased implements MLM by choosing 15% of the input tokens, of which 80% are masked using the special token [MASK], 10% are replaced with a random token, and 10% are left unchanged. The model is then trained for cross-entropy loss between predicted and original tokens as objective. For the present work, the same process for MLM, but with Whole Word Masking (see Section II.D.2) is used to perform mixed-domain pre-training, but with the AdamW optimizer [34], which is an enhanced variant of the originally used Adam optimizer. The second learning task is NSP, which involves training the model on sentence pairs and predicting whether the second sentence comes after the first. The input for this task uses the special token [CLS] to mark the beginning of the first sentence and a [SEP] token to separate the pair. More recent studies on BERT-like models have noted that the contribution of NSP to the performance of downstream tasks may be unreliable. In some cases, it may even have a negative impact on the overall learning of the model [35–37]. Therefore, NSP is not implemented in the pre-training process of Aviation-BERT.

Pre-training is performed on two models: one utilizing the ASRS plus NTSB dataset with the original BERT-Base-Uncased vocabulary and tokenizer (referred to as Model 2 in Results), and the other using the updated vocabulary and trained tokenizer (Aviation-BERT, referred to as Model 3 in Results). A learning rate of $1 \times 10^{-4}$, batch size of 32, weight decay of 0.01, and warm-up step ratio of 15% are selected for pre-training over 3 epochs on a single Tesla V100 32GB GPU [38]. These hyperparameters are optimized through preliminary runs, considering minimum optimizer loss and perplexity as pseudo-metrics. The chosen maximum sequence length for input is 256 tokens, and a chunking strategy described in Section II.D.1 is implemented to prevent any text from being wasted due to truncation. The model with the original BERT-Base-Uncased vocabulary is trained over 41,500 steps and 132.4M tokens, while the other model is trained over 38,600 steps and 123.3M tokens.

D. Enhancement Techniques:

1. Handling Long Text

The ASRS plus NTSB dataset comprises texts of various lengths. Figure 3 illustrates the distribution of text lengths after tokenization using the updated vocabulary, assuming they are not divided into train and test sets. With a maximum input sequence length of 256 tokens, there are over 140,000 texts that exceed this limit. Truncating these texts would result in a significant and unnecessary loss of valuable data, especially considering that the total available texts amount to approximately 540,000. For this purpose, a chunking strategy is implemented to break down texts longer than 256 tokens and integrate them into the original datasets.

The chunking approach draws inspiration from a course offered by the Hugging Face community [39]. The un-truncated text is initially tokenized, with a special [CLS] token denoting the beginning of each text and a [SEP] token marking the end. The entire tokenized text is then concatenated into a single long list and divided into equally sized chunks of 256 tokens. The final chunk, which may not consist of exactly 256 tokens, is extended with special [PAD] tokens, acting as fillers, to match the size of the other chunks.

Texts within the chunked datasets may or may not have the special tokens [CLS] and [SEP] positioned precisely at the start and end of each sequence. Since NSP is not utilized in the current pre-training process, the exact placement of these special tokens at the beginning and end of a sequence does not hold significant importance [35–37]. However, the special tokens already present are retained to differentiate the internal beginnings and ends of narratives within a single sequence.

---

https://huggingface.co/bert-base-uncased
https://github.com/huggingface/transformers
https://huggingface.co/learn/nlp-course/chapter7/3?fw=tf#preprocessing-the-data
A summary of dataset sizes before and after chunking is provided in Table 2. The slight reduction in size after chunking is worth noting, as this method efficiently uses every available token in a sequence and avoids the use of \([PAD]\), except in the last chunk. This reduces the pre-training time with little to no effect on the overall performance.

<table>
<thead>
<tr>
<th>Split</th>
<th>Original Size</th>
<th>Tokenized (BERT-Base-Uncased Vocab.)+Chunked</th>
<th>Tokenized (Updated Vocab.)+Chunked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>464,771</td>
<td>443,030</td>
<td>412,918</td>
</tr>
<tr>
<td>Test</td>
<td>82,018</td>
<td>78,186</td>
<td>72,832</td>
</tr>
</tbody>
</table>

Fig. 3 A histogram showing length of tokenized text sample vs. frequency of occurrence in dataset

2. Whole Word Masking

The original BERT-Base-Uncased model utilizes a masking strategy for pre-training that randomly selects tokens, potentially resulting in partial masking of complete words. In contrast, Whole Word Masking (WWM) masks all tokens associated with a word whenever any corresponding subword is chosen. This approach presents a greater challenge for MLM pre-training and has demonstrated enhanced performance in certain downstream tasks [39]. Therefore, in the current work, WWM is selected as the pre-training masking strategy. Table 3 illustrates an example depicting a possible masking scenario with both the original BERT strategy and WWM.

<table>
<thead>
<tr>
<th>Tokenized Text</th>
<th>Original BERT Masking</th>
<th>Whole Word Masking</th>
</tr>
</thead>
<tbody>
<tr>
<td>the gyro #copter crashed on taxiway</td>
<td>the [MASK] #copter crashed on taxiway</td>
<td>the [MASK] [MASK] crashed on taxiway</td>
</tr>
<tr>
<td>marshall #ing wand #hs had low battery</td>
<td>marshall #ing [MASK] #hs had low battery</td>
<td>marshall #ing [MASK] [MASK] had low battery</td>
</tr>
</tbody>
</table>
E. Evaluation

When evaluating pre-trained language models specifically for MLM, there are currently no standard metrics to quantify their accuracy. Researchers typically evaluate pre-trained models on various downstream tasks using task-specific metrics like accuracy, F1-score, precision, recall and mean reciprocal rank [15][40]. Perplexity, a statistical measure of how effectively a language model can predict the next token given the previous tokens, is frequently used to compare the general ‘goodness’ of various language models [15][35]. Favorable perplexity may provide a high confidence level of the model, but it does not always equate to high accuracy [41].

It is important to have a fast and efficient way to assess the accuracy of pre-trained models before investing resources in downstream tasks. Considering this, the present work proposes a simplified formulation of the Top-k sampling technique (for k=5) to calculate MLM accuracy of such pre-trained models. Top-k sampling is widely used in language modeling, aiding in both training and evaluation processes [42], as well as enabling the performance of downstream tasks such as text generation [43][50].

The performance of pre-trained models in predicting masked words is evaluated using the top-5 MLM accuracy formula (Equations 1 and 2). To begin with, tokens are masked in the test dataset using the same masking strategy described in Section II.D.2. However, unlike the method for training, all 15% of the randomly selected tokens are masked without any alterations. Additionally, care is taken to restrict masking numbers, as the model may not be able to predict them accurately. Next, the evaluation algorithm returns the top-5 most probable predictions for each masked token position, and the accuracy is calculated based on these predictions. The calculation involves determining whether the true token is among the top-5 predicted tokens or not. Accuracy for each token is marked as 1 if the correct token is among the top-5 predicted tokens; else, it is marked as 0. Finally, the proportion of total masked tokens correctly predicted by the pre-trained model is computed, giving the Top-5 MLM Accuracy.

\[
\text{Top-5 MLM Accuracy} = \frac{\sum_{i=1}^{n} \text{acc}_i}{n} \tag{1}
\]

where \(n\) is the total number of masked tokens, and \(\text{acc}_i\) is the accuracy for the \(i\)-th masked token, defined as:

\[
\text{acc}_i = \begin{cases} 
1 & \text{if the correct token is in the top-5 predicted tokens} \\
0 & \text{otherwise}
\end{cases} \tag{2}
\]

The value of k=5 has gained increasing popularity and is now a prominent feature in the widely utilized mask-prediction pipeline provided by the Hugging Face platform [‡]. A low value of k may be excessively restrictive, whereas a high value could be overly lenient towards favoring MLM accuracy and thereby hindering model comparisons. Here, k=5 offers a balance for objectively comparing the comprehension of domain-specific vocabulary and context for the pre-trained models. Overall, the Top-5 MLM Accuracy method allows for quick and efficient identification of promising models for further fine-tuning and optimization on specific tasks.

III. Results

This section compares the accuracy from both a quantitative and qualitative standpoint for three models:

- **Model 1**: Original BERT-Base-Uncased model pre-trained on English Wikipedia and BookCorpus
- **Model 2**: Model 1 pre-trained further with ASRS plus NTSB dataset using the original BERT-Base-Uncased vocabulary and tokenizer
- **Model 3**: Model 1 pre-trained further with ASRS plus NTSB dataset using the updated vocabulary and trained tokenizer – Aviation-BERT

A. Top-5 MLM Accuracy

Figure 4 shows a comparison of Top-5 MLM Accuracy for the three models, clearly demonstrating the improved performance achieved through pre-training on domain-specific data. The results indicate that by incorporating ASRS and NTSB texts in the pre-training phase, the accuracy for predicting aviation-related words rises from 55.73% to 66.99%, even without modifying the vocabulary. Moreover, when high-frequency words from the ASRS plus NTSB dataset are included in the vocabulary, the percentage of correctly predicted tokens further increases to 73.78%. Overall, these findings underscore the effectiveness of domain-specific pre-training in enhancing the performance of masked language models, especially when high-frequency words from the domain are added to the vocabulary.

[‡] https://huggingface.co/tasks/fill-mask
B. MLM Word Predictions

Figure 5 shows examples of the top-5 masked token predictions for the three models, along with the probability assigned to each prediction. This represents a qualitative assessment of the domain-specific understanding demonstrated by each model. The language used in all examples closely resembles that of ASRS or NTSB narratives. Special attention has been given to ensuring that none of these texts are present in the actual dataset, thereby preventing the models from relying on memory when making predictions.

<table>
<thead>
<tr>
<th>#</th>
<th>Text</th>
<th>Masked Token Prediction</th>
</tr>
</thead>
</table>
| 1 | “They contacted the pilot as he missed the assigned exit.”
   They contacted the [MASK] as he missed the assigned exit. | 1. driver: 0.126, captain: 0.457, pilot: 0.235
   2. police: 0.037, reporter: 0.338, reporter: 0.201
   3. guard: 0.027, pilot: 0.070, pic: 0.199
   4. officer: 0.017, captain: 0.033, pilot: 0.090
   5. doctor: 0.016, pic: 0.026, rtp: 0.059 |
| 2 | “Air traffic control gave us a heading to intercept the course between LIPIETE (IF) and WASON.”
   [MASK] gave us a heading to intercept the course between LIPIETE (IF) and WASON. | 1. it: 0.334, center: 0.657, aic: 0.346
   2. this: 0.320, approach: 0.120, elle: 0.120
   3. that: 0.113, controller: 0.084, xls: 0.065
   4. he: 0.080, shu: 0.045, center: 0.038
   5. they: 0.066, he: 0.012, stay: 0.035 |
| 3 | “HE WAS TAXING AIRCRAFT INTO POSITION FOR DEPARTURE ON RUNWAY 04L.”
   HE WAS TAXING [MASK] INTO POSITION FOR DEPARTURE ON RUNWAY 04L. | 1. back: 0.327, back: 0.829, aircraft: 0.625
   2. it: 0.078, out: 0.040, back: 0.147
   3. himself: 0.058, slowly: 0.033, e17: 0.032
   4. her: 0.044, aircraft: 0.021, acft: 0.028
   5. up: 0.038, straight: 0.011, out: 0.016 |
| 4 | “HE WAS TAXING [MASK] INTO POSN FOR DEP ON RWY 04L.”
   HE WAS TAXING [MASK] INTO POSN FOR DEP ON RWY 04L. | 1. back: 0.138, back: 0.755, acft: 0.658
   2. it: 0.083, out: 0.111, back: 0.103
   3. himself: 0.081, slowly: 0.014, smt: 0.034
   4. them: 0.042, straight: 0.013, def: 0.024
   5. traffic: 0.027, forward: 0.013, out: 0.023 |
| 5 | “The most taxing part of my entire trip was the 8-hour layover.”
   The most taxing [MASK] of my entire trip was the 8-hour layover. | 1. part: 0.504, point: 0.286, phase: 0.373
   2. thing: 0.141, part: 0.215, portion: 0.269
   3. aspect: 0.126, time: 0.071, point: 0.150
   4. task: 0.029, phase: 0.066, time: 0.079
   5. event: 0.019, portion: 0.064, component: 0.032 |
Example 1: This simple example demonstrates the growing probability of predicting aviation-specific terms for Models 2 and 3, which are further pre-trained on the ASRS plus NTSB dataset. While it can be observed that no prediction leads to a syntactically incorrect sentence, Model 3 displays a more notable capability to predict entities that are pertinent to aviation narratives.

Example 2: Masking in this example is intentionally not one-to-one and therefore an exact prediction is not expected. However, it is worth noting that Models 2 and 3 demonstrate the ability to recognize words such as “center,” “controller,” “ato,” and “ctlr” as synonymous entities with Air Traffic Control. The words “ZHU,” “ZLA,” and “ZNY” are air route traffic control center codes corresponding to Houston, Los Angeles, and New York, respectively. Due to their high frequency, the pre-trained models have also acquired the knowledge to identify these codes as related to air traffic control.

Examples 3 and 4: These examples illustrate the impact of distinct writing styles, specifically the use of abbreviations and the presence of typographical errors (typos). In terms of abbreviation usage, Example 3 contains complete words, which prompts Model 3 to predict the full word “aircraft.” In contrast, Example 4 contains mostly abbreviations in the text, resulting in “acft” being the top predicted word. Furthermore, it is worth noting that Model 2 exhibits limited proficiency in predicting any word synonymous with “aircraft,” highlighting the effect of a limited vocabulary. In the case of typos, neither Model 2 nor Model 3 show a decline in performance due to the presence of the misspelled version of “taxiing.” The word “taxing” occurs thousands of times in the ASRS plus NTSB dataset, and both models have learned to recognize its intended meaning as “taxiing.” The impact of “taxing” being a homograph is discussed in Example 5.

Example 5: This example shows how the pre-trained models maintain their contextual understanding. Since “taxing” can also refer to something challenging or stressful, Models 2 and 3 in this example predict more abstract nouns like “phase” and “time” that are contextually appropriate, rather than aviation-specific terms. This indicates that the models have not disregarded their original pre-training from BERT and have not been excessively trained to solely accommodate the ASRS plus NTSB dataset.

Comparison of Top-5 MLM Accuracy between the three models reveals the significant benefits of pre-training on domain-specific data. Model 2, which is further pre-trained with the ASRS plus NTSB dataset, shows improved performance in predicting aviation-related words compared to the original BERT-Base-Uncased model. The addition of high-frequency words from the ASRS plus NTSB dataset to the vocabulary, as demonstrated in Model 3 (Aviation-BERT), further enhances the accuracy of word predictions. Qualitative assessment through masked token predictions confirms the pre-trained models’ domain-specific understanding, closely resembling the language used in ASRS or NTSB narratives. Notably, these models demonstrate the ability to recognize synonymous entities within the aviation domain and effectively handle different writing styles. This highlights how Aviation-BERT is more desirable over general-purpose language models, which may lack the understanding of aviation-specific terminology.

IV. Conclusions and Future Work

Transformer-based language models like BERT form the state-of-the-art in natural language processing. However, they lack domain specificity. The present work demonstrates an aviation domain-specific BERT model pre-trained using aviation accident and incident narratives from NTSB and ASRS. Aviation-BERT shows an improvement over BERT-base when tested for masked word prediction over aviation domain. It also shows the ability to incorporate aviation-specific context better than BERT-base while showing the capability to handle synonymous entities, different writing styles, and homographs. This preliminary Aviation-BERT model is expected to result in improved performance of downstream tasks like question-answering, named entity recognition, and text summarization. By enabling the automatic processing of aviation text data, Aviation-BERT is expected to improve the processing of millions of aviation safety reports. By extracting safety concepts and their frequencies from safety reports such as NTSB, ASRS, Service Difficulty Reports, etc. Aviation-BERT will help quantify NAS-level safety models like ISAM. While domain-specific pre-training of BERT is common, the authors found very little archival literature that serves as a comprehensive guide on how to do so. Therefore, more important than providing an Aviation-BERT model, the present work seeks to serve as a comprehensive step-by-step guide to pre-train and adapt BERT to domain-specific language.

The bigger the training dataset, the better the contemporary large language models get. Therefore, future work in this area will explore options to incorporate additional aviation-specific text databases to improve Aviation-BERT. Aviation-BERT is expected to be used for numerous downstream tasks like classification and question answering. Additional ablation studies to determine the impact of pre-training options on the performance of downstream tasks will also be conducted and reported in the future.
Acknowledgements

This research was supported in part through research cyberinfrastructure resources and services provided by the Partnership for an Advanced Computing Environment (PACE) at the Georgia Institute of Technology, Atlanta, Georgia, USA. This work was funded by the US Federal Aviation Administration (FAA) as a part of the Top-down Safety Risk Modeling For Facilitating Bottom-up Safety Risk Assessment project under FAA Award Number: 692M15-21-F-00225. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the FAA or other project advisors. The authors would like to thank Huasheng Li, Shane Bertish, Tom Tessitore, Anis Ommmezine, Dereck Wilson, Poornima Shankar, Alan Durston, and Chris Andrziecek for their guidance and support. The authors would also like to thank Maxence Turpin for his help with BERT.

References


[38] PACE, Partnership for an Advanced Computing Environment (PACE), 2017. URL http://www.pace.gatech.edu


