PERCEIVE: PROACTIVE EXPLORATION OF RISKY CONCEPT EMERGENCE FOR IDENTIFYING VULNERABILITIES & EXPOSURES

A DISSERTATION SUBMITTED TO THE GRADUATE DIVISION OF THE UNIVERSITY OF HAWAIʻI AT MĀNOA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF SCIENCE

IN

COMPUTER SCIENCE

MAY 2021

By

Carlos V. Paradis

Dissertation Committee:

Rick Kazman, Chairperson
Daniel Port
Daniel Suthers
Martha Crosby
Misty Davies

Keywords: software vulnerabilities, CVE, safety threats, social network analysis,topic modeling, natural language processing, survey design, natural language understanding, ASRS, aviation safety reporting system

To my family,

For supporting me through humble beginnings to where I now stand.

"Per Aspera Ad Astra"

ACKNOWLEDGMENTS

This dissertation would not be possible without the support of many people, and much of it dates further back than the start of this doctorate. Nearly a decade ago, Rick Kazman agreed to mentor an undergraduate student who wouldn't just stop pestering him about wanting to work together, even if remote. It was also around the same year I had the opportunity to meet Misty Davies, not surprisingly under similar circumstances. It is my immense joy to have both mentors being part of this life milestone.

I am also very thankful to the members of this committee, Martha Crosby, Dan Port, and Dan Suthers, for agreeing to be part of the committee, for challenging the presentation of the work, and the valuable feedback provided since the earlier days of this work and which I had the opportunity to learn as a student. I am equally thankful for my family and friends both in Brazil and in the U.S. who continually provided me with support to pursue this journey, in particular my loving wife, Kathryn Paradis, who, despite being from an entirely different background, could probably explain much of this dissertation work after years of listening to me talk about it! Many ideas, diagrams, and visualizations in this work in text mining are also thanks to discussions with Susanne Still and Lisa Miller, and data pipelines thanks to the ERDL lab working with Eileen Peppard.

Last but not least, I am thankful for living in paradise for over half a decade of my life, the Ohana of the University of Hawai'i at Manoa and Honolulu, and the warm coastal weather which many times served as a reminder of my old home and family. From the Slater family who accepted an abroad stranger to rent a room to the friends I made in IEEE HKN and the university labs, thank you for being part of this journey. I am humbled for the opportunity I have been given, and I can't wait to pay it forward in the next chapter of my life.

ABSTRACT

National databases that collect various kinds of textual threat reports such as ASRS, CERT, and NVD manually process their reports individually. They then offer data products to disseminate the aggregate information, like newsletters, alerts or individual report searching. The goal of this research is to connect these individual reports thematically and temporally to identify emerging or recurring threats, by analyzing large collections of text, source code, collaboration and communication patterns. This capability, I argue, enables us to identify the emergence and recurrence of such themes, and the contexts in which they re-occur, facilitating faster and more capable mitigation. I propose two models to shed light on this goal: An empirical model of vulnerabilities as bugs, the *commit flow model*, and one of the vulnerabilities and aviation safety threats as topics, the *topic flow model*. I use as gold standard existing manual workflows in both domains, reflected in the existing data products by these organizations, and empirically evaluate if the automated models can match or outperform existing manual practices.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Motivation

The timely identification and communication of software vulnerabilities and safety threats have been a key concern of the IT industry and many governments. For example, in the U.S., the National Vulnerability Database (NVD)[1], which uses as data feed MITRE's Common Vulnerabilities and Exposures (CVE) Database[2], and the Aviation Safety Reporting System (ASRS)[3], have existed for decades. We begin by sharing two examples—an aerospace safety threat [5] and a software vulnerability —to motivate the goals and relevance of this dissertation.

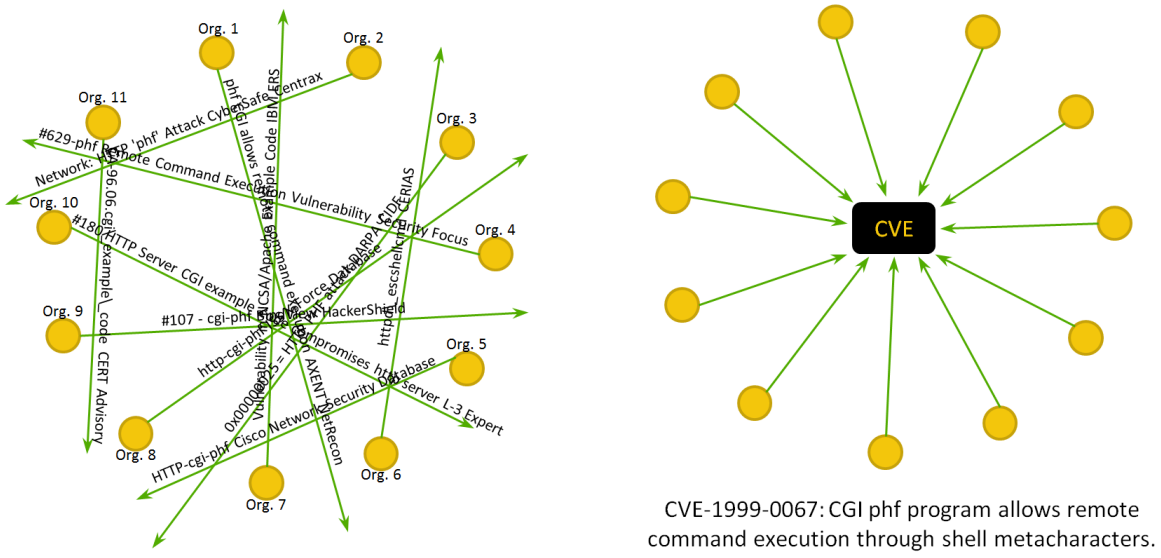
> On December 1, 1974, TWA Flight 514 was inbound through cloudy and turbulent skies to Dulles Airport in Washington, D.C. The flight crew misunderstood an ATC clearance and descended to 1,800 feet before reaching the approach segment to which that minimum altitude applied. The aircraft collided with a Virginia mountaintop, killing all aboard.
>
> A disturbing finding emerged from the ensuing NTSB accident investigation. Six weeks prior to the TWA accident, a United Airlines flight crew had experienced an identical clearance misunderstanding and narrowly missed hitting the same Virginia mountain during a nighttime approach. The United crew discovered their close call after landing and reported the incident to their company. A cautionary notice was issued to all United pilots.
>
> Tragically, at the time, there existed no method of sharing the United pilots' knowledge with TWA and other airline operators. Following the TWA accident, it was determined that future safety information must be shared with the entire aviation community. Thus was born the idea of a national aviation incident reporting program that would be non-punitive, voluntary, and confidential.[4]

The example above, and others like it, motivated what eventually became ASRS. Now consider the following vulnerability example. In the "about us" page[5] of CVE Mitre, one can find a gif image that illustrates several different organizations communicating about *the same* software vulnerability to one another in two time periods, labeled "Before CVE" and "After CVE". We display a simplified version in Figure 1.1.

What is common to both examples is that ASRS and CVE Mitre inception averts a babel tower scenario, i.e., it serves as a method of sharing expert knowledge beyond the boundaries of the organizations.

Both ASRS and CVE are unstructured text repositories with metadata and have, over the years, accumulated valuable incident and vulnerability information. As unstructured text datasets

[1] `https://nvd.nist.gov/`
[2] `https://cve.mitre.org/`
[3] `https://asrs.arc.nasa.gov/`
[4] `https://asrs.arc.nasa.gov/publications/callback/cb_317.htm`
[5] `https://cve.mitre.org/about/index.html`



(a) Before CVE

(b) After CVE

Figure 1.1: Before and After CVE

increases, new opportunities to leverage this dataset also emerge. Less than two years after its inception, an ASRS quarterly report emphasized the importance of analyzing the data *over time*. ASRS designers were acutely aware of the potential value of data derived from individual occurrences in highlighting deficiencies and discrepancies across the national aviation system. They also believe that other, perhaps more valuable, insights into system problems could be gained *only by studying a large body of incident data* [8]. However, as of July 2019, ASRS has processed more than 1.6 million reports over its 42-year history [76, p.13], making the analysis of a large body of incident data a daunting task.

The same dramatic increase of data as well as the perceived value to understand emerging threats can be observed in the CVE repository of software vulnerabilities. In March 2014, security researchers found a catastrophic vulnerability in OpenSSL[6], one of many source code projects, later published by the NVD as CVE-2014-0160 and named Heartbleed. Heartbleed allowed attackers to read sensitive memory from vulnerable servers, potentially including cryptographic keys, login credentials, and other private data, and it was a simple bug for hackers to understand and exploit [18]. Version 1.0.1 of OpenSSL added support for the heartbeat functionality and enabled it by default, thereby inadvertently making the implementation vulnerable by default. This vulnerability remained until 1.0.1g or roughly two years later [12]. Heartbleed is one of the over 140,000 vulnerabilities identified at the time of this writing, according to the Common Vulnerability Enu-

[6] `https://github.com/openssl/openssl`

meration (CVE)[7], the source of NVD vulnerability enumerations. While the number of indexed vulnerabilities is not nearly as large as ASRS reports, they are still large enough to prevent manual analysis from being performed on a regular basis.

To add value to their ever-increasing databases, both ASRS and NVD generate data products for the community that provide the reports to regulatory agencies, leading to a sustainable cycle. ASRS, for example, provides a report query capability, issues alerts to organizations, performs searches about a topic, publishes newsletters, etc. NVD provides additional metadata through severity scores and mitigation strategies. There are also other taxonomies, such as the Common Weakness Enumeration (CWE)[8] that further abstract CVEs into weaknesses, which are groups of vulnerabilities that are conceptually similar.

Both ASRS and NVD, therefore, serve as continuously evolving knowledge bases for their respective communities. Despite the perceived benefit that automation could lead to these organizations, they still rely mostly on manual effort to leverage their data sets. Indeed, much literature has focused on addressing aspects of automation. Still, little has been adopted by these organizations, especially concerning the emergence of vulnerabilities and safety threats.

Automation could also be leveraged in extending the textual analysis of software vulnerabilities by mining software repositories. Because vulnerabilities are reported in open source projects like OpenSSL, the project's development process and source code are available for inspection, providing another dimension to understand software vulnerabilities as bugs from a software development standpoint, instead of only knowledge aggregation.

While existing data products already provide value, *they are still seen in large part as isolated records*. Given all the available data, it is clear that more could be done to assist in the understanding of vulnerabilities and threats beyond manual inspection by analyzing how their content *evolves*.

1.2 Thesis Goal

Current data products only leverage individual reports. **We hypothesize that by analyzing source code and textual corpora and associated data/metadata over time, we can automatically identify topics and trends that provide value for the users of these corpora in terms of quickly and capably identifying emerging threats**.

Specifically, the goal of this thesis is to capitalize on available but yet not used source code repositories and databases of vulnerabilities and threat data so that we can connect individual vulnerabilities and threats temporally. This capability enables us to identify their emergence, recurrence, and the contexts in which they re-occur. This modeling, we argue, is a precondition for faster and more capable mitigation. We propose two models to shed light on this goal: An empirical

[7] `https://cve.mitre.org/`
[8] `https://cwe.mitre.org/`

model of vulnerabilities as bugs, i.e. the *commit flow model*, and one of vulnerabilities and threats as topics, the *topic flow model*, building from and improving prior literature [14, 71, 21, 32, 78, 50]. We have named the framework that these two models provide PERCEIVE: Proactive Exploration of Risky Concept Emergence for Identifying Vulnerabilities & Exposures.

We found, when modeled as topics, the manual analysis bottleneck of software vulnerabilities shares nearly identical motivations and needs as that of safety threats. This has motivated us to generalize and refine a model of how we analyze them. And, we hypothesize, this model would be appropriate for other domains that share similar concerns of broadly collecting and disseminating risk.

1.3 Outline

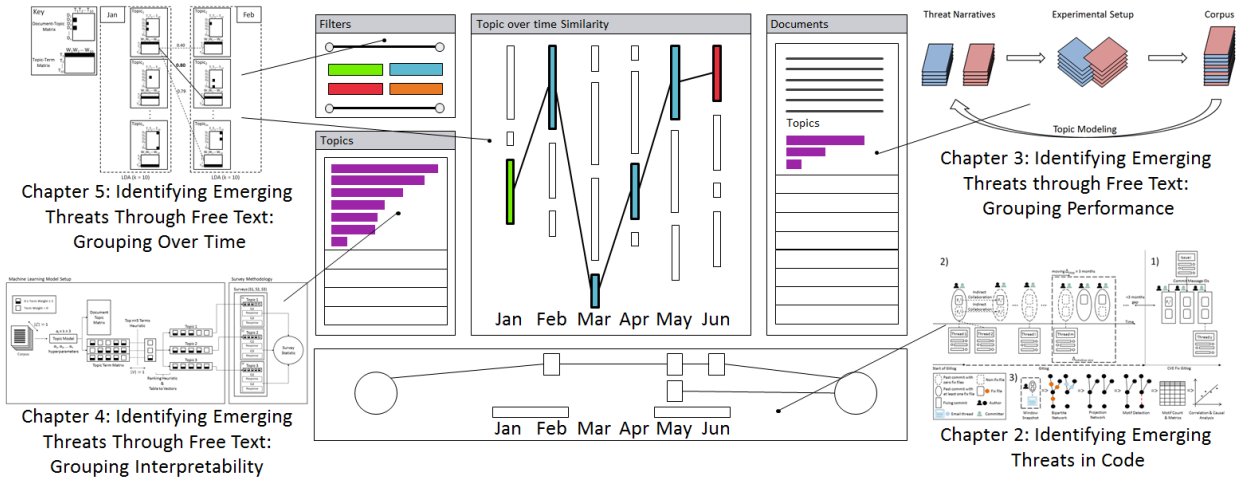


Figure 1.2: Thesis Outline [14, 71, 21, 32, 78, 50]

.

A visual representation of the thesis outline is presented in Figure 1.2. The diagram presented in the center, with the exception of the bottom pane, is based on the concept of "topic flow" [71]. We chose this representation because many of the chapters in this dissertation were motivated by the value and the limitations of this tool. The focus of the thesis, however, is not on tools but rather on developing a methodology that can deliver operationally relevant results. Using Figure 1.2 as a reference, the remainder of this thesis is as follows:

In chapter 2, we explore the first of the two models by modeling vulnerabilities as bugs. We start by presenting a qualitative study of the chosen dataset, OpenSSL, to choose what features could be used to observe emerging vulnerabilities in the community. The conclusions are then formalized in a quantitative method, Commit Flow.

In chapters 3, 4, and 5 we model vulnerabilities and threats in free-form text by defining topic

flow. Differently from the previous chapter 2, where we define a notion of time in a single chapter through commit flow, we separate the evaluation of topic in three chapters.

In chapter 3, we perform two studies, one for vulnerabilities and one for threats, to evaluate the automated identification of groupings from free-form text at a single point in time. In 4, we evaluate the human comprehension of the derived groupings. Finally, in the last chapter 5 we combined learned lessons and methods to create a notion of topic evolution over time, capturing emerging threats. Together, the two methods, as shown in Figure 1.2, realize PERCEIVE, the goal of this dissertation.

1.4 Contributions

The main contribution of this dissertation is a holistic framework to leverage textual knowledge bases to identify emerging threats, as shown in Figure 1.2. Different from prior work, which relies mostly on one data source, one method, or focuses on a small feature of a dataset (e.g., severity score of a vulnerability), we observe the problem from an operational standpoint. Specifically, rather than leveraging only source code and free text, we approach the problem by evaluating what data captures the phenomena of interest, i.e., software vulnerabilities and safety threats, and can lead to operationally relevant results in these and similar organizations that retain knowledge bases. We elaborate on the specific contributions compared to the existing literature in the following chapters after presenting our research questions and methods.

CHAPTER 2
IDENTIFYING EMERGING THREATS IN CODE

This chapter is derived in part from the IEEE Transaction in Software Engineering journal "In Search of Socio-Technical Congruence: A Large-Scale Longitudinal Study".

In this chapter, we begin exploring how to identify emerging threats, in particular software vulnerabilities. To identify software vulnerability emergence as bugs, we hypothesize, based on existing literature [78, 62], the socio-technical behavior of a project's community has implications on emerging software vulnerabilities. Our goal, therefore, in this research was to determine the extent to which developer social interactions—communication (as determined by e-mail exchanges) and collaboration (file changes by multiple contributors)—can positively or negatively affect the security properties of a codebase. To examine this question, we analyzed the code and the social networks of a prominent open source project—OpenSSL—over a 20-year time span to understand their relationships.

To better motivate our method beyond related literature and theory, we begin this chapter with a socio-technical qualitative case study of OpenSSL. Later in the chapter, we then capture this understanding in commit flow, the proposed method of this chapter, using socio-technical metrics.

We chose OpenSSL as its past software vulnerabilities, such as the infamous Heartbleed bug, had enormous repercussions. We performed content analysis on OpenSSL to understand how various technologies affected community behaviors; and understand if these behaviors contributed to vulnerabilities. We defined five research questions and used both social network analysis and metrics to answer them. Our primary results are: 1) that attempts to provide an informal mailing list infrastructure did not drive community engagement and adoption, but the introduction of an issue tracker did; and 2) that the OpenSSL architecture contains substantial technical debt, which makes software vulnerabilities more likely to be introduced. In the latter portion of the chapter, we complement the analysis and show that socio-technical problems in OpenSSL consistently preceded the introduction of vulnerabilities using commit flow.

2.1 Introduction

Despite improved tooling to detect software vulnerabilities, these bugs continually plague the IT landscape. Much research has suggested that software architecture complexity is a key contributor to the introduction of vulnerabilities [36, 22]. In this work, we examine a related question—looking at the extent to which developer communication (e-mail exchanges) and collaboration (file changes by multiple contributors) can positively or negatively affect a code base. Thus we are

interested in examining the relationships between the network of code artifacts (the architecture) and the network of developers and how this relationship may mediate the introduction and detection of vulnerabilities. Developer collaboration can be mediated by community infrastructure by, for example, introducing capabilities to improve community engagement such as mailing lists. The complexity of the code base, in turn, can affect developer collaboration processes, e.g., by being increasingly difficult to change, eventually leading to software vulnerabilities, such as the Heartbleed bug.

To better understand this socio-technical relationship, we first used content analysis [35, 31] to identify decisions within the OpenSSL project[1] starting from its first release. Specifically, we looked for decisions that may have affected community interaction. We have derived three research questions to capture and evaluate the major decisions that may have affected the project's socio-technical ecosystem and subsequently led to software vulnerabilities. To answer these RQs, we modeled these interactions using architectural and social network analysis [87].

To aid reproducibility, we reference throughout our work 36 artifacts, which we refer to with letters followed by numbers in this work. These artifacts include e-mails from OpenSSL mailing list, which begin with the letter M, blog posts (B), and commits (C). For example, to provide evidence of an event announced in the mailing list, we may refer to the e-mail as [M1]. We included the citations in the Appendix A.

Our research questions, derived from the content analysis, are as follows:

> RQ1: Do community [STATUS] threads drive community engagement?

In the early days, we observed that OpenSSL provided status e-mail threads in its mailing list. We see this as a conscious intervention on the part of OpenSSL's core developers to maintain community engagement. These e-mails had a well-defined structure and contained several features which resembled a modern issue tracker, as seen in Figure 2.5.

> RQ2: Did the community actually adopt the Request Tracker after its inception?

A second intervention made by OpenSSL was the introduction of a Request Tracker[2], which coexisted with the mailing list until both were eventually replaced by the project's move to GitHub in 2016-2017. Because this intervention may have had implications on the contributions to the code base, we also investigate it as a research question. RQs 1 and 2 were also of interest to OpenSSL core developers: in a blog post announcing the change, they emphasized their desire to encourage engagement [B7].

[1] `https://www.openssl.org/`
[2] `https://bestpractical.com/request-tracker/`

RQ3: Did the architecture practices of OpenSSL lead to major problems for project success?

In RQ3, attempt to discern the possible contribution of architectural complexity to vulnerabilities, as prior studies have hypothesized [36, 22].

RQ4: Were there socio-technical problems that consistently preceded the introduction of software vulnerabilities?

In RQ4, we attempt to determine whether socio-technical problems—social smells—, defined in the subsequent method section, were correlated with the introduction of vulnerabilities. We prosecute this question via a correlation analysis.

RQ5: Are there causal links between socio-technical smells and the resolution of software vulnerabilities?

In RQ5, we ask a deeper question: can we find causal links between socio-technical problems and software vulnerability-related development activities (both vulnerability introduction and resolution). To analyze this question, we turn to causal modeling [23]. While this sounds like an extreme claim, surely *something* causes every bug to occur—perhaps a communication breakdown or lack of programmer knowledge or skill. A major contribution in this paper is to deeply examine the former sorts of root causes as these are generally less well understood or explored in the software engineering research literature.

Our contributions of this chapter are as follows:

- A study of software vulnerabilities, similar to [36] which analyzed Google Chrome. Extending prior work, we have included content analysis and use it to derive our RQs.
- Compared to [36] and [22], we analyze not just file dependencies but also communication and collaboration networks as a lens to understand software vulnerabilities as a single network.

2.2 Related Work

This section briefly surveys research focusing on the socio-technical relations between software processes, products, and people with respect to (cyber-)security issues manifesting.

Our work can be seen as an extension of [78, 32, 21, 36]. For step 1 of Figure 2.1, we adapted the work of [21] to create timelines of issues. Step 2 —slicing a project's history into 3-month time windows has been done in [32], among others. In step 3 we capitalized on the idea that social smell

[78] metrics can be calculated from window snapshots. We performed a case study on OpenSSL, similar to the work of [36] which studied Chromium for software vulnerabilities.

The Heartbleed bug in particular, and the OpenSSL project in general, have received an enormous amount of scrutiny, e.g. [89] and [12]. These papers look at the causes of Heartbleed in terms of poor programming practices, and discuss how better testing and analysis approaches might have helped. In [18] the authors analyze the impact of Heartbleed on the world.

Several studies have examined the effects of architectural decay and community decay on project success, such as [77] which looked at the decline of SourceForge.

Social smells reflect recurring sub-optimal organizational structure conditions (i.e., patterns). Under these conditions, additional friction is added to functional software teams connected to nasty organizational behavior, e.g., sub-optimal knowledge sharing, recurrent sharing delays, misguided collaboration, and more. In the past, researchers tried to identify the recurrent conditions around community smells for the benefit and use of software practitioners [80, 81].

2.3 Method

This section describes the analysis techniques employed to understand the nature of the OpenSSL developer community, their interactions, and the resulting output in connection with the cybersecurity conjectures we introduced earlier.

2.3.1 Dataset

OpenSSL Mailing List

We analyzed over 20 years of OpenSSL's project history, beginning in 1997 after its first release. This presents challenges as, in longer time periods, we must account for transitions in the project's infrastructure (such as changes in mailing list technologies and the introduction of a Request Tracker, which impacted identity matching). In addition, OpenSSL only began utilizing GitHub for issue management and communication in 2016[3].

OpenSSL maintains several mailing list archives[4]. OpenSSL maintains a developer list covering a period from 2014 to 2018 (hosted using mailman), as well as three other archives, as noted in their website `https://mta.openssl.org`: Marc (1998-2018), The Mail Archive (2016-2018), and Google Groups (2002-2018). We chose to use Google Groups to obtain the mailing list archive, as Marc discourages crawlers, and the other sources provided too little data. Unfortunately, Google Groups redacts e-mails, which makes matching authors with different names and e-mails more challenging. However, as we note in the identity match section 2.3.3, using e-mails for matching would lead to a greater threat to validity.

[3] `https://www.openssl.org/blog/blog/2016/10/12/f2f-rt-github/`
[4] `https://www.openssl.org/community/mailinglists.html`

OpenSSL Version Control System

We obtained the git log from OpenSSL's GitHub repo[5]. To analyze RQ3, which relies on static analysis of OpenSSL's code, we took snapshots (via "git clone") of the repository at points of interest on the project's timeline, as noted in the discussion of RQ3.

To identify the security bugs in OpenSSL, we relied on the project's use of CVE numbers. CVE—Common Vulnerabilities and Exposures—is a widely adopted numbering system used to catalog publicly acknowledged vulnerabilities. In our work, we parsed CVE IDs directly from OpenSSL's commit messages. While this may pose threats to validity in terms of whether developers properly annotate vulnerability fixes, we believe that this risk is minimal, as vulnerabilities and associated commit hashes are posted on the OpenSSL website[6] [B23].

CVE and CWE Data

CVE—Common Vulnerabilities and Exposures—is a widely adopted numbering system used to catalog publicly acknowledged vulnerabilities. As part of the effort to model and analyze CVEs, they have also been grouped into CWEs—Common Weakness Enumeration. In our work we parsed CVE IDs directly from OpenSSL's commit messages. CVE - CWE linkage were obtained from the National Vulnerability Database[7].

2.3.2 Content Analysis

We manually inspected the early e-mail threads of OpenSSL for insights into the communication and collaboration processes among contributors, starting from the first e-mail when the project began in 1998. The OpenSSL blog was used to elicit a set of named vulnerabilities and to validate our findings.

2.3.3 Pre-Processing

File Filters

We filtered the files that we extracted so that we only analyzed source files with extensions .cpp, .c, h., .java, .js, .py, and .cc. This limitation is consistent with the tooling that we use to analyze architectural complexity and architectural flaws: DV8[8] and Depends [9] and covers about 98% of the source code files of OpenSSL.

[5] `https://github.com/openssl/openssl`
[6] `https://www.openssl.org/news/vulnerabilities.html`
[7] `https://nvd.nist.gov/vuln/data-feeds`
[8] `https://archdia.com/`
[9] `https://github.com/multilang-depends/depends`

Identity Matching

A critical component of conducting socio-technical analysis in open source communities is assigning a consistent identity to users who may employ multiple variants of their name and e-mail addresses in their project interactions. Several approaches to match identities have been proposed (e.g. [9], [92]). Exact name matching (either names or e-mails) or partial matching (e.g. based on edit distance) are two commonly used schemes.

In this work, we chose to use name matching only. Still, our decision is based on our qualitative analysis of OpenSSL and not based on precision and recall accuracy results (we will provide more details in Section 2.4.1 when we present the content analysis results). Our identity matching was designed as a 3 step pipeline: formatting, name-email separation and pair-wise matching. Formatting includes the removing symbols such as '< >', commas, or replacing 'at' to '@', while avoiding modifying a name, such as Matt. Name and e-mail separation handles cases where first or last or both names are not provided, multiple word name, etc. Finally, pair-wise matching handles mostly comparisons of name and e-mail, or reverse names.

When comparing names and e-mails, we observed that the Request Tracker would also replace the e-mail domain of developers in the mailing list with the request tracker. This, in essence, duplicated the network of combinations of names and e-mails. In several cases, the e-mail of core developers was consistently the same, effectively leading to collapsing all core developers under the same identity. Because of this, and considering that Google groups already redacts e-mails, we deemed it appropriate to use name matching only.

In total, the steps of formatting, name separation, and name matching amounted to 31 test cases, which were successfully implemented in our R package. At the end of this step, users in both the version control system and mailing list who matched via the tests we implemented were assigned an appropriate ID.

2.3.4 Data Modeling - Commit Flow

To address RQs 4 and 5, we needed to analyze the CVE-related commits to OpenSSL, and its developer social network, over the project's entire history. For this purpose we created a technique and tooling that we call *Commit Flow*.

In Figure 2.1, we summarize the major steps of our data extraction and analysis pipeline, namely: 1) identification of a set of files in a CVE-resolving commit, 2) mining prior changes to these CVE-resolving files over a set of time windows' snapshots, and 3) analyzing each time window's snapshot to extract metrics that allow us to determine social network motifs indicating collaboration without communication. This "Commit flow" technique is based on [32, 21, 78]. The pipeline is implemented in an open-source R package, Kaiaulu[10]. The remainder of this section explains the various steps of Figure 2.1.

[10] `https://github.com/sailuh/kaiaulu`

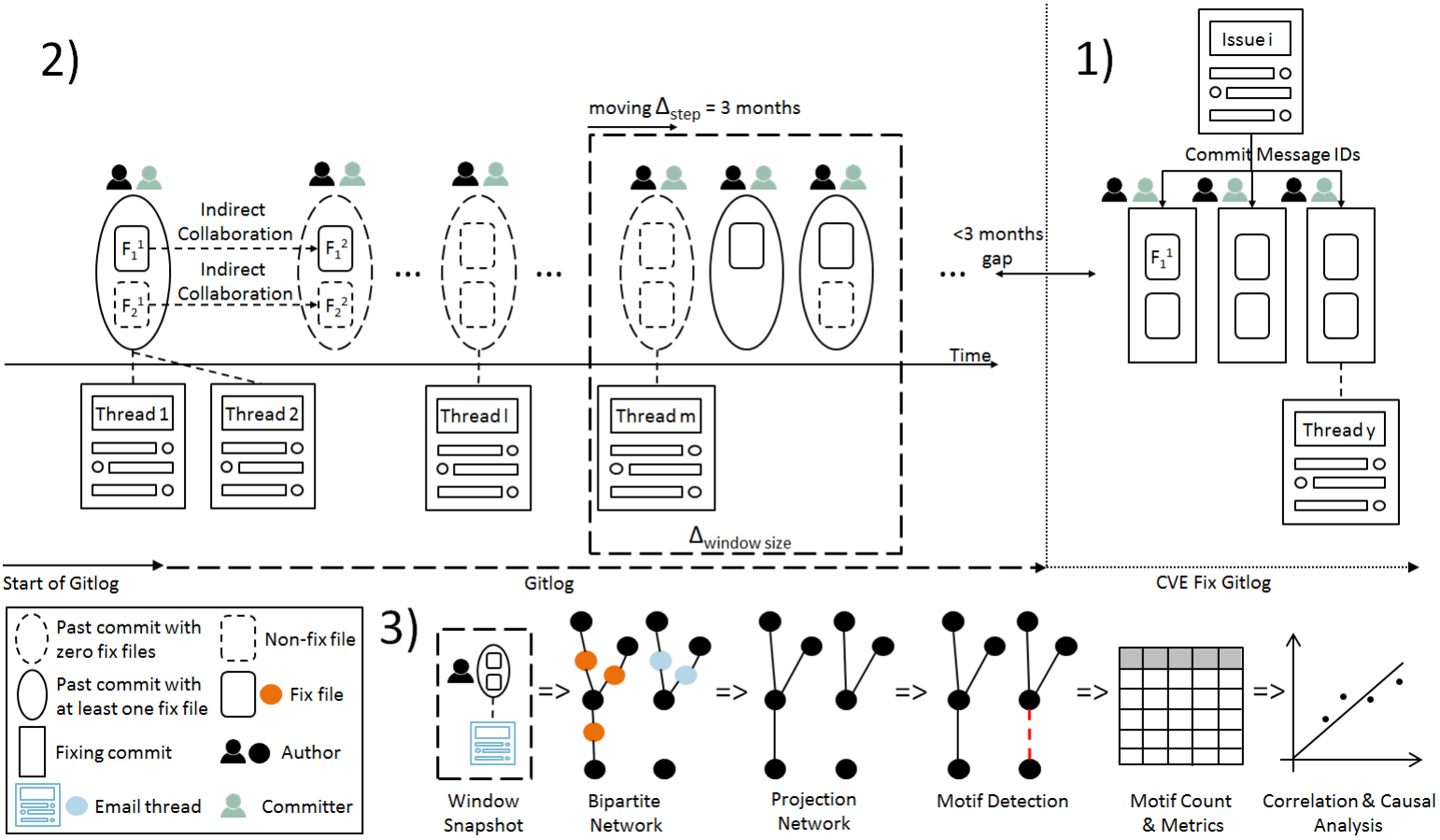


Figure 2.1: Commit Flow method for a single issue.

Step 1). The first step of the commit flow takes, as input, a regular expression (regex) that identifies the issue labels of interest from commit messages in the git log; in this work, the regex was *CVE-**, as OpenSSL uses this regex when **fixing** a CVE in the source code. This, therefore, allowed us to extract the set of CVE-fixing file changes from the project git log, which may be contained in one or more commits annotated with the same CVE ID. Next, we extract all unique file names for each CVE's fixing commits. We call these the CVE_i**-fixing files**.

Note in this definition, the same file may have been modified at a different point in time to address a different CVE. We will later present the topology of files' relationship to CVEs in the results section as a network to better understand this relationship. More importantly, each CVE's resulting timeline is guaranteed to include the set of file changes that introduced the vulnerability.

Step 2). In the second step, we use the CVE_i**-fixing files** from Step 1 to obtain multiple CVE timelines. For each set of CVE_i**-fixing files**, we subset the entire git log, **prior to the** CVE_i**-fixing file changes**, to obtain the CVE_i**-precursor file changes**. There may be years worth of a given CVE's precursor file changes, as they include all changes until each file was created. We are interested in the precursor file changes of a given CVE because we are interested in the factors that preceded the vulnerability. In Figure 2.1, the set of CVE_i**-precursor file changes** are shown above the timeline. The CVE_i**-fixing file changes** is shown separately to the top right of the Figure. Similar work to this step is by [21], where the authors use "seed files" to construct a precursor timeline.

Once again, we remark that, since the same file may be fixed at different points in time to address different CVEs, it will be true in this case a subset of past file changes will be shared between two or more different CVE timelines that include the same file. This, however, does not affect our analysis, which considers each CVE timeline the unit of analysis.

Another dimension of the data, and of importance to the analysis in this work associated to file changes is that they are written by **authors**, and approved by **committers**. Authors may be committers if they are able to push code to the repository. Committers may also be authors if they authored the change. While we described how we obtain the CVE_i**-precursor file changes** in Figure 2.1, we have not yet made explicit how the authors of *consecutive* file changes "collaborate". In [32, 78] the authors consider that git authors who modified the same file consecutively are **indirectly collaborating**. In the scope of this work, we consider only authors, and authors who are committers due to authors playing a more significant role in how we define indirect collaboration. How we precisely define indirect collaboration influences the interpretation of the results once we represent it in a social network. Therefore, before we discuss it further, we first explain the network construction method.

For each CVE timeline, the precursor file changes are discretized in 3-month time windows. The choice of 3-month time windows is due to related work finding file changes to be quasi-stationary in this time period [32]. Using the start and end date of each 3-month time window, we identify

developer mailing-list activity, specifically, developers communication using e-mail threads. We include communication data here because our socio-technical assumption relies on the dynamics of collaboration and communication.

In figure 2.1, Step 2, the mailing list data is shown below the timeline, and the dashed rectangle captures the 3-month time windows. Intuitively, the process can be imagined as follows: Given a CVE_i precursor file changes (which defines a timeline), a 3-month window starts at the first CVE precursor file change, takes a snapshot, then moves forward to immediately after where it ended (so there is no data overlap between time windows), and the process repeats until all precursor file changes are accounted for. Each snapshot includes both the precursor time changes and the associated mailing list, and at the end, we can define the precursor file changes of each CVE as a set of snapshots augmented with mailing list data.

The process may result in snapshots of four types, where git log and mailing list data are available within the 3 months, one is missing, or both are missing. Now that each CVE's precursor file changes is partitioned in snapshots containing mailing list data, we describe how we process each snapshot individually in Step 3.

Step 3. In the third step, we have a sequence of transformations each snapshot goes through. We explain each of the transformation steps in detail in the following subsections.

Step 3 - Network Construction

In the first transformation, a socio-technical network is formalized for each snapshot by a graph $G_{st} = (V, E)$, where the set of nodes $V = V_a \cup V_f \cup V_t$ comprises authors V_a, source files V_f and e-mail threads V_t. The set of edges E = $E_{comm} \cup E_{chg}$ models communication and collaboration between authors, where communication is done via $E_{comm} \subseteq V_a \times V_t$, and file changes via $E_{chg} \subseteq V_a \times V_f$. Observe by this construction, the socio-technical network is in fact two bi-modal bipartite networks $G_{st} = G_{chg} \cup G_{comm}$. Both G_{chg} and G_{comm} are also weighted (representing an author's count of changes to a file within a 3-month time window, and the number of replies submitted to an e-mail thread respectively), and undirected (the direction is irrelevant in this case because it could only go in one direction in each bipartite network). In Figure 2.1, step 3, we can see at the bottom both bipartite networks represented separately by black vertices (authors), and different color vertices (files and e-mail threads).

Step 3 - Indirect Collaboration and Communication Networks

In Figure 2.1, Step 3, we are now concerned on moving from bipartite networks to uni-modal networks of authors. This is because the socio-technical metrics we will calculate depend on comparing in various forms if the collaboration network mirrors the communication network and vice-versa.

Earlier in Step 2, we mentioned the consecutive same file change by different authors is defined as indirect collaboration. Indirect collaboration played a important role in the interpretation of the

results. This occurs in this transformation step, and we elaborate on it here.

In Figure 2.1, we show we obtain the uni-modal network via a graph projection operation. Intuitively, the projection operation eliminates the 'colored' nodes, and connects the adjacent black nodes together, where the resulting edge weight is the sum of the eliminated edges. Indirectly, this definition tells us something about our assumptions regarding indirect collaboration. For example, if five authors modify the same file within a snapshot, then the projection operation tells us we *assume all five authors indirectly collaborated*, regardless of the order in time of changes. Note the derived uni-modal networks are un-directed. We define this as the projection transformation to go from bi-modal to uni-modal networks.

Let's now consider a second approach to obtain uni-modal networks. In [32], the authors define one method to construct uni-modal networks from the same data by defining indirect collaboration using the notion of incremental contributions through the commits' timestamps. For example, if author A modifies a file, and the immediately following change to the same file is performed by author B, then B is said to have **indirectly collaborated with A**. A similar intuition and transformation could be used to e-mail replies. We define this method to go from the bipartite network to the uni-modal network a *temporal* transformation (as it relies on the timestamps). Observe in this case, the derived uni-modal networks will be directed graphs (which indicate the flow of time). The edge's weight is defined as the sum of lines of code added by both developers in their respective file changes.

Which method to derive uni-modal networks should we choose? Comparing the same example of both networks reveal a pattern of under or overestimating indirect collaboration, as shown in Figure 2.2. As it is shown, the projection approach will generate more edges when compared to the temporal approach.

Another consideration that must be taken into account in this decision is the entity used to derive indirect collaboration. Consider now Figure 2.3. As we can see, the choice of granularity will also affect the number of edges generated, where a file granularity generates more edges than function granularity. A larger number of edges, in turn, may impact the social smell metrics, as the existence of connections between developers in one network, and the absense of edges in another network, may inflate the number of social smells. The authors in [32] used a combination of both temporal transformation and function granularity. In Kaiaulu, the tool which implements commit flow, we implemented both the file granularity, and generalized the function granularity to *entities*, where an entity can be any source code block region of interest (e.g. functions, classes, or language specific features like structs in C). We chose to use file granularity and projection. Our decision on file granularity is that, from an architectural standpoint, which we include in the analysis of this work and the authors in [32] do not, architectural changes are more well understood, and hence facilitate the evaluation of the method proposed here, commit flow, with more known quality metrics (e.g. churn is commonly defined at file level in the literature). We chose the projection

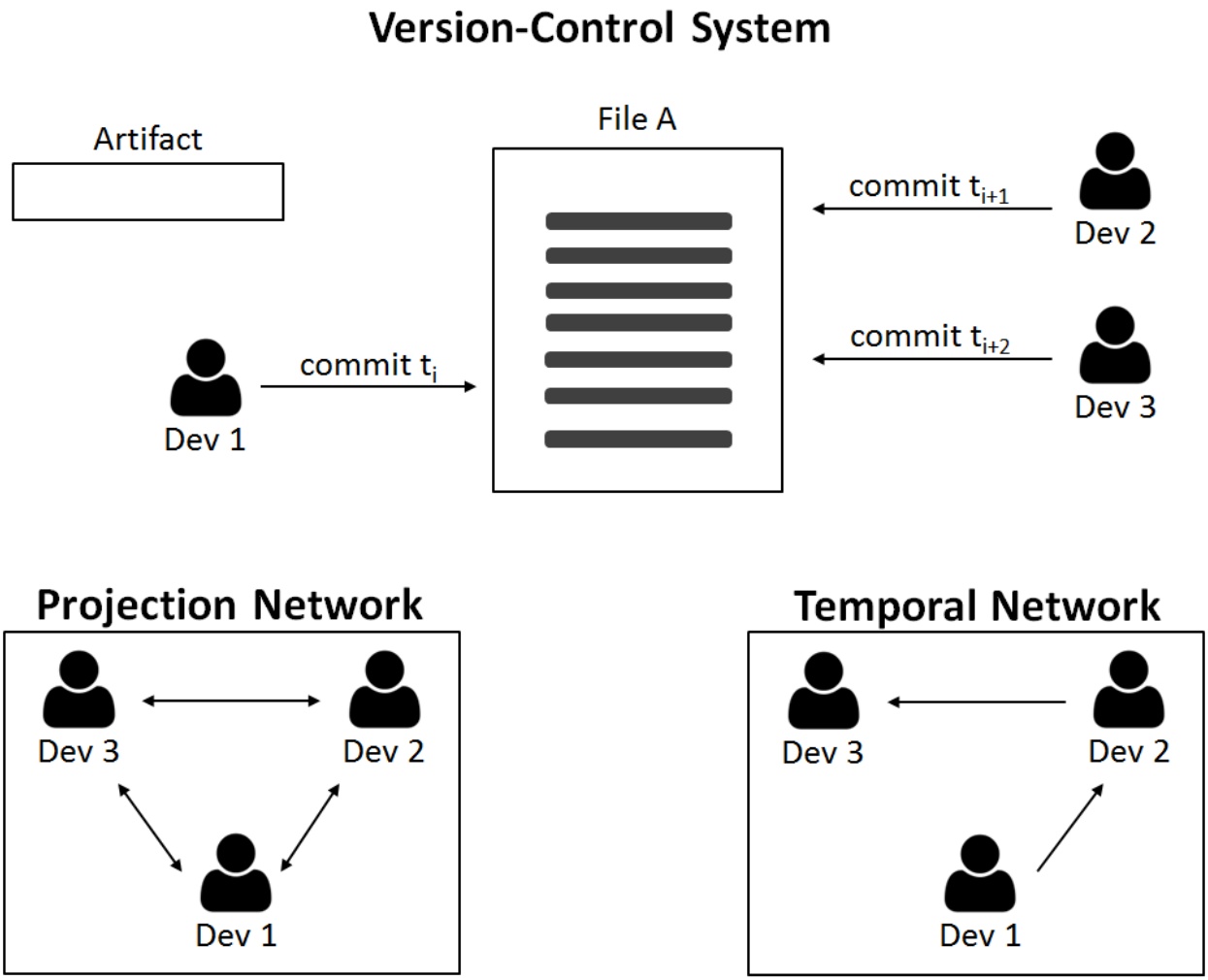


Figure 2.2: Temporal vs Projection Networks, format adapted from [33].

.

transformation over temporal, because we believe that given our 3 month time window due to the quasi-stationary assumption, it is reasonable to assume at file-level all authors had to some degree understand each other's recent changes.

Motifs and Metrics

Having defined the method to obtain the uni-modal networks, what is left is then to compute socio-technical metrics which capture our intuition of communication without collaboration for each snapshot uni-modal networks. We borrow the definition of social smells from [78].

Social smells reflect recurring sub-optimal organizational structure patterns connected to organizational behavior patterns, e.g., sub-optimal knowledge sharing, recurrent sharing delays, misguided collaboration and more. We chose to use 3 of these smells—Organizational Silo, Missing Link and

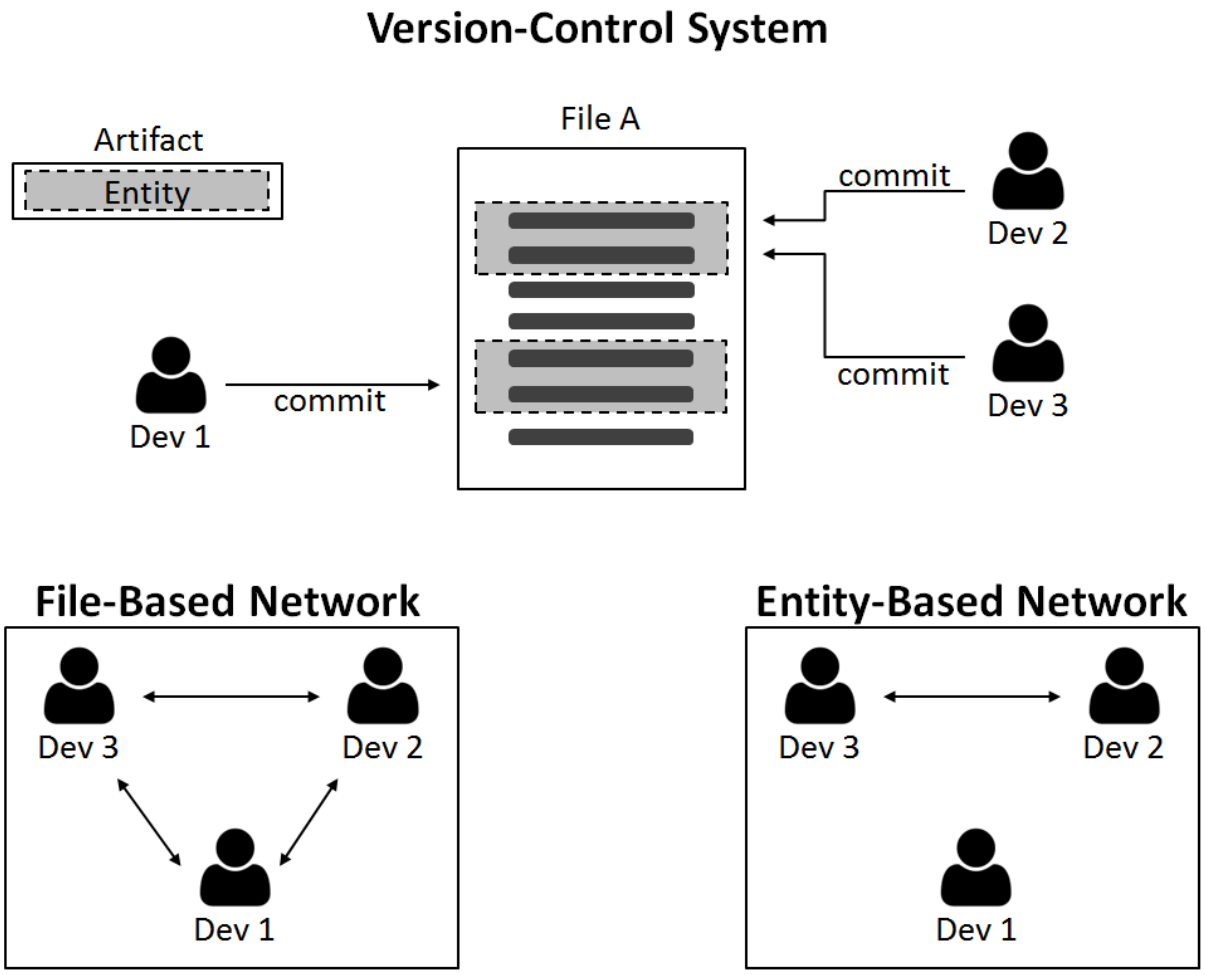


Figure 2.3: File vs Entity (e.g. Function) Networks, format adapted from [33].

.

Radio Silence—and two related metrics: socio-technical congruence and missing communicability [78]. The theoretical choice of social smells to inspect the past of CVE timelines was motivated by Design Rule Theory [36], which argues that problems in design (vulnerabilities in this case) can be resolved by communication and interfaces. We explain in more detail one of the social smells for completeness, but defer the additional details of these metrics can be found in [78].

We depict the motif identification step shown in Figure 2.1 (red line, bottom right comparing the projected networks) in Figure 2.4, we depict in more detail the step from the method for the missing link social smell defined in [78].

In Figure 2.4, the collaboration network projection is shown in green to the left. The communication network is shown in blue to the right. Recall the intuition behind the networks is developers who changed the same file in a snapshot, and communicated via a common e-mail thread in the mailing list. In this 3 month time window, we can see to the left highlighted in red that developers

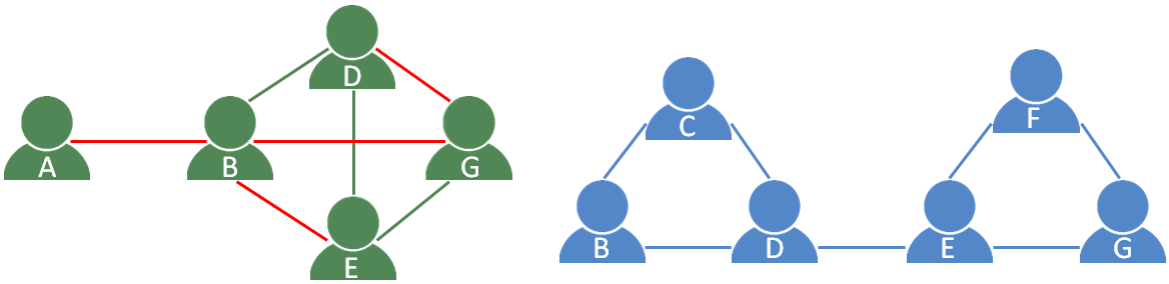


Figure 2.4: Missing Link Social Smell [40]

(A,B), (B,E), (B,G), and (D,G) collaborated (i.e. they have an edge in the green graph), but do not communicate (they do not have an edge in the blue graph). Therefore, the social smell missing link is counted 4 times for this snapshot. Note the green network edges (B,D), and (E,G) are not considered a social smell, because they occur in the blue graph.

We also included three project "outcome" measures for each 3-month snapshot: number of git log authors (code_dev), number of commits, and churn (committed lines of code). These are all measures of the amount of work required to resolve the vulnerability.

In total, we use the communication and collaboration structure of each 3-month time window to derive networks and compute social-network metrics which capture dynamics of collaboration without communication: # of developers, the sum of code churn, # of commits, # of mailing list contributors, socio-technical congruence, communicability, # of org silo motifs, # of missing link motifs, # of radio silence motifs, and # of files committed.

Correlation and Causal Analysis

We measured the correlation between the social smells and the outcomes variables and we performed a causal analysis to answer our research questions. Causal Discovery is the analysis of a dataset to construct a graph describing the data-generation process, that is, the causal mechanisms by which the dataset might have been created ([65], [73], [52], [29], and [23]). To perform causal discovery, we used the Fast Greedy Equivalent Search algorithm [66], Tetrad, version 6.8.0.

2.3.5 Data Modeling - Exploratory Networks

CVE and CWE Issue Network

In Step 1 of commit flow defined in this method section, we identify CVE as issues and construct timelines for each CVE using commit flow. We also noted that files may participate in multiple CVEs, and CVEs may contain multiple files. To better visualize this relationship to answer RQ3, we define a graph of file and CVEs.

$V = V_a \cup V_f \cup V_t$ comprises authors V_a, source files V_f and e-mail threads V_t. The set of edges E $= E_{comm} \cup E_{chg}$ models communication and collaboration between authors, where communication is done via $E_{comm} \subseteq V_a \times V_t$, and file changes via $E_{chg} \subseteq V_a \times V_f$.

Formally, we define the graph $G_{cve} = (V, E)$, where the set of nodes $V = V_f \cup V_{cve}$ comprises source files V_f and CVEs V_{cve}. The set of edges E $= E_{commit}$ models dependencies between source files and CVEs based on their match to the regex "CVE-*" in commit messages in the git log. The weight of the edge is defined by the count of regex matches in the commit messages. The CVE graph G_{cve} is therefore bi-modal, weighted, and not directed.

We also create an additional graph with CWE information, which groups CVEs, to identify if CWEs may indirectly connect vulnerabilities across files not linked by CVEs.

Formally, we define the graph $G_{cwe} = (V, E)$, where the set of nodes $V = V_f \cup V_{cve} \cup V_{cwe}$ comprises source files V_f, CVEs V_{cve} and CWEs V_{cwe}. The set of edges E $= E_{commit} \cup E_{nvd}$ models dependencies between source files, CVEs and CWEs. The edges between CVEs and CWEs E_{nvd} are obtained from NVD, which security experts manually groups new CVEs under existing or new CWEs.

The edges between CVE and CWEs E_{nvd} are unweighted, as CVEs are not assigned to the same CWE multiple times or vice-versa. The source code, CVE and CWE graph G_{cwe} is therefore tri-modal, partially weighted, and not directed.

File Dependency Network

To complement the previously defined bipartite graphs for collaboration and communication in git logs and mailing lists respectively, we also constructed two additional graphs for file dependency and CVEs (later presented in Figure 2.8).

To construct the File-Dependency Network, we perform static analysis of the source code files of the *entire* project using Depends [11]. We computed these dependencies at file granularity, consistent to the commit flow method. The dependencies identified at the time of analysis include:

Formally, we define the dependency graph $G_{dep} = (V, E)$, where the set of nodes $V = V_f$ comprises source files. The set of edges E $= E_{dep} \cup E_{chg}$ models dependencies between source files, where a dependency is one or more of the dependencies listed in table 2.1. The weight of the edge is defined by the count of all types of dependencies between any pair of vertices (files). The dependency graph G_{dep} is therefore uni-modal, weighted, and not directed.

2.3.6 Data Modeling - Decoupling Level

Throughout the commit flow method, although we use the file changes to construct timelines, from it identify collaboration and communication patterns, we never evaluate the content of the files itself. We fill this gap, in order to answer RQ3, by using the decoupling level metric (DL) [51] in

[11] `https://github.com/multilang-depends/depends`

Dependency Type	Description
Call	function/method invoke
Cast	type cast
Contain	variable/field definition
Create	create an instance of a certain type
Extend	parent-child relation
Implement	implemented interface
Import/Include	for example, java import, c/c++ #include, ruby require.
Mixin	mix-in relation, for example ruby include
Parameter	as a parameter of a method
Return	returned type
Throw	throw exceptions
Use	use or set variables
ImplLink	the implementation link between call and the implementation of prototype

Table 2.1: File Dependency Types

the entire project snapshot preceding the Heartbleed vulnerability. Note this is not applied per snapshot, but for the entire project, given the definition of the metric.

Decoupling Level measures how well a design is decoupled into modules based on the Design Rule Hierarchy (DRH) clustering [90]. If a module influences all other files directly or indirectly in lower layers, its DL is 0—this is the worst case where a system's files are fully connected. And if a system's files had no dependencies on each other, its DL would be 1. All real systems fall somewhere within this range. The more files a given file influences in lower layers, the lower its DL. The larger a module, the more likely it will influence more files in the lower layers, hence the lower its DL. Conversely, the more files in a lower layer are independent of files in the upper layers, the higher the DL.

Decoupling level is implemented in Dv8/ArchDia, and uses as input the git log and dependencies calculated by Depends.

2.4 Results

2.4.1 Content Analysis: OpenSSL Socio-Technical Evolution

In this section, we discuss decisions taken by OpenSSL leadership that could have an effect on the community from a socio-technical standpoint, based on manual inspection and content analysis [35] of the openssl-dev mailing list. In further sections, we return to these decisions from a quantitative standpoint.

Early Software Infrastructure: The Meeting Minutes Issue Tracker

From its genesis, OpenSSL was a community event. OpenSSL first release 0.9.1c in December 23rd of 1998 was in reality a fork of SSLeay's (an open-source implementation of SSL versions 0.8.1 and 0.9.0b), and the unreleased 0.9.1b version from C2Net. The project was renamed due to a transition of the maintainers, and a separate mailing list, codebase using CVS, and a website were created [B16]. Along with its features, the developers also inherited the design choices made in SSLeay.

Four mailing lists were made available: openssl-users (user discussion), openssl-dev (developer discussion), openssl-cvs (automated commit messages notifications), and openssl-announce (major announcements), which followed the standards at the time in other projects. In addition, CVS, which had been ported to a SVN repository could also be read via a browser [M10]. For the remainder of this work, we focus on openssl-dev [M5] and CVS, as our interest was in better understanding socio-technical effect outcomes among developers as they could influence the code base.

More formal communication in the mailing list constituted of prefixed e-mail titles with the specific tags PATCH [M6] and STATUS [M7]. As the name suggests, patches consisted of code contributions to OpenSSL. Status messages followed a structured format as shown on Figure 2.5 [M1]. We redacted the message to emphasize the various socio-technical affordances this choice of project management offered at the time.

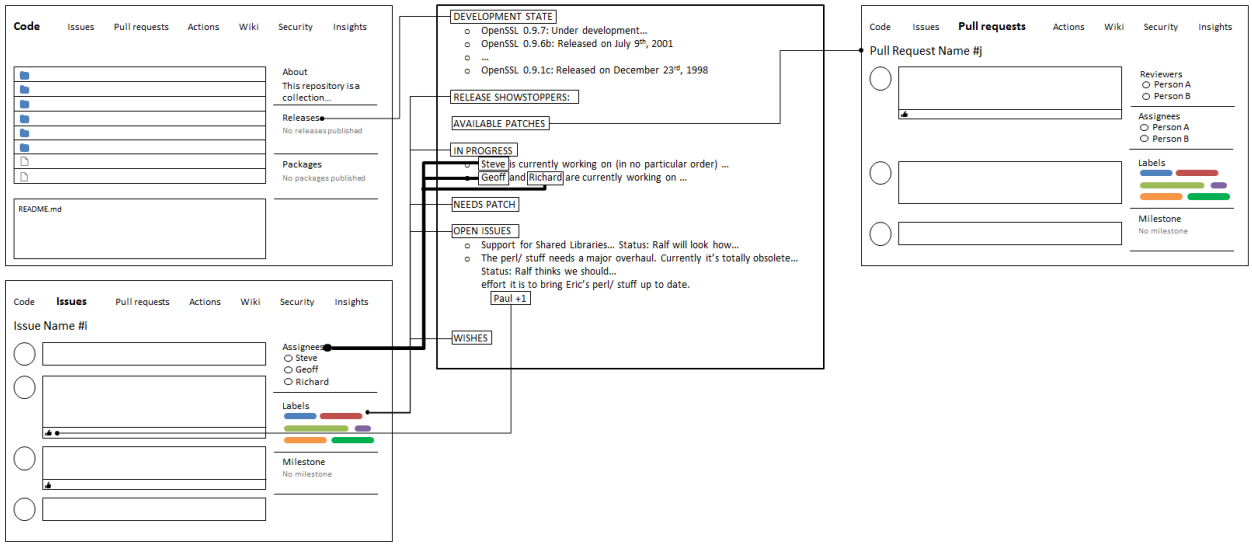


Figure 2.5: Project Status Reports in early OpenSSL.

We determined that status messages constituted the main structuring artifact in the mailing list, which was largely an unstructured environment. These status messages included many affordances—although in a rudimentary format—currently available in popular repositories today

such as GitHub, except that they were fully maintained by hand. Specifically, in Figure 2.5 we can see a few main sections: Development State, Release Showstoppers, Available Patches, In Progress, Needs Patch, Open Issues, and Wishes. Development State is reflected in GitHub's releases; Release Showstoppers, In Progress, Needs Patch, Open Issues, and Wishes are now condensed on Issue trackers by leveraging tags; and Available Patches are implemented as Pull Requests. Person attribution and upvoting is also included. Unsurprisingly, and as noted in Figure 2.5, this manual approach led to it "never (being) up-to-date".

Handling Issues on the Mailing List: The transition to Request Tracker and GitHub Bug Bankruptcy

In May 2002, OpenSSL made an informal announcement that they were moving to Request Tracker (RT) [M2]. Requests were explicitly encouraged to be posted, as they stimulated discussion. RT remained in use for over 14 years, until it was eventually phased out in favor of GitHub [B7]. This decision emphasized that e-mail integration was of interest, and would be attempted to be preserved via GitHub hooks.

Another important decision during the transition was "bug bankruptcy". The remaining set of 326 open issues which were 2 years or older would be closed. The authors of the re-issue were notified individually of the decision, and were encouraged to re-evaluate if these bugs persisted [M3]. A notification for the mailing list was also sent containing all the issues that were going to be closed [M4]. The bug bankrupcy would later be frowned upon by OpenBSD, and contributed towards the LibreSSL fork.

Other Community Changes

Several other milestones occurred throughout OpenSSL since its inception, such as changes to facilitate usage [B1] [B2] [B3], and a more organized way to report status, such as a release strategy, roadmap and blog [B4] [B5] [B6] [B8]. The contribution framework and rules also matured: a committer group was defined [B9], and license changes were made [B10] [B11] [B12] [M11]. The code base was also improved, by defining a coding style [B13] and doing a cleanup [B14]. For its first 15 years, OpenSSL developers were a small collection of individuals working on a part-time basis and this group fluctuated, but by 20 years it was a more stable community [B16].

OpenSSL Fork: LibreSSL

While changes in infrastructure, community and policies took place in OpenSSL, its impact on the code base, and in particular software vulnerabilities was minimal. On April 2014, shortly after the occurrence of Heartbleed, OpenBSD created a fork of OpenSSL [B17]. Several public critiques appeared at that time concerning what developers in OpenBSD considered to be unacceptable in

OpenSSL [B18] [B19], and over 90,000 lines of code were removed in the fork [B20]. These critiques also extended to OpenSSL's project practices, such as their negligence with respect to the Request Tracker [B21].

In an interview in 2014, when the vulnerability was fixed, the developer that introduced the Heartbleed vulnerability noted [B22] it as an "oversight". However, based on the problems that motivated the LibreSSL fork, we decided to investigate the 2 files required to fix the vulnerability. Figure 2.8 shows the CVE networks to date and highlight Heartbleed (CVE 2014-0160).

As seen in the figure, the two files involved in the fix are implicated in many other CVEs.

2.4.2 Addressing RQ1

> RQ1: Do community [STATUS] threads drive community engagement?

In this RQ we were interested to know if the method adopted by OpenSSL to inform the community of project status, as shown in Figure 2.5, improved community engagement. For this purpose we measure engagement as the numbers of replies generated for e-mail threads. If either a) the [STATUS] thread generated a larger number of replies in its own thread, or b) 'rippled' to more e-mail threads when it was posted, we assume it served as a useful method to provide community engagement.

Figure 2.6 gives insight into RQ1. To create the visualization, we detected e-mail threads (initial e-mails and replies) containing the term '$[STATUS]$'. We observed from our qualitative analysis that "[STATUS]" was used consistently and without variation by OpenSSL. We also observed that [STATUS] threads contained the date they were posted (e.g. [STATUS] OpenSSL (Sun 20-Oct-2002)). We also leveraged this information to analyze status postings back to the year 2000, since, as we noted in the data collection section, e-mail information was missing prior to 2002. According to Figure 2.6, we observe that instead of [STATUS] threads driving community involvement, the community had a will of its own, i.e. it behaved independently of the [STATUS] threads. Between 2001 and 2003, we can see that [STATUS] threads (dots in the bottom plot) had few replies. Moreover, the number of threads increased considerably after the Request Tracker was introduced, despite the majority of [STATUS] threads continuing to have few responses. As shown in the figure, this format of communication stopped being used around 2003. Therefore, the answer to RQ1 is no: [STATUS] threads did not drive community engagement.

2.4.3 Addressing RQ2

> RQ2: Did the community actually adopt the Request Tracker after its inception?

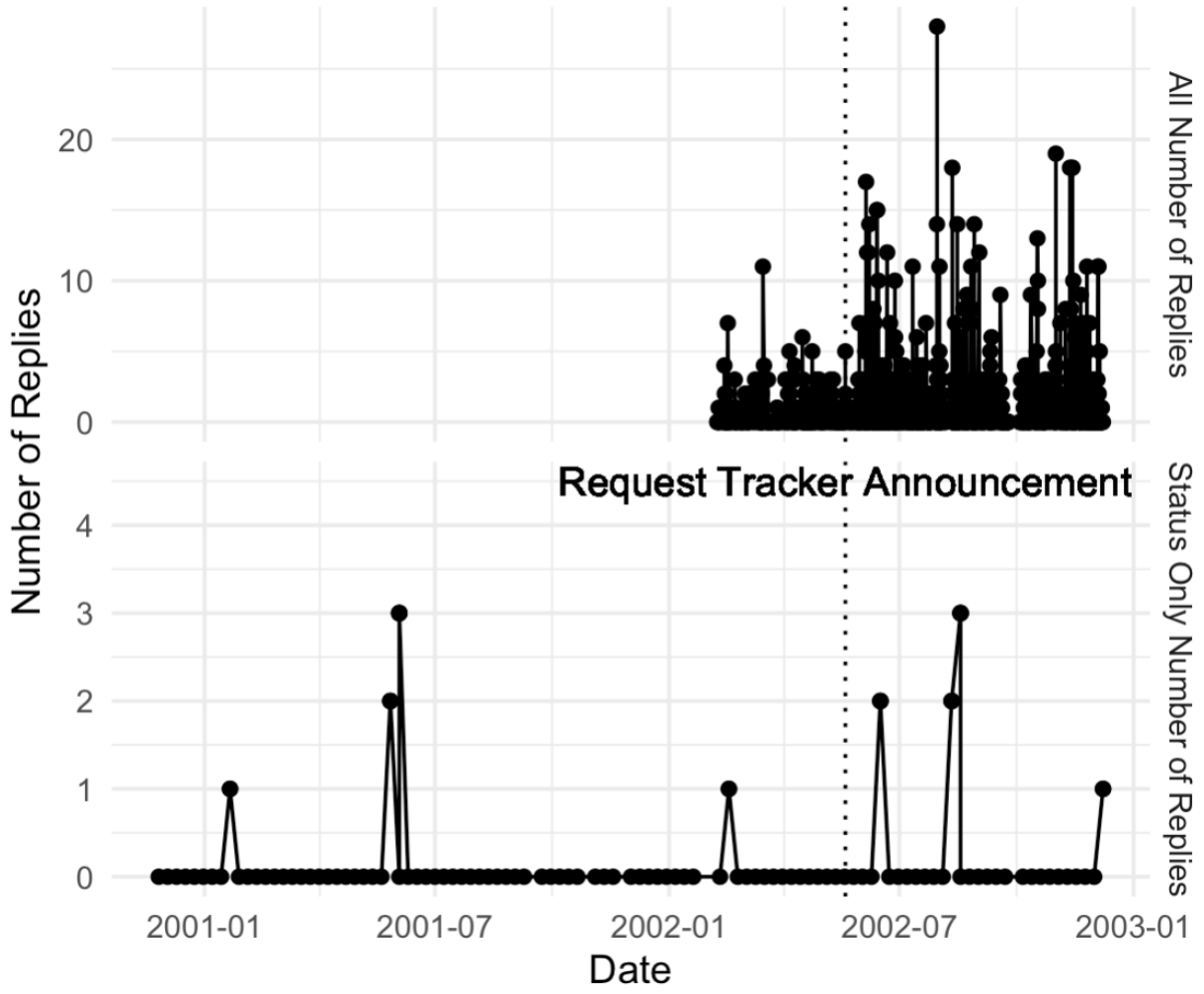


Figure 2.6: Number of replies per [STATUS] thread versus Number of replies to any e-mail thread. Each dot represents one [STATUS] e-mail thread in a given day.

To answer RQ2, we leveraged the fact that OpenSSL's Request Tracker posts to the mailing list any new ticket or reply with an annotated id (e.g. '[openssl.org #3544] Remove MWERKS support'). E-mail threads which did not contain an issue id of this form, were therefore authored directly in the mailing list. By defining adoption as the number of e-mail threads from the Request Tracker vs the Mailing list, we are able to assess if the community adopted the Request Tracker, as shown in Figure 2.7.

We can see from this figure that the Request Tracker was adopted immediately, and was more heavily used than the mailing list until 2002. From 2002 through 2014 the mailing list was used slightly more heavily, but both channels were actively used. As we noted in our content analysis OpenSSL underwent many changes. However, none of these seem to impact the overall activity of the community, as the average number of threads consistently hovered around 100. In late 2016

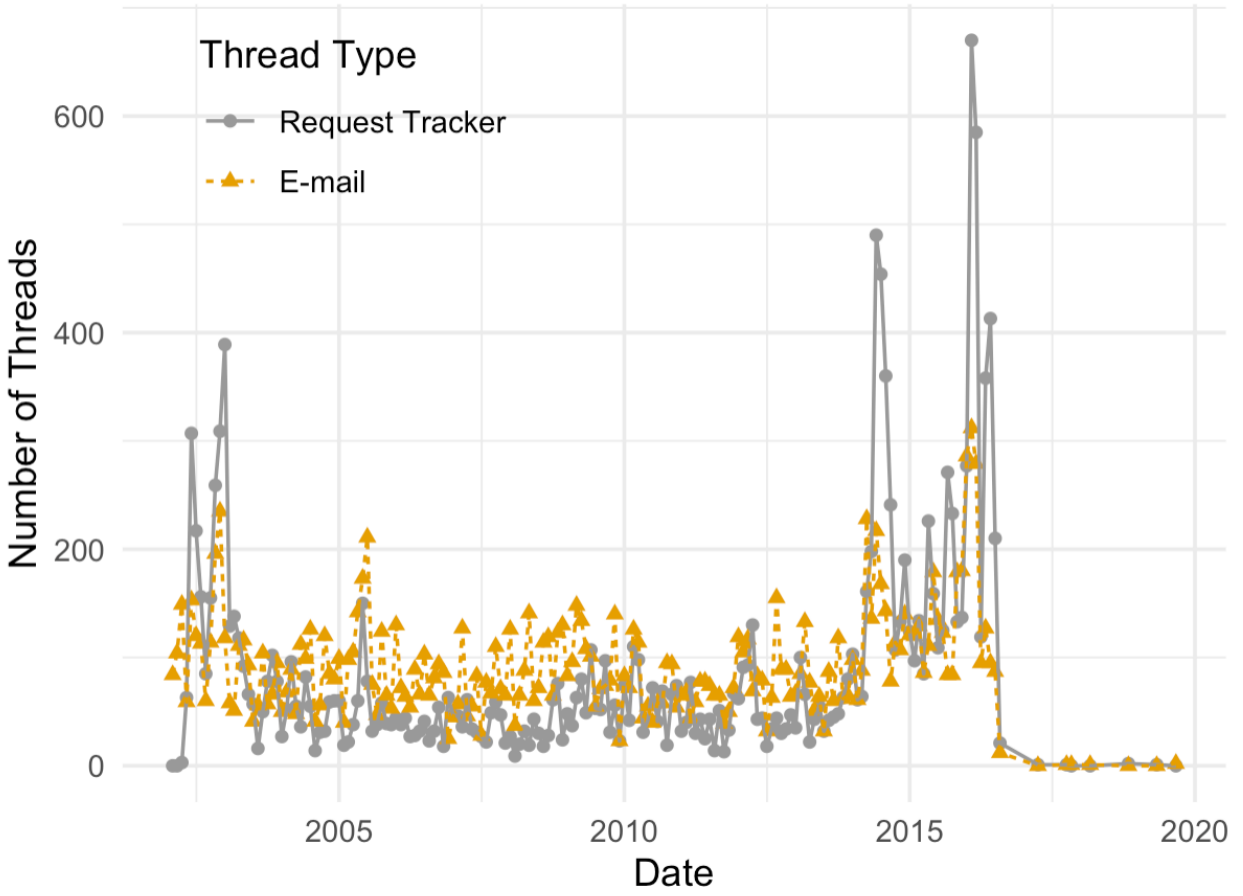


Figure 2.7: Number of threads generated by Request Tracker vs via Mailing List.

and early 2017 we also observe a sudden drop in activity on both mediums. This is due to OpenSSL announcing their complete transition Github [B7].

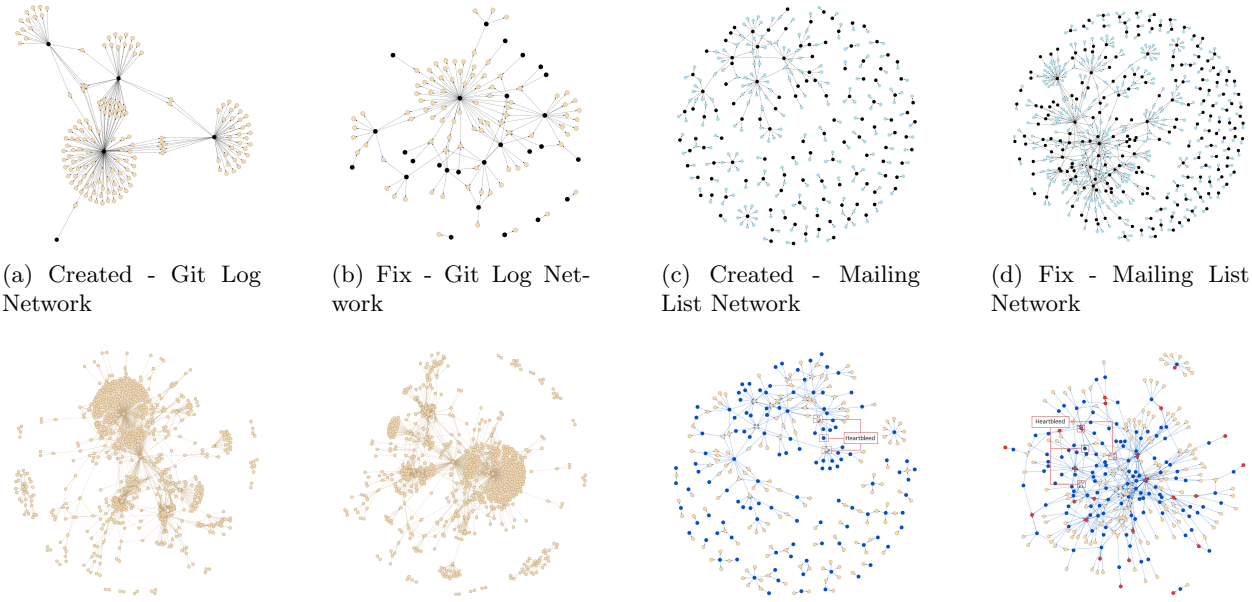
Therefore, our answer to RQ2 is yes, and more specifically, that the Request Tracker coexisted with the mailing list and was used in roughly same proportion as the mailing list, suggesting an infrastructure with both mailing list and issue tracker appeared to cater to community preferences. This in turn suggests that the full transition to GitHub [B7], which can be seen as an alternative to Request Tracker, and the complete elimination of mailing lists may have not been ideal, as the community still preferred to use both resources throughout the project history.

2.4.4 Addressing RQ3

RQ3: Did the architecture practices of OpenSSL lead to major problems for project success?

To understand the architecture of a project we need to choose one or more snapshots to study. In this case we have focused our study on architecture of OpenSSL in the context of a named vulnerability. In particular we chose the Heartbleed bug to study, given the enormous world-wide impact it had.

The architecture and code base of OpenSSL had long been subject to public criticism. As mentioned above, the poor quality of the code and architecture were driving factors in the LibreSSL fork. The exact commits which introduced [C1] and fixed [C2] the Heartbleed bug were used to anchor our analysis and to better contextualize the claims of LibreSSL.



(a) Created - Git Log Network

(b) Fix - Git Log Network

(c) Created - Mailing List Network

(d) Fix - Mailing List Network

(e) Created - File Dependency Network

(f) Fix - File Dependency Network

(g) CVE Network to date

(h) CWE Network to date

Figure 2.8: Heartbleed: Vulnerability Introduced and Fix Snapshots

To provide more context to the state of the project in this RQ3 and the subsequent RQ4 and RQ5, in Figure 2.8 we plot various networks snapshots of the OpenSSL project when Heartbleed was introduced, and when it was fixed using the ForceAtlas2 layout (please refer to the method section for the formal definition of these networks, properties and edge weight definitions).

The object of our interest in this chapter for RQ3, RQ4, and RQ5 are vulnerabilities, which are shown in sub-figures g and h. Yellow vertices represent files, blue vertices represent CVEs, and red vertices CWEs. We also highlight where Heartbleed is in the network. As we can see, the files which were modified to fix the vulnerability were previous implicated in others (yellow nodes are connected to many other blue nodes), suggesting the problem may be caused by other factors, which in this work we assumed to be socio-technical related. In RQ3 we focus on Heartbleed, but in RQ4 and RQ5 we use all the vulnerabilities identified in the project shown in this Figure.

We can see from the Figure 2.8 that the communication was more silo-ed when the vulnerability was introduced in contrast to when it was fixed (sub-figures c-d, blue nodes represent e-mail threads, black nodes represent authors). We can also see collaboration was centralized in very few authors when the vulnerability was introduced, in contrast to when it was fixed (sub-figures a,b, again black nodes represent authors, and yellow node files). Together, we see fewer authors and sparse communication was present during the vulnerability introduction. The intuition we elaborate here between collaboration and communication is captured in the social smells metrics we use in this work. We elaborate on the social smells in OpenSSL in the subsequent RQ4 and RQ5.

The difference in the dependency network is more subtle (sub-figure e,f, yellow nodes represent files), but we can observe the giant component in the center of the graph appears slightly smaller when the vulnerability was fixed, instead of created. This suggests there is less coupling between all files in the system (as ForceAtlas2 layout would gravitate them together otherwise). The intuition we elaborate here for the architectural dependency of files is captured in the Decoupling Level (DL) metric.

To analyze the architectural quality of OpenSSL's code base, we calculated both its architectural design flaws and its Decoupling Level (DL). These are well-established and empirically validated measures of architecture quality [50], [51]. DL is a measure of the degree to which the modules of a system are decoupled–higher numbers are better because high decoupling means that developers can work independently in the knowledge that their changes will likely not affect each other's files. Architectural flaws are known violations of design best practices. These flaws are strongly correlated with bugs, churn, effort, and vulnerabilities [22].

Our assessment of the state of the architecture at the time when the Heartbleed bug was introduced indicates that OpenSSL is one of the worst structured systems we have ever encountered. The DL score of its latest release is 13.58%—the lowest score we have ever seen reported in a software project in the literature. And while its DL score in 2014 was somewhat better, at 42.87%, this still puts OpenSSL at that time in the bottom 20th percentile of all projects reported [51]. Furthermore, it has nearly as many design flaws as it has files in the project, and this is an unacceptably high burden of technical debt that would crush any project. Thus we feel confident in answering RQ3 with "yes"–it is highly likely that the architecture of OpenSSL led to major problems for project success. What we can not conclude from this analysis is whether the communication issues led to these architectural structural problems, derived from structural problems, or neither.

2.4.5 Addressing RQ4

To answer RQ4, we performed Spearman correlation tests on the summary of the obtained timelines, as defined in Section 2.3. We observed, first, that across all vulnerability timelines, the mean number of commits was strongly and negatively correlated to the mean communicability (ρ=0.67,p<0.001), and socio-technical congruence (ρ=0.71,p<0.001). This means the more developers there were in

an OpenSSL commit, the less their decisions were diffused; that is, they largely worked in silos, without synchronizing through the communication channels available in the project. Consistent results for moderate negative correlation can be also observed for the mean sum of churn for mean communicability (ρ=0.51,p<0.001) and socio-technical congruence (ρ=0.56,p<0.001). Therefore, we conclude that the answer to RQ4 is yes, with respect to communicability; communicability issues preceded vulnerabilities along the identified timelines. Second, we observed strong—this time positive—correlations between the mean number of "organizational silos" and the mean number of commits (ρ=0.85,p<0.001) as well as mean "missing social link" (ρ=0.88,p<0.001) smells [79]. We also again observe strong correlations between mean sum churn and org silos (ρ=0.68,p<0.001) and missing links (ρ=0.72,p<0.001). This means the greater the numbers of developers making contributions to the code base of OpenSSL without being registered in the mailing list (org silo) or at least communicating in one e-mail thread within a 3 month time window, the larger the number of commits and code changes that were made. Therefore, the answer to RQ4 is yes with respect to social smells. We do not, however, draw any conclusions about radio silence, as the observed p value was not statistically significant (p>0.5).

2.4.6 Addressing RQ5

In contrast to RQ4, the analysis of RQ5 does not involve summing metrics across the commit timeline; rather, it is an analysis of what happens within and across successive time periods, and especially whether there is evidence that socio-technical measures (e.g., social smells, communicability, and congruence) for the current time period have direct causal relationships with the outcomes for the *next* time period. Also, as this is a causal analysis, it accounts for the effects of other variables, such as number of developers involved in a time period, which could inflate multiple measures. (This analysis thus directly addresses a possible threat to validity for RQ4, as a causal analysis will systematically control for such common causes.) For this reason, we included a third project outcome, *code_dev*, to the other measures taken for both the current and next time period: commit and churn. To differentiate the names of the outcomes for the next time period from the outcomes for the current period, we append a "2" to their names; resulting in code_dev2, commit2, and churn2.

Our goal was to investigate whether socio-technical problems occurring during one time period could directly affect the three project outcomes in the next time period, after controlling for the project outcomes from the current time period. (It would not be too surprising if we find that each outcome is a direct cause of the corresponding outcome for the next time period, e.g., churn directly causes churn2.) Including outcomes for both time periods allows us to controlling for just such unsurprising cross-period outcome relationships.

Recognizing that some CVEs might have inherent idiosyncrasies that could be reflected in some metrics we partitioned the dataset by CVE. Indeed, we later found 14 CVEs (out of 121) that had

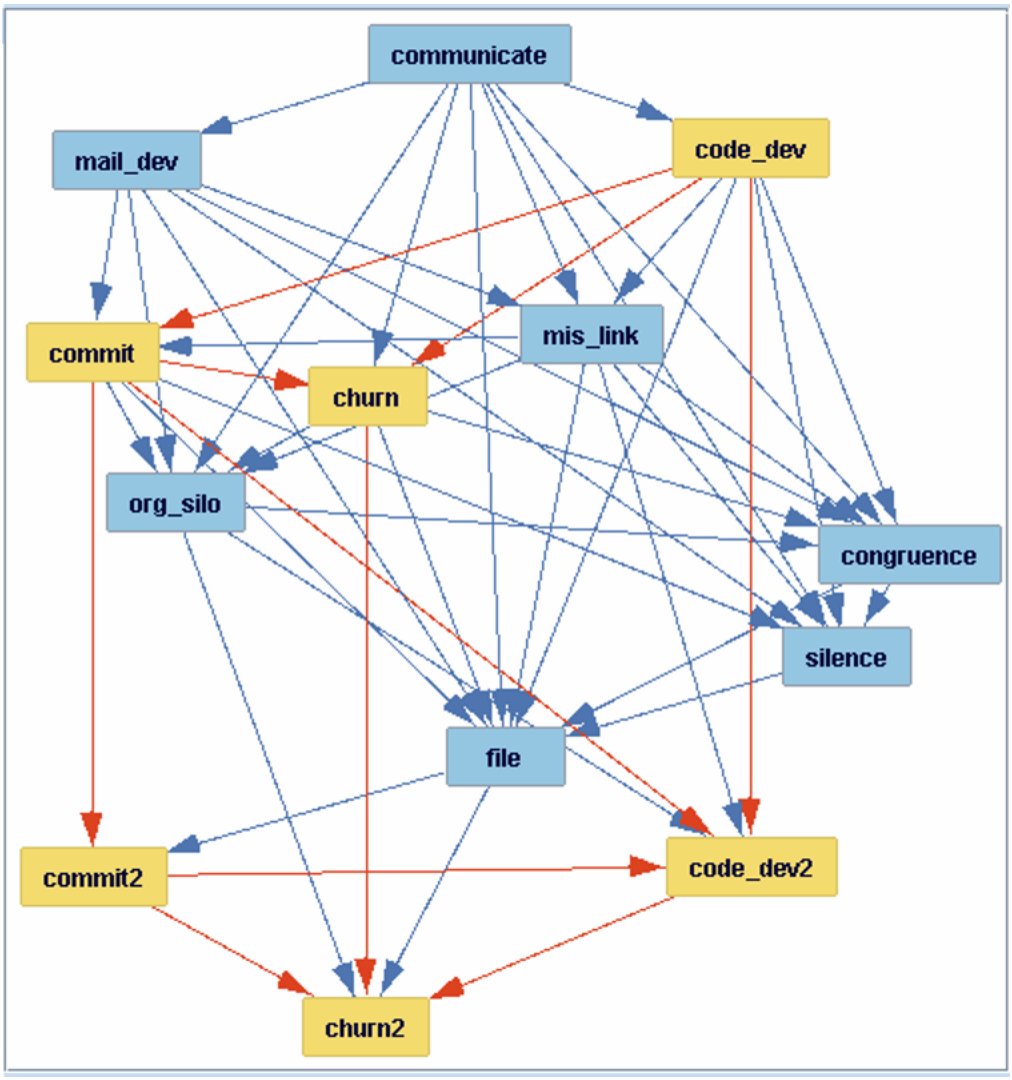


Figure 2.9: Causal graph with 1% trim

just such idiosyncratic effects on outcomes. This also suggests some limits to generalizability of our causal results—if our causal structure (shown in Figure 2.9) was fully generalizable to other contexts without modification to the causal graphs obtained, we would not expect to see such idiosyncratic effects; however, the degree of generalizability seems reasonable: 88% of the CVEs do not engage in causal relationships with outcomes, according to the analysis that we describe in greater detail, next.

As only 17% of the time periods had missing data, primarily due to the project transitioning to new bug tracking and communication platforms, we addressed missing data through listwise deletion, which presumes that data is missing completely at random (MCAR), which seems reasonable [3].

For this dataset, we imposed two types of causal constraints on the variables in the dataset: (1) none of the CVE indicator variables are causally related to the others; further, the CVE variables are exogenous (not caused by socio-technical variables or outcome variables); and (2) none of the outcomes (*code_dev, commit* and *churn*) can be causes of variables in the prior time period. The former constraint is not strictly true——the injection or remediation of one CVE could in fact increase the likelihood of suffering from another, but we have reason to believe that such cases are in fact rare, and it is an assumption that allows us to simplify the analysis. The latter constraint is simply a reasonable chronological limitation.

We used Tetrad with all default settings except that we set Penalty Discount = 2 (resulting in a sparser causal structure than with the default setting of 1; set the number of Bootstrap Samples = 2000 (to help ensure some level of repeatability of our results); Ensemble Method = Preserved, which tracks every time an edge appears in a bootstrapped search result; and the first step of the search is required to be symmetric (try both orientations for the first edge)[12].

To more robustly identify the causal mechanisms responsible for the three outcomes (i.e., to keep the false-discovery rate low), we applied a variant of the causal search technique described in [30] for the first and third steps: (1) prior to search, we expanded the dataset to include randomized versions of many variables, called "null variables," but given the larger number of variables representing CVEs, we only used a sample of these variables after an initial screening search to select which CVEs to treat in this way; (2) the (expanded) dataset was repeatedly sampled and each resulting bootstrap sample was then searched, with edge statistics accumulated, including for the random edges (edges involving a null variable); and (3) we trimmed the resulting graph up to the 1st percentile (i.e., pairs of variables are ranked by increasing frequency of no edge, and edges are only assigned to those pairs for which no edge appears less often than the 1st-percentile random edge frequency). We could do this because our dataset was large, allowing us to choose more conservatively.

The result of causal discovery and trimming are indicated in Figure 2.9. This figure shows the parents and grandparents of outcomes (variables having a suffix "2") for the next time period. For emphasis, outcomes from the current and the next time period are colored gold. Note the direct causal relationship that each outcome has with its corresponding outcome in the next time period, which is not surprising. But note also that org_silo and mis_link have a direct causal relationship with the outcomes experienced in the *next* time period, even after controlling for those same outcomes from the current time period. Thus, socio-technical behaviors have a significant and persistent effect that lasts into the next time period in spite of differences in how the next time period might unfold, which would seem to be reason for concern——many social smells cause harm and simply do not "fade away."

What is this figure saying regarding the impact of socio-technical behaviors? (1) The communi-

[12] `https://www.ccd.pitt.edu//wiki/index.php?title=Fast_Greedy_Equivalence_Search_(FGES)_Algorithm_for_Continuous_Variables`

cability metric directly affects (causes) many social smells, such as missing links, and socio-technical congruence (which in turn affects other variables such as the number of organizational silos, radio silence, and socio-technical congruence); (2) these variables directly affect the number of developers involved in a commit and the number of commits which, in turn, affect churn. Moreover, two variables in the current time period (org_silo and mis_link (missing links)) do affect outcomes in the next, providing strong evidence in favor of RQ5: there are indeed strong causal links between socio-technical smells and the software vulnerabilities extending to the next time period.

2.5 Conclusions and Future Work

In this exploratory study, our immediate objective was to study the OpenSSL project, its social networks, and the consequences of social smells on project outcomes, specifically vulnerabilities and the effort required to resolve them. Our secondary objective was to explore the use of causal modeling to be able to make stronger claims than are normally found in correlation analyses.

We performed a content analysis to understand the decisions that OpenSSL made in its community and its code base. We defined a social network model to represent developers, e-mail threads, files, commits, and software vulnerabilities and weaknesses as a network over time. We used this model to calculate various socio-technical metrics, and to understand the relationships between these metrics and vulnerabilities as outcomes. We identified a number of problems in OpenSSL, both in terms of its community structure and engagement and in terms of its architecture. We noted that the project did make some attempts to drive community engagement, such as providing semi-structured mailing lists and introducing an issue tracker. The issue tracker was adopted almost immediately, but over a decade, it was used less and co-existed with the mailing list. Overall, it appears that OpenSSL's attempts to improve its community engagement were unsuccessful.

Our results show what we believe are strong and interesting results: that social smells are indeed important factors that mediate significant project outcomes in terms of the effort associated with security vulnerabilities.

This emerging research is important, we believe, in two ways: 1) it guides potential future work in establishing socio-technical guidelines for maintaining security in critical software, and 2) it introduces causal modeling as a tool for analyzing software engineering datasets to better explore the relationships between the measured variables.

CHAPTER 3
IDENTIFYING EMERGING THREATS THROUGH FREE TEXT: GROUPING PERFORMANCE

This chapter is derived in part from the ICMLA'18 paper "Indexing Text Related to Software Vulnerabilities in Noisy Communities Through Topic Modelling" and the Scitech AIAA'21 paper "Augmenting Topic Finding in the NASA Aviation Safety Reporting System using Topic Modeling".

In previous chapters, we modeled software vulnerabilities as bugs, and captured in the commit flow method how socio-technical practices correlate to these bugs. We now focus on modeling software vulnerabilities as topics, consistent with PERCEIVE holistically framework goal.

3.1 Introduction

In this chapter, we analyze the changes to files by comparing the similarity of their content instead of constructing socio-technical metrics by using topic modeling, an unsupervised approach. We chose an unsupervised approach because, according to [55], with supervised learning methods, the topics are fixed in advance instead of being allowed to emerge; in addition, evaluation is harder in supervised contexts, as labeled data is scarce. Before we can construct a notion of content similarity *over time*, as we did with commit flow, we examine their performance in discrete *points in time*, as a building block step. More specifically, we define the following research question:

RQ0: Can topic modeling correctly group similar documents?

We perform two studies to assess the viability of automatically identifying topics, one grouping software vulnerability discussions in a security mailing list and another grouping aviation safety threat report narratives. Different from related literature (e.g. [37, 1], which interprets topics subjectively through a set of terms, in both studies, we leverage available data to evaluate the performance of our groupings empirically.

We will more precisely define what it is meant by 'correctly' and 'similar' in the validation section by posing more specific research questions.

The contributions of this chapter are as follows:

- We provide two evaluation methods for empirical evaluation are provided, one for each study. Commonly, the literature only uses the top terms to subjectively infer meaning from the topics.

- We assess the applicability of topic modeling in two different domains.

The remainder of this chapter is organized as follows: We begin this chapter by presenting the topic modeling method common to both use cases. We then discuss a general architecture for data evaluation of the model, the major contribution of this chapter. Then, we fork the chapter in the two use cases, software vulnerabilities and safety threats, to introduce their specifics (e.g., data collection and the merit of using automated groupings in their respective domains). We then discuss and compare the use case findings, conclusions, and future work.

3.2 Method

In this Section, we define how we automatically group documents at discrete points in time, and how we can evaluate whether documents can be grouped correctly given the data available. In the next section, we explain, for each study what constitutes the document and the used evaluation framework.

3.2.1 Notation and Terminology

We use a notation similar to Blei et al. [10] to describe the equations in this section, except by distinguishing words, documents, and corpora by distinct letters and adding an explicit definition of topics:

- A word w is an item from a vocabulary indexed by $1, \ldots, V$. The vth word in the vocabulary is represented by a V-vector w such that $w^v = 1$ and $w^u = 0$ for $u \neq v$.
- A document is a sequence of N words denoted by $d = (w_1, w_2, \ldots, w_N)$, where w_n is the nth word in the sequence.
- A corpus is a collection of M documents denoted by $\mathbf{c} = d_1, d_2, \ldots, d_M$.
- A topic z is a distribution over words denoted by $\mathbf{z} = w_1, w_2, \ldots, w_V$.

Topics are the "bags of words" that we use to identify documents within a corpus.

3.2.2 Topic Model

We chose two implementations of Latent Dirichlet Allocation (LDA), one using variational expectation-maximization (LDA-VEM) [10] and WarpLDA [15] to apply to each of the documents. LDA is a generative probabilistic model of a corpus, where documents are represented as random mixtures over latent topics, and each topic is characterized by a distribution over words [10].

By definition, a *document* is the set of words we are interested in grouping based on content similarity in each use case. Once the document is defined in each use case, a document-frequency

matrix is constructed to use as input for the topic modeling step. A document-frequency matrix represents each document as a row, and each word occurring in at least one of the documents as a column. A given cell in the table defines the frequency with which the word (on the column) occurs in a given document (on the row). Because the full corpus vocabulary (the set of unique words) is typically much higher than the usage in any given document, document-frequency matrices are usually sparse.

For a given document-frequency matrix, LDA provides (for the chosen number of topics): the distribution of the documents over topics (the document-topic matrix), and the distribution of topics over words (terms), commonly known as the topic-term matrix. As the name implies, a document-topic matrix lists the documents as rows, and topics as columns, where the number of columns is specified by the a priori choice of the number of topics, and cells indicate the proportion each document is about a given topic. A topic-term-matrix, in turn, lists topics as rows and terms as columns, and cells can be interpreted as the relevance or contribution of the term in representing the topic, expressed as a probability.

Earlier in this chapter we claimed that the literature often presents a subjective analysis based on the set of words of each topic instead of an empirical evaluation of the groupings. By this we meant that related work uses a subjective interpretation of the *topic-term matrix*, instead of using the *document-topic* matrix. In this chapter, our goal is to construct an objective empirical evaluation of the *document-topic* matrix. In the following Chapter 4, we evaluate the *topic-term matrix*.

3.2.3 Model Tuning

LDA requires the number of topics and other hyperparameters to be specified a priori, which can be obtained through measurements such as perplexity [10] (Equation 3.1), where a lower perplexity score indicates better generalization performance.

$$perplexity(c_{test}) = exp\left\{-\frac{\sum_{d=1}^{M} \log p(w_d)}{\sum_{d=1}^{M} N_d}\right\} \tag{3.1}$$

Once a measurement is defined, such as perplexity, we can calculate it using for example the elbow method, where a human identifies the point in the curve where the decrease in perplexity is too small to justify more topics. Several other optimization methods also exist such as differential evolution [74, 53] and approaches which rely on the elbow method [1]. However, this was impractical given the large number of topics in our corpus. Thus, we automated this method as shown in Algorithm 1.

Additionally, we found that while perplexity was useful to narrow the search space for the number of topics, in practice the number of documents that could be assigned to a topic (see Equation 3.2) could be too hard to interpret for any conclusions to be drawn from the model. As

[1] `https://cran.r-project.org/web/packages/ldatuning`

such, the final choice of the number of topics was not only based on perplexity but also the number of documents assigned to topics.

Algorithm 1 Choose K

1: $up\_to\_n\_topics \leftarrow 15$
2: $z\_docs\_median \leftarrow new\_vector(up\_to\_n\_topics$ - 1)
3: $perplexities \leftarrow new\_vector(up\_to\_n\_topics$ - 1)
4: **for** $k = 2$ to $up\_to\_n\_topics - 1$ **do**
5: $\quad model \leftarrow$ LDA(n_topics)
6: $\quad z\_docs\_median_{k-1} \leftarrow$ GET_MEDIAN_DOCS(model)
7: $\quad perplexities_{k-1} \leftarrow$ GET_PERPLEXITY(model)
8: **end for**
9: candidate_k $\leftarrow$ FILTER_KS(perplexities,z_docs_median,10)

From lines 1 to 8 we calculate the perplexities and median documents per topic by creating models ranging from 2 to *up_to_n_topics*. On each iteration, the perplexity is calculated using Equation 3.1. To calculate the number of documents, we rely on the deterministic mapping of Equation 3.2 to first assign documents to topics, which are then counted. *get_median_docs* then returns the median number of documents across all topics. Line 9 performs two steps: it filters the models with a median of at least 10 documents per topic, and it automates the elbow method by identifying the highest slope of all points remaining. By applying the two rules, we identify the chosen number of topics for a corpus. The iteration of the number of topics up to 15 was due to computational constraints, while the median number of documents was based on the authors' judgments of the suitable number of documents that an analyst would wish to, or be able to, evaluate.

3.2.4 Deterministic Mapping

For the purposes of interpreting documents to be mapped to a topic, we define a deterministic map from documents to topics in the document-topic matrix for each month, using the highest probability of the topic given the document in Eq. 3.2.

$$argmax_{z_i} p(z_i|d) \tag{3.2}$$

Thus we claim a document is assigned to a topic if that topic has the highest proportion of the words in the document in the original matrix [64]. Since each row is a document in the document-topic matrix, we can interpret the document-topic matrix as labeling the topic to which a document belongs.

The motivation behind our choice to use this deterministic mapping is because of how we empirically evaluate both use cases. In the ground truth we construct from available data, only direct assignments are available (i.e. "document A belongs to group i"), instead of probabilistic

assignments.

3.2.5 Evaluation

In RQ0, we broadly defined our aim as evaluating whether topic modeling can "correctly" group "similar" documents. We now define more precisely what we mean by that, by defining how we empirically evaluate both use cases.

As there is no true definition of universal groupings for documents, we consider a grouping to be "correct" if it is identical to another grouping done manually. Moreover, we accept that documents may be "similar" based on different or even conflicting criteria. For example, if documents describe aviation safety threats, they may convey more than one safety threat, and depending on the safety threat used for grouping, the groupings may be different. To minimize this threat to validity, we used manually curated groupings that are *operationally relevant* to each study. Specifically, while it is true that multiple groupings could be considered correct, as the groupings were defined a-priori by each respective organization, and not by us, the manually curated grouping made available in each study is assumed to be *more operationally relevant* than the alternatives.

This intuition differs from standard practice, which assumes that the use of perplexity alone, as described in section 3.2.3 suffices for evaluation. Indeed, there is a disconnect between how topic models are evaluated and why we expect topic models to be useful [11]. Topic models are often used to organize, summarize, and help users explore large corpora, and there is no technical reason to suppose that holdout perplexity or other measures corresponds to better organization or easier interpretation [11, 41].

In addition to the choice of measures such as perplexity, holdout is a type of internal criterion, but there are also external criterion methods. External criteria use direct evaluation as a gold standard (a surrogate for user judgments for the groupings) [41]. The gold standard is ideally produced by human judges with a good level of inter-judge agreement [41]. In this chapter, we have our 'ground truth' serve the purpose of a gold standard, as it was defined a-priori by an inter-judge agreement.

Set Matching and Adjusted Rand Index

An intuitive way to evaluate clusters using an external criterion is using *set matching* [84], specifically classification error rate. However, this approach has faced criticism due to disregarding part of the unmatched groups in the evaluation metric [84]. An improvement over this metric is pair counting, such as the Adjusted Rand Index (ARI). Intuitively, we consider every possible pair of documents within each grouping, and evaluate whether a) they were originally or not part of the same grouping in the ground truth dataset and b) whether topic modeling considers them to be part of the same grouping. Thus, we are able to generate a confusion matrix, and calculate the

ARI, a real value where 0 indicates the performance equivalent to a random model, -1 worse than random, and 1 better than random.

Random Topic Model

Using Equation 3.2 we can interpret topic modeling as an algorithm that assigns topics to documents. To provide a baseline for evaluation, we created a 'random' model to perform label assignment. Specifically, provided some topics, the algorithm randomly assigns, with replacement, a topic label to each document. Although simple, this algorithm has the same output as LDA (after using Equation 3.2), and is used to assess how much better our process is, as compared to a random assignment.

3.3 Results

In the following subsections, we present our two studies applying the discussed methods. Each study begins with the relevance of automated groupings in its respective domain, the identification of the unit of analysis, i.e. the *document*, and how the evaluation setup is instantiated following the guidelines defined in our method. We present the results of the method in their respective case study, but discuss the implications comparing both studies at the end of the chapter.

3.3.1 Study 1 - Topic Modeling Software Vulnerabilities

Informal public disclosure of security information, such as vulnerabilities or exploits, often occurs well before they are properly classified in security databases [44]. New security vulnerabilities are often disseminated on social communities. Many security investigators favor "full disclosure", sometimes supported by a detailed exploit, and do not collaborate with the vendors of vulnerable software, under the belief this puts pressure on vendors to fix vulnerabilities [28]. Even among responsible disclosure through bug bounty programs, there is evidence of intentionally delaying disclosure to identify chained vulnerabilities, increasing monetary and recognition returns [85]. Security investigators learn from their community, and when personal contacts are insufficient, they seek out online information, typically in the form of expert blog or web forum posts, and rarely through formal channels [85]. Cybersecurity information obtained through discourse is kind of software vulnerability sense-giving, reducing the uncertainty of vulnerabilities, adding meaning to their potential causes, and updating the investigators' mental models [75].

Staying informed is time-consuming, because important information is scattered across many sources [82]. One existing solution to improve information accessibility is the Common Vulnerabilities and Exposures (CVE) project, which defines itself as a community-based mechanism to make identifying, finding, and fixing software product vulnerabilities more rapid and efficient [43]. Contrary to its motivation, however, CVE offers little means to navigate its numerous entries, and

its growth rate is increasing [68]. It is through The Common Weakness Enumeration (CWE)[2], that investigators can find the means to navigate, top-down, to related CVEs based on a topic of interest. Unfortunately, navigating CWEs is not without problems. Because there are many CWEs to choose from, the mapping between CVE and CWEs is limited to a subset of the weaknesses[3]. The use of a subset, in turn, leads to a coarse-grained classification, causing emerging trends to be buried in related but often vague CWE classifiers [55].

With vulnerabilities being discussed daily in 'noisy' communities— containing Spam or irrelevant discussions—we see an opportunity to investigate whether textual vulnerability themes can be leveraged to save security investigators time in exploring new content. Specifically, we aim to determine if it is feasable to fully automate aggregation of related software vulnerability content in social communities, which are inherently noisy. If yes, such a tool would aid investigators in more quickly identifying related discussions of interest, instead of either having to read the entire online source manually or using word matching via search boxes to zero in on specific strings. Our vision could be realized by offering an intermediate layer, identifying groups of related discussions, or topics, to narrow down an investigator's search for relevant content.

Our instantiated research questions inspired from RQ0 are therefore:

RQ1. Can vulnerability topics in free-form discourse be accurately grouped using a bag of words approach?

RQ2. Are the observed patterns due to random effects?

Literature Review

Several proactive cybersecurity approaches have been proposed to reduce the damage of cybersecurity attacks. These methods vary in what data they leverage and what they attempt to predict. For instance, Bilge et al. [7] use computer logs to predict which machines are at risk of infection. Liu et al. [39] used features of an organization network to predict organizational breaches. Soska and Christin [72] uses data from websites (e.g. traffic, file structure, and content) to predict if benign websites will be compromised in the future. A more consolidated area of research of proactive cybersecurity uses as data the code structure [4], [86] and/or related systems to software development, such as reported 'CWE-*' in code repositories' issue trackers [54] to identify software vulnerabilities. For text data, recent work has focused on entity recognition through (hand or semi-hand labeled) identifiers [44], [88], and alerts in the deepnet [59] or twitter [49]. Unlike these

[2] `https://cwe.mitre.org/`

[3] For currently used categories, see: `https://nvd.nist.gov/vuln/categories`

works, instead of focusing on entities (websites, organizations, issue trackers, tweets, etc.), we are interested in software vulnerabilities as textual concepts discussed on the web.

To our knowledge, the most similar work to ours is by Tsai et al. [83], where the authors analyzed blogs for categories of threats related to the detection of security threats and cyber crime. They use an Author-Topic Model extending Latent Dirichlet allocation (LDA). However, their analysis results are limited to just listing relevant keywords of different topics. Our work hinges on using e-mails labelled with CVE-IDs as ground truth. With this dataset we are able to calculate an accuracy measure for our model, as discussed in Section 3.3.1.

In Sabottke et al. [68], the authors use text data to assess the risk vulnerabilities that may be exploited, arguing that existing methods such as the Common Vulnerability Scoring System mark too many as high risk. Tweets and exploit databases—requiring CVE-* identifiers—are used to create a linear support vector machine classifier (supervised learning). Contrary to [68], our work does not rely on pre-labelled data, but instead focus on investigating how accurately we can group related discussions relying only on how their textual content overlaps as a set of topics. We see our work as complementary to these authors, as one use of our method could be to infer CVE-IDs of overlapping textual themes, which they could then use to assess risk. Indeed, CVE Details, a popular third party website to help security investigators analyze CVE data, uses text matching to infer missing CWE-IDs[4].

Dataset

Full Disclosure Corpus: Full disclosure[5] is a public, vendor-neutral forum for detailed discussion of vulnerabilities and exploitation techniques, as well as tools, papers, news, and events of interest to the community. The list was created on 9 July 2002, moved through different owners and was 'rebooted' in March 2014 as part of seclists.org. The data that we collected for vulnerability discourse was from the Full Disclosure Mailing List (2008-2016).

Each *email* is uniquely identified by a URL with year, month, and a page id. All emails contain a title, body, author's name, email address, and timestamp. Also, the corpus publishes categorizations of the emails threads by month.

Data Collection

To download the Full Disclosure corpus, we created a crawler to obtain all HTML pages, and parsers to extract the content of each email. Documents were defined as the combination of title and body, and document authors by their name + email address. For the validation dataset, we downloaded the files for the associated years, in CVRF (Common Vulnerability Reporting Framework) format[6],

[4] `https://www.cvedetails.com/how-does-it-work.php`
[5] `http://seclists.org/fulldisclosure/`
[6] `https://cve.mitre.org/data/downloads/index.html`

and parsed the Full Disclosure mailing list references contained therein.

Each document is defined as an e-mail in full disclosure, and is a vector of words (tokens). Our filtering criteria was to remove tokens that were punctuation, symbols, separators, URLs, and stop-words (common words such as "the", "it", etc.). We also filtered out tokens of 2 characters or fewer, as these were usually hexadecimal data from dumps.

Evaluation

We created our data set by partitioning all nine years of the Full Disclosure mailing list into distinct months, for a total of 12*9=108 corpus. Therefore, topic modeling is applied 108 times, one time for each corpus. The month granularity was chosen to hold a sufficient amount of documents for topic construction, while not being too coarse for manual inspection. Also, monthly partitions are already employed by the Full Disclosure list[7].

CVE Validation Dataset: To validate our topic modeling, we have relied on the CVE repository, maintained by MITRE. Launched in 1999,[8] the CVE repository documented known *vulnerabilities*[9] [43] for public usage. Each *vulnerability* is uniquely identified by a CVE ID. Each entry contains a textual description, and may also include fields specifying references, vulnerable software, version and vendors affected by it. A subset of the CVEs refer to Full Disclosure (and other security list) emails. We use these references as ground truth labels of a subset of Full Disclosure e-mails, which are then used to validate the accuracy of our topic modelling which creates topics based on the entire mailing list.

Having defined documents as e-mails, and our evaluation criteria the grouped documents being or not about the same CVE, we can now answer RQ0 and its associated instantiated questions. Specifically, what is left then is to see if there is an agreement between the IDs generated by LDA, and the IDs that were chosen, by hand, in the CVE repository.

Our method occurs in two stages, as shown in Figure 3.1: First a set of topics are created for each of the 108 corpus. Each corpus is as a group of non-labeled documents. CVE-ID labels are not used at this stage, and topics are created solely based on the textual patterns of the e-mails. Second, after the pipeline is executed, each corpus will be assigned a topic by the algorithm. It is only then that the CVE-IDs are used to evaluate if the generated topics group related e-mails or not. Because topics are constructed based solely on a bag of words approach, high accuracy in grouping related vulnerabilities would result in a much smaller search space for a security analyst who was trying to identify new vulnerabilities. We formalize the explanation of accuracy in Algorithm 2.

[7] `http://seclists.org/fulldisclosure/`

[8] `http://makingsecuritymeasurable.mitre.org/docs/capec-intro-handout.pdf#page=17`

[9] `http://cve.mitre.org/about/faqs.html#b3`

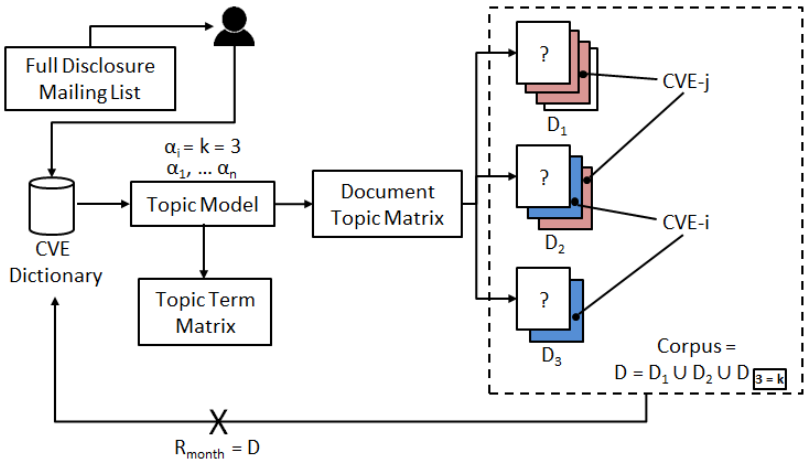


Figure 3.1: Experimental Protocol.

Algorithm 2 Calculate Accuracy

```
correct_count ← 0
total ← LENGTH(cves)
for cve in cves do
    documents ← GET_DOCUMENTS(cve)
    topics ← GET_TOPICS(documents)
    if LENGTH(topics) = 1 then
        correct_count ← correct_count + 1
    end if
end for
accuracy ← correct_count / total
```

General Statistics

Validation Dataset. The number of emails labeled with CVE-ID is relatively low until the period from 2014 to 2016, with December 2014 reaching the highest. Upon manual investigation, we identified 135 CVE entries referencing the same email, which consisted of one advisory referencing multiple CVE-IDs. To validate the monthly topic models, however, we require that at least 2 documents sharing the same CVE-ID exist, such that we can use Algorithm 2 to validate if the documents sharing the same CVE-ID were mapped to the same topic. For validation cases, Figure 3.2 shows that there are a total of 64 CVE-IDs sharing 2 documents, 11 sharing 3 documents, 3 sharing 4 documents, and 1 sharing 5 documents. This resulted in a total of 79 viable test cases for our approach.

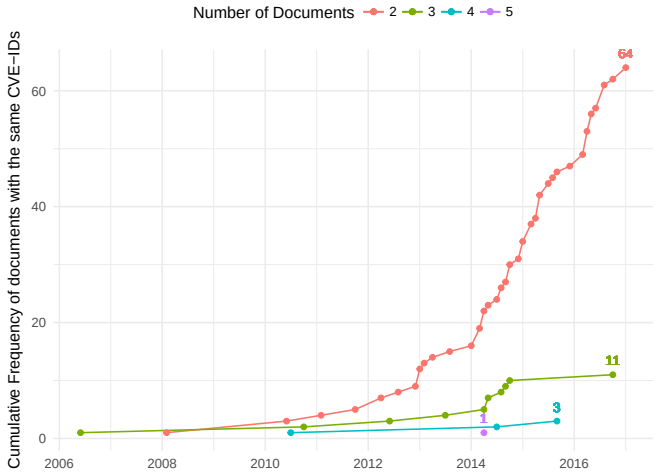


Figure 3.2: Cumulative Frequency of number of Documents with the same CVE-ID per month.

Although the CVE repository does not exhaustively list all documents in Full Disclosure, we can see that it references this email list consistently throughout the years, with a slight trend towards increased the coverage of at least 2 emails. These are the cases that our model will be evaluated against.

Results

> **RQ1**. Can vulnerability topics in free-form discourse be accurately grouped using a bag of words approach?

To answer RQ1, we evaluated the model using Algorithm 2 to obtain the proportion of documents in each month that shared the same CVE-ID and were assigned the same topic ID. Recall, from Figure 3.2, that our validation dataset consisted of a total of 64 cases, where 2 documents shared

the same CVE-ID, 11 cases where 3 documents shared the same CVE-ID, 3 cases of documents sharing 4 CVE-IDs, and 1 case of 5 documents sharing the same CVE-ID.

Using Algorithm 1 to choose the number of topics k for each of the 108 corpus, we obtained an average of 12 topics between 2008 and 2014, 8 topics between 2014 and 2016, and 5 topics after 2016. Each model had a median of 32 documents assigned to a topic between 2008 and 2012, decaying to 15 documents on average after 2012.

We observed that the model performs consistently well, as most points have 100% accuracy, except for June, July, October of 2010, February and October of 2011, and July of 2013, where the points have 0% accuracy. By counting the number of correct label assignments out of the total cases, we evaluated the model accuracy as 86%. We, therefore, conclude for RQ1 that the answer is *Yes*.

The e-mails associated with the poor performance of the model are located in topics 3 (2010_Jul_324, 2010_Jul_337) and 7 (2010_Jul_281, 2010_Jul_317). We identified a single author who posted 4 e-mails, 2 with different titles, and one including a poem (!). The model was successfully able to group the documents in topic 3, most likely due to the shared topic and to a copy of the prior e-mail being included inline, and missed the link to the other e-mail due to the poem noise. Topic 7 however, despite sharing the same title, did not share the same content as e-mails as topic 3, but they were still able to be grouped.

This short analysis served not only to identify that minimal aspects of the e-mail can suffice to establish relationship between documents to being part of the same topic, but also how conservative the accuracy measure is. For example, in this particular case, we could say that while the model did not get all the grouping correctly, it at least got 50% of it correct. However, in our final accuracy evaluation, we assigned a 0 accuracy score to this case.

Since document inspection proved to be the most accurate (although the most labor-intensive) way of understanding the behavior of the model, we defaulted to inspecting those documents associated with topics with poor performance. Upon close inspection we determined a number of reasons for some of these poorer accuracy scores:

- 2010 July, CVE-2010-3187 references 4 documents by the same author. The exploit is provided as the source code, and the content of the first e-mail is poetry. The following e-mails discussed updates to the exploit making it 'stronger'. The poetry and short e-mails concerning just updates may have biased the model.

- 2010 June 2010-2075, and 2010 October CVE-2010-3847 both contain C code, which may explain the model failing to determine that these were related to the same content.

- 2011 Feb CVE-2011-1071 primarily contains references to external links, instead of containing an actual discussion in the mailing list.

- 2011 Oct CVE-2011-3368 was an Advisory. Advisories are usually formatted in ASCII containing special characters to format the content.

Given that topic modeling relies on the word distribution content, the poorer performance in these cases is understandable. And we would like to stress that our evaluation of the model is conservative; for example, we consider, in cases of 3 or more documents, that having just one document with a different topic ID to be an incorrect assignment.

RQ2. Are the observed patterns due to random effects?

We created ten thousand random topic models (Section 3.2.5) and assessed their accuracy against the data-generated topic model, to understand if our results could achieve be due to random luck. Figure 3.3 shows the accuracy of the random models as a bar-chart, with the accuracy of the LDA models indicated by the red dot. Given the large difference between the possible results obtained through a random model and the performance of the LDA-based model, we conclude that both generalization and document assignment are not due to randomness, and therefore RQ2 answer is *No*.

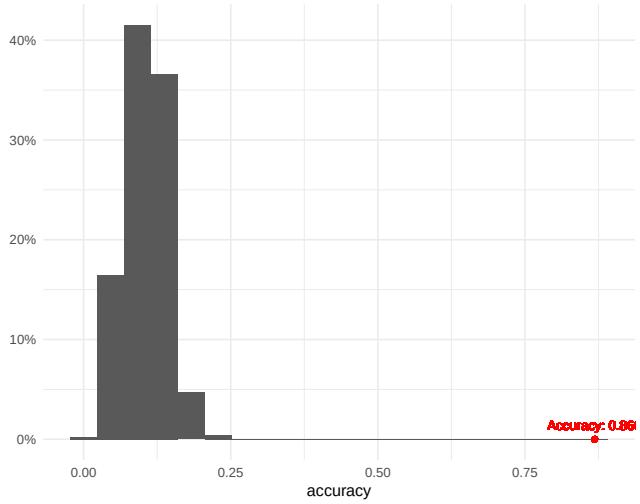


Figure 3.3: Random Model versus LDA Accuracy.

Threats to Validity

Non labeled ground truth. In our method, we used emails that were referenced by MITRE's CVE list. However, this list is not exaustive, and may bias the representativeness of our dataset compared to the entire mailing list. We believed this to be a fair trade-off (compared with hand-labeling the e-mails) on quality versus quantity, as we avoid subjectivity.

Emails which share the same CVE-ID and email Thread. We found instances where two vulnerabilities not only belong to the same CVE-ID, but also were part of the same e-mail thread. In these instances, the body of the second e-mail may contain the first e-mail, aiding the algorithm in identifying overlapping themes. However, this also means spam and other unrelated content is also repeated several times. We consider this to be a fair trade-off, and that even on communities where information about threads is not available, other metadata could be used to aid the algorithm in identifying related content (e.g. overlapping authors, referenced hyperlinks, etc).

3.3.2 Study 2 - Topic Modeling Safety Threats

As motivated in the Introduction chapter 1, the timely identification and communication of safety alerts has been on the forefront of ASRS interest since its inception [5].

Our choice to use topic modeling in ASRS differs from more recent literature in mining safety incident reports, which relies on supervised learning [61], [91], and from older literature which emphasized qualitative studies over metadata of interest (e.g. Anomalies, Human Factors, etc.) [69], [58], [34]. We share the same motivation as [56] in the choice of unsupervised learning instead of supervised learning in that this allows us to identify not only existing topics, but also emerging trends, in an automated fashion. We chose topic modeling using LDA instead of deep learning because it is a reasonably well established method for software engineering research [46]. Given this choice, we pose the following research questions:

> RQ3: Can topic modeling identify, from shuffled report sets, the original groupings in ASRS?

To be useful, topic modeling must be able to separate and group documents in a meaningful way. The standard practice in research employing topic modeling is to present a table showing the top n terms [1] in the documents,but this does not provide insight into how the documents are separated into groups. Here we leverage a novel model setup to evaluate the separation of documents using ASRS data. Being able to semi-automate or fully automate the identification of relevant themes in aerospace safety incidents reports would be useful both to ASRS analysts in issuing alerts, and to the aerospace community to perform their own explorations of the ASRS corpus.

> RQ4: Are there specific report sets that are easier to group than others?

If topic modeling may not present a perfect solution, it may be possible it can still identify certain topics well enough for practical use.

Our findings are consistent with suggestions made in a recent review of the topic modeling literature [11] regarding the need for better visualization and user interfaces for sense-making. More careful evaluation of topic modeling optimization criteria is also suggested as the more popular

optimization method, held out accuracy, may lead to less meaningful topics [13]. In addition, stability measures are also needed for assessment due to the random initialization of topic modeling algorithms [42].

Literature Review

Our work is not the first focused on analyzing aviation safety incident narratives. In older work [69, 58, 34, 20], we observed that the ASRS-related literature focused on just a specific subset of the available taxonomy. This is not ideal to provide context for new reports nor to observe changes in a given time window. For example, [69] provided a qualitative study of human errors in aviation. The data was obtained using Aircraft/Make Model involving passenger fights on Part 121 carriers and conventional aircraft, and analyzed in terms of observable error outcomes and underlying cognitive mechanisms. A related taxonomy available in ASRS is Person/Human Factors, which include labels such as confusion, distraction, time pressure, etc. A qualitative study was also published by NASA on memory errors in the cockpit of a random sample of records from 2001 [58]. Another qualitative work by Jones et al. [34] used Person/Human Factor reports labeled with "Situational Awareness" to better understand the types of situational awareness errors in aviation. The resulting 3-level taxonomy reflects errors due to perception of information, comprehension, and projection of actions, and was previously applied by the author on the National Transportation Safety Board (NTSB) accident investigation reports [20].

More recent work has focused extensively on building classifiers [67, 61, 91] for auto-labeling narratives with existing ASRS taxonomy metadata, which do not align with our primary interest, as our long-term goal is to identify *emerging* safety threats. Specifically, [67] evaluated the accuracy to query a new narrative, and assign it the correct label from Person/Primary Problem and Person-/Contributing Factors. The model was trained using ASRS data from 01/2011-01/2013, and the test data from 01/2009-12/2009. The results are not encouraging, with Person/Primary Problem having 49% accuracy, and it is also unclear why the author decided to predict past instead of future data. Oza et al [61] built a classifier on both ASRS and the Aviation Safety Action Plan (ASAP), a private dataset similar to ASRS, to classify a subset of 22 out of the 60 categories due to data sparsity. The results showed it outperformed LDA in a classification task, and manual evaluation of 100 reports suggested that the top three categories achieved agreement of at least 70%. A similar goal to classify metadata was done in [91], however, the metadata was converted in a smaller set of labels based on their risk. The authors also used two learning models in an ensemble, where one leveraged metadata, and the other the narratives.

Our approach aligns with the older literature, in that we seek to provide navigation over the processed narratives on a monthly basis to facilitate, instead of fully automate, the identification of emerging threat topics, while retaining the context of prior metadata and narratives. Specifically, when applying topic modeling to the ASRS, we identify groupings of reports, which can be under-

stood as an additional piece of information added to each report—the grouping label—however the reports can still be read individually in their original format. The automated step thus serves to reduce the search space of the reader.

Dataset

As motivated in our introduction, ASRS began as a reporting program for sharing aerospace knowledge and reporting irregularities to prevent accidents. Figure 3.4 presents the workflow within ASRS for new incoming reports. Briefly, starting from the top a reporter fills a form with details of the incident. Depending on the reporter role, a different form may be used. Common to all reports are the selection of some metadata from multiple choice (e.g. type of weather), but also a free text box, *the narrative*, where the details are provided. There may be more than one narrative associated to a report.

After a screener assesses the relevance of the report and whether other agencies should be made aware of it due to it being criminal or accident related, an ASRS analyst follows up with the reporter for with any necessary clarification, and de-identifies the report. In addition, the analyst will also include a *synopsis*, which summarizes the narrative and highlights safety concerns and, if evident, contributing factors.

Using the report database, ASRS also generates *data products* to the community. Specifically, ASRS provides a report query capability, issue alerts to organizations, performs thematic searches, publishes newsletters, etc. [76]. Relevant to our method is the *report sets* data product. These report sets are a thematic grouping of existing reports in the database[10]. In the following Method Section 3.5, we will explain how we leverage them in our experimental protocol to evaluate topic models.

Evaluation

To stay consistent with related literature in topic modeling terminology we refer to each *report's narrative* in ASRS as a *document*.

For each of the 30 choose 2 = 435 possible pair combinations of documents, we obtain 435 document-frequency matrices input to LDA which provides (for the chosen number of topics, — here $k = 2$ —): the distribution of the documents over topics (the document-topic matrix), and the distribution of topics over words (terms), commonly known as the topic-term matrix.

Figure 3.5 provides two experimental protocols. The *Real Dataset setup* reflects the more commonly used approach, where topics are presented based on the subjective evaluation of set of words. The *Toy Dataset Setup* reflects an alternative protocol where additional metadata is available for evaluation.

Real Dataset Setup.

[10] `https://asrs.arc.nasa.gov/search/reportsets.html`

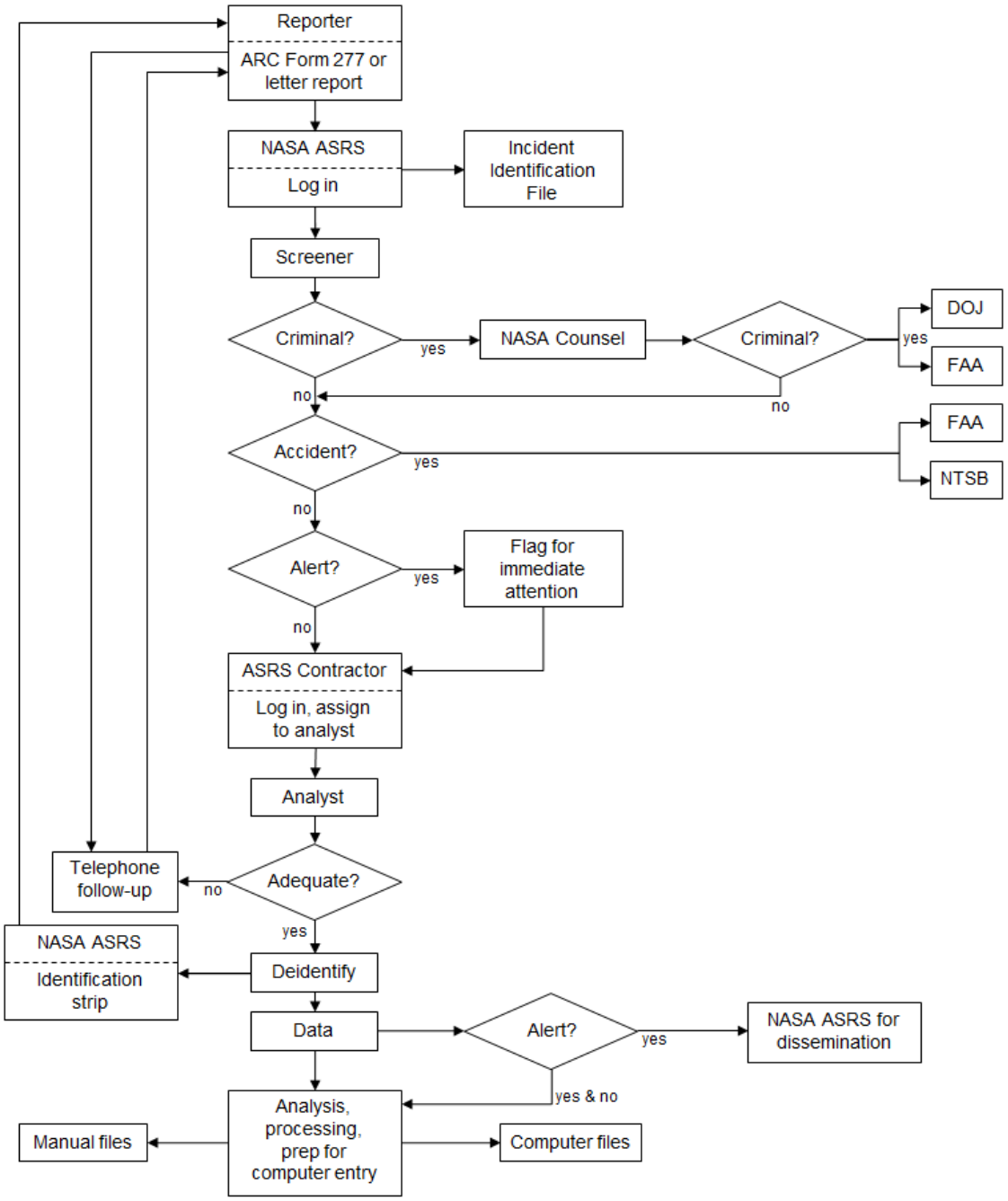


Figure 3.4: How Reports are Generated: Handling of NASA Aviation Safety Reports (ASRS) [8].

As noted in the Data Model Section 3.3.2, ASRS receives on an ongoing daily basis several reports. These reports are indexed by year and month, and thus we are able to *query* from ASRS online database the order they are received down to a month time granularity. In Figure 3.5, we showcase as an example a single month query being performed. The parameter k controls the

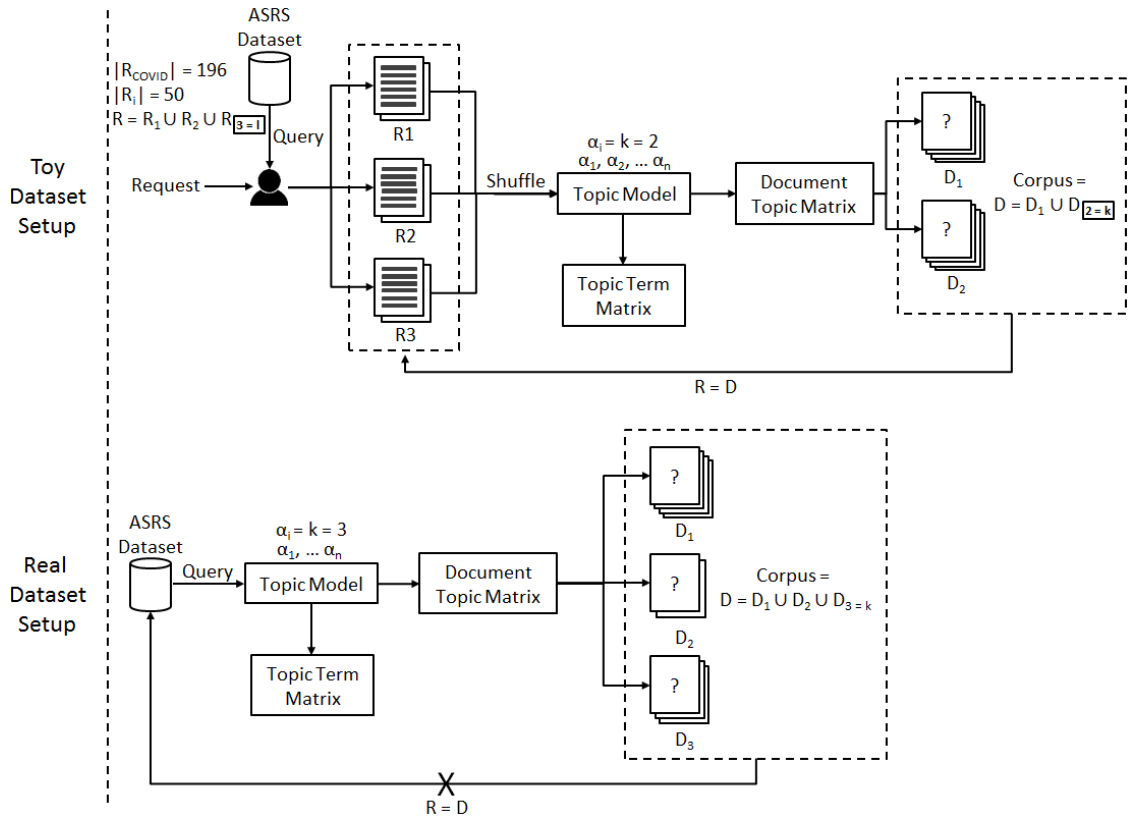


Figure 3.5: Report Set (Toy Dataset) vs Real Dataset (ASRS Database) Setup. R defines Report Sets. D defines the identified groupings by WarpLDA. k is the number of topics requires to be chosen apriori to execute WarpLDA (2 in this diagram), and l is the number of report sets used for shuffling (3 in this diagram). The goal is $R = D$, i.e. the algorithm is able to automatically identify the manual report sets groupings. In our experiment, we use l = 30, k = 2 although other values can be tested in the protocol.

number of groupings we expect the topic model to identify, $D = \bigcup_{i=1}^{k=3} D_i$, which we expect to be the number of topics to be identified if they were analyzed manually by ASRS analysts.

A validation limitation the related literature suffers in this setup, is that no manual annotation of each report's topic is known, i.e. we can't assess if $R_{month} = D$, because the partition of the reports is unknown. We thus propose the creation of the *Toy Dataset Setup* by leveraging the already available *report sets*, shown in the upper portion of Figure 3.5 to answer RQ3 and RQ4.

Toy Dataset Setup.

Report sets are manually generated on topics of interest requested to ASRS, and provided in sets of 50 reports for a given topic (e.g. bird strikes, flight deviation). The main motivation for using synthetic datasets is that they are cheap, easy to generate and the independent generative factors can be easily controlled [24]. When compared to the *Real Dataset Setup*, the number of report sets, l, and the size of each report set $|R_i|$ is *synthetic* (artificially chosen), however, the

corpus itself is not artificially generated, although a trained topic modelling algorithm could also generate report sets. Due to this distinction, we chose to name our setup as *Toy Dataset Setup*, as the reports are not artificially generated, just their groupings.

In the *Toy Dataset Setup*, we combine different report sets together (in Figure 3.5 we exemplify $l = 3$ report sets), and evaluate if the topic model is capable to recover the report set each report was originally manually assigned, given the choice of k. **While we exemplify in the diagram $l = 3$, and $k = 2$, in our experiment we tested for $l = 30$, $k = 2$.** More specifically, to answer RQ3 and RQ4 we evaluated in the 30 choose 2 = 435 possible combinations of report sets (hereafter referred as $30C2$**, or** $l = 30Ck = 2$), if the topic model was able to uncover the original groupings of report sets. Our rationale to use $k = 2$ instead of a larger number, is because we would like to assess how well the topic model performs from an easier task. That is, we assume the learning task to separate a pair of report sets ($l = 30Ck = 2$) to be easier than a large number of topics ($l = 30Ck$, $k > 2$). Likewise, we assumed that the task of separating reports when the value of $l = k$ to be easier to assess than $k > l$. For example, if $k > l$, then the additional groupings identified by k could be considered correct for identify k sub-groupings instead of only l groupings. Lastly, the threat of validity imposed in our choice would be incurred for any other chosen combination, as it is not computationally feasible to test every single possible value of l and k in lCk.

Results

We now present our findings to the three posed research questions.

> RQ3: Can topic modeling discover ASRS report sets from a collection of mixed reports by their topic?

In Figure 3.6 we present the results of all the 435 trials as a histogram, using ARI, as defined in Section 3.2.5. The top histogram uses WarpLDA to obtain the groupings. The bottom histogram replaces the algorithm by a random assignment for each report to one of the report sets of each trial. Example top, middle and bottom pairs of report sets (with respect to their ARI scores) is shown in Table 3.1.

Figure 3.6 illustrates that WarpLDA results are overall better than random assignment for most pairs. However, the algorithm's ability to identify the original topics is not ideal. Ideally, the upper histogram results would be closer to ARI = 1 (i.e. with most values shifted to the right).

We conclude that the large standard deviation (width of the histogram in the Figure) is not promising for usage on a Real Dataset Setup. In the future, we plan to use a more flexible evaluation metric to account a report may be about multiple topics, including those which were deem an incorrect assignment in this experiment.

Table 3.1: Top 5, Middle 5, and Bottom 5 Adjusted Rand Index (ARI) Report Set Pairs out of 30C2 = 435 combinations.

Report Set Title 1	**Report Set Title 2**	**ARI**
Cabin Smoke, Fire, Fumes, or Odor Incidents	Controller Reports	1.00
Controller Reports	Flight Attendant Reports	1.00
Maintenance Reports	Wake Turbulence Encounters	0.97
Flight Attendant Reports	Global Positioning System (GPS) Reports	0.93
Controlled Flight Toward Terrain	Flight Attendant Reports	0.93
Bird or Animal Strike Reports	Rotary Wing Aircraft Flight Crew Reports	0.46
Commuter and Corporate Flight Crew Fatigue Reports	Multi-Engine Turbojet Aircraft Upsets Incidents	0.45
Controlled Flight Toward Terrain	Non-Tower Airport Incidents	0.45
Pilot / Controller Communications	Inflight Weather Encounters	0.45
Checklist Incidents	General Aviation Flight Training Incidents	0.45
Rotary Wing Aircraft Flight Crew Reports	Non-Tower Airport Incidents	-0.01
Controller Reports	Parachutist / Aircraft Conflicts	-0.01
Unmanned Aerial Vehicle (UAV) Reports	Inflight Weather Encounters	-0.01
RNAV Arrival Reports	Unmanned Aerial Vehicle (UAV) Reports	-0.01
Commuter and Corporate Flight Crew Fatigue Reports	Emergency Medical Service Incidents	-0.01

Table 3.2: Ranking of Mean, Median and SD Adjusted Random Index (ARI) ordered by Mean per Topic.

Report Set Title	**Mean**	**Median**	**SD**
Maintenance Reports	0.81	0.83	0.11
Flight Attendant Reports	0.73	0.81	0.22
Cabin Smoke, Fire, Fumes, or Odor Incidents	0.66	0.70	0.18
Runway Incursions	0.61	0.66	0.19
Passenger Misconduct Reports	0.61	0.69	0.20
Passenger Electronic Devices	0.60	0.63	0.14
Controller Reports	0.53	0.51	0.28
Multi-Engine Turbojet Aircraft Upsets Incidents	0.50	0.51	0.24
NMAC Incidents	0.49	0.56	0.23
Non-Tower Airport Incidents	0.45	0.48	0.26
Commuter and GA Icing Incidents	0.43	0.41	0.25
Fuel Management Issues	0.42	0.42	0.21
Bird or Animal Strike Reports	0.41	0.40	0.16
Checklist Incidents	0.41	0.39	0.24
Controlled Flight Toward Terrain	0.40	0.37	0.29
General Aviation Flight Training Incidents	0.39	0.36	0.28
Wake Turbulence Encounters	0.39	0.31	0.25
Altitude Deviations	0.38	0.32	0.29
Parachutist / Aircraft Conflicts	0.38	0.37	0.21
Penetration of Prohibited Airspace Incidents	0.35	0.31	0.23
Air Carrier (FAR 121) Flight Crew Fatigue Reports	0.34	0.35	0.23
Inflight Weather Encounters	0.32	0.32	0.27
Commuter and Corporate Flight Crew Fatigue Reports	0.31	0.31	0.25
Pilot / Controller Communications	0.30	0.30	0.22
Emergency Medical Service Incidents	0.30	0.26	0.23
Unmanned Aerial Vehicle (UAV) Reports	0.30	0.27	0.27
RNAV Arrival Reports	0.30	0.21	0.28
Rotary Wing Aircraft Flight Crew Reports	0.28	0.26	0.25
Global Positioning System (GPS) Reports	0.27	0.19	0.28
Cockpit Resource Management (CRM) Issues	0.26	0.32	0.21

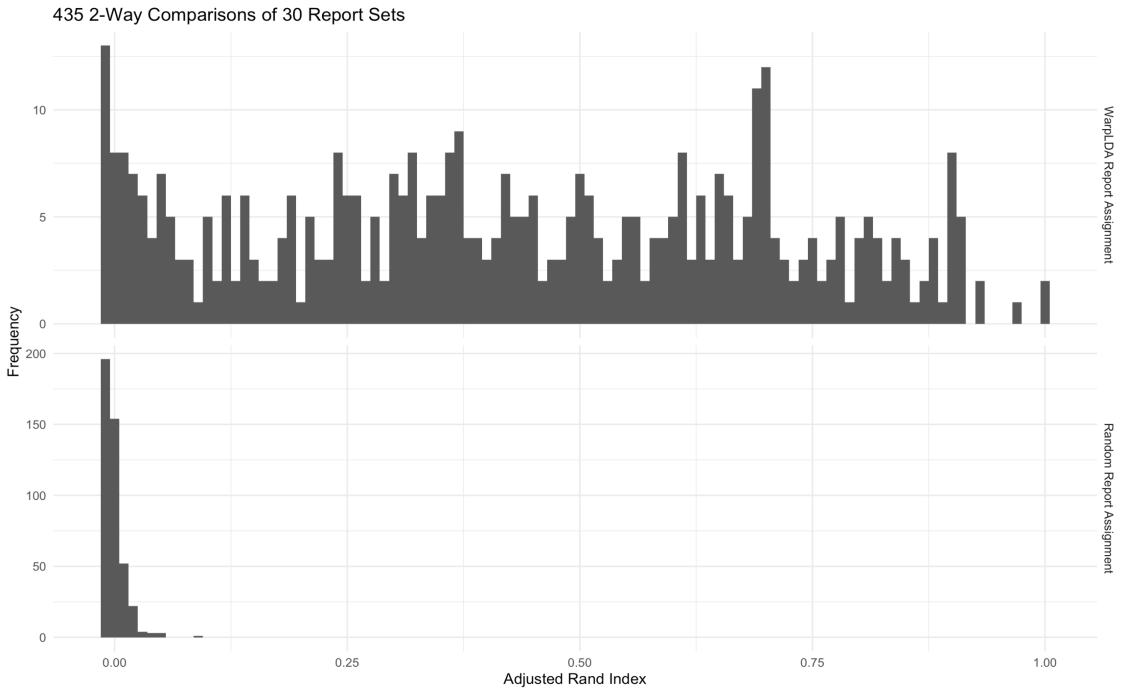


Figure 3.6: WarpLDA evaluation of distinguishing Report Sets "Cabin Fumes, Fire, Fumes, or Odor Incidents", vs "Controller Reports" when compared to manual separation of reports over multiple runs: Accuracy is defined as Adjusted Rand Index, ranging from -1 to 1.

RQ4: Are there specific topics that are easier to group than others?

A question that follows from RQ3 is whether certain report set topics are easier to be separated from *any* other report set (e.g. for problems described using a unique vocabulary). For example, does Cabin Smoke, Fire, Fumes or Odor Incidents in Table 3.1 consistently receive a high ARI score across all pairs that include it? To answer this we present the mean, median and standard deviation (SD) of each report set across all pairs in Table 3.2. We also show a histogram of the median ARI column of the table in Figure 3.7.

From Figure 3.7, we can see that there are a few report sets that are consistently separable (Mean ARI ¿ 0.8) from other report sets. In particular, we noticed from Table 3.2 that the most consistently separable report set is not Cabin Smoke (which ranks 3rd) but rather Maintenance Reports.

In future work, we intend to further investigate why report sets such as Maintenance Reports are more separable than lower performance ones, such as Cockpit Resource Management. WarpLDA relies on the distribution of the narrative's words to separate report sets. We expect that the vocabulary used in Maintenance Reports to be substantially different than what is found in other report sets. It is also possible that lower performance report sets can in fact be multi-topic, which

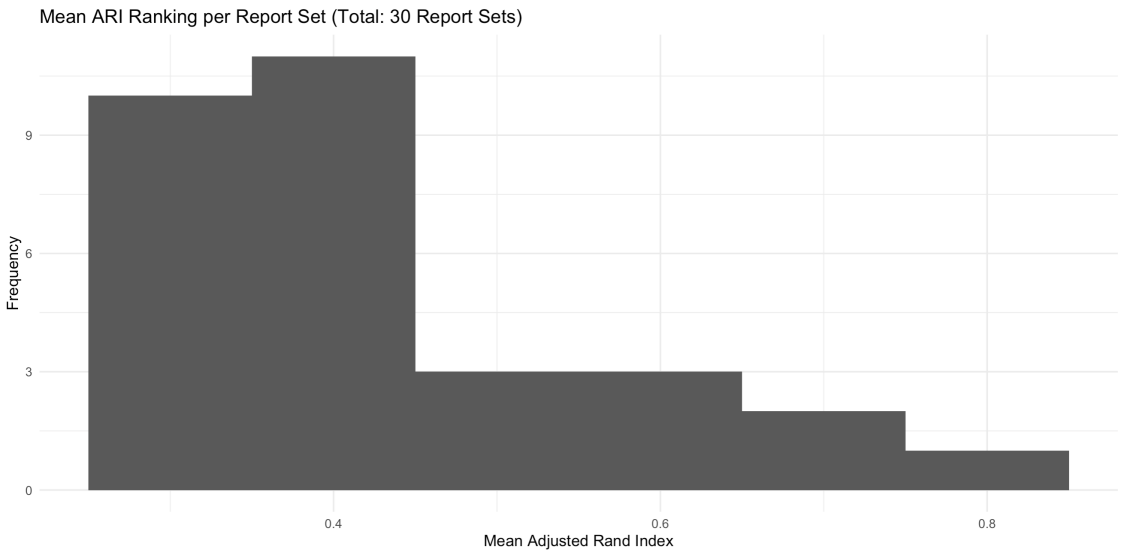


Figure 3.7: Ranking per Report Set.

violates our assumption of deterministic mapping discussed in section 3.2.4.

Threats to Validity

In Figure 3.4, we can see that an analyst must decide (diamond "Adequate?" towards the bottom) if a follow up phone call should be made before the document is deidentified. Such modifications introduce a threat to validity in the underlying distribution of the report set corpus we used in our experimental protocol (Toy Dataset Setup in Figure 3.5). However, given that ASRS intentionally attempts to keep the original narrative unchanged (in contrast to the *summary* field of the report which is made by an analyst entirely), we believe that this threat to validity is minimal.

The results presented in this work only use one implementation of LDA, i.e. WarpLDA [15], however other implementations could be used. Indeed, other non-generative machine learning methods which are discriminative such as various clustering algorithms could be applied to our experimental protocol which could provide worse or better results. We chose WarpLDA under the rationale that as the number of reports continue to grow, an efficient implementation of algorithm would be preferred in practice, but a more detailed investigation of the trade-off between performance versus accuracy could be performed in further studies utilizing our experimental protocol.

As noted in the Method Section 3.5, in utilizing the *Toy Dataset Setup* instead of the *Real Dataset Setup*, we accept the risk that the Toy Dataset Setup's underlying report set word distribution may not fully reflecting that of the Real Dataset Setup's incoming monthly corpus. We did so to be able to use the report sets already produced by the ASRS as a gold standard to evaluate the automated step results. Specifically, in the experimental protocol Figure 3.5, the parameter l, the number of report sets, could be much larger than what we have tested, and $|R_i|$ may be a

different size. It may also be possible that reports about a given topic are associated with different months or years, which may or not affect their underlying distribution compared to the real dataset setup. While this is a limitation of the generalization of the results, we believe our work provides a starting point for tests in future work.

3.4 Conclusion and Future Work

Motivated by the problems faced in both domains of the studies, we observed the use of topic modeling to automate topic discovery and navigation would be helpful. Also, according to the literature of both studies, current state of the art investigating the applicability of the method, in particular empirically, is not common.

We have identified opportunities, through additional metadata available in each domain, to construct an evaluation setup that would enable us to have a better sense of how well topic modeling performs in respect of grouping "similar" documents "correctly" by identifying what constitutes a document.

Our findings show the grouping of documents is promising when performing deterministic assignments of documents to groups, and that results are better than a random model, which provides empirical basis to investigate automatic grouping of topics *over time*, which is done in the final chapter of this dissertation.

CHAPTER 4
IDENTIFYING EMERGING THREATS THROUGH FREE TEXT: GROUPING INTERPRETABILITY

This chapter is derived in part from the Aviation AIAA'21 paper "Assessing Topic Interpretability in the NASA Aviation Safety Reporting System using Topic Modeling".

In the previous chapter, we evaluated how well topic modeling can group reports (using the document-topic matrix) by comparing against existing manually curated ASRS report sets. As noted in the related work Section 4.2, another output of topic modeling is the topic-term matrix, where each of the rows is referred as topics. This is done to exploit text-oriented intuitions, but no epistemological claims are actually made regarding these latent variables beyond their utility in representing probability distributions on sets of words [10]. In this chapter, we conduct a survey of the machine learning outputs to assess their meaningfulness and usefulness in the context of ASRS.

4.1 Introduction

There is a longstanding assumption that the topics output by topic models are *meaningful* and *useful* [13, 11]; however, evaluating such assumptions is difficult because discovering topics is an unsupervised process. There is no gold-standard list of topics to compare against for every corpus, thus requiring exogenous data for evaluation [13]. We agree with [57] that topics themselves are not the end goal, but rather using topics to improve some end-user task. To that end, we chose ASRS as a case study because the writing of *synopses* from documents is already part of their existing process. We, therefore, borrow from [57] the following research questions (the authors only address RQ1):

RQ1: Are individual topics meaningful and usable?

RQ2: Are assignments of topics to documents meaningful and usable?

Before we can discuss or measure if topics are *meaningful* and *useful*, we must define how topics are comprehended by humans, i.e. our *topic construct*. It is not practical, for example, to read topics as-is, as they often contain tens of thousands of terms each. Rather, the related literature (discussed in subsequent sections) contains a rich and diverse array of *displays* of topics [11], each of which build on different assumptions to aid in *topic comprehension*.

Figure 4.1 shows our defined topic construct based on the research literature. We can see the topic construct is broken down into 3 parts: *usefulness*, *display* and *topic comprehension* (meaningfulness). We renamed meaningfulness as *topic comprehension* because a longer and more mature field of *reading comprehension* already exists, and we consider topic meaningfulness to be a part of this. Finally, *display* is concerned with how topics are presented to users. In the following subsections, we will explore each of the three components.

4.2 Related Work

4.2.1 Topic Comprehension

According to the educational psychology literature, *reading comprehension* is a largely unanalysed construct, and many factors may contribute to high performance on standardised measures of comprehension [60]. Nonetheless, a simple model of reading comprehension from Gough et al. [25] is helpful to define the construct we wish to measure.

Gough et al. defines *reading comprehension* as a product (in a mathematical sense) of *word decoding* and *linguistic comprehension*. *Word decoding* is defined as resulting from sounding out words and context-free recognition of the words. *Linguistic comprehension* is said to be the process which lexical information, sentences and discourse can be interpreted.

They claim word decoding and linguistic comprehension can, in theory, be assigned a value from 0 to 1, and its product computed resulting in a measure of reading comprehension. This also means that, to achieve reading comprehension, one can't exist without the other. Decoding is said to be necessary, but not sufficient for reading, for if *print* can't be translated into *language*, then it can't be understood. Conversely, linguistic comprehension is argued to also be insufficient without word decoding, for knowing a language does not suffice to make one literate, as is the case with 5-year olds. They can understand spoken language, but they are unable to read it.

A more recent and extensive literature building from reading comprehension that agrees with [25] can be found in [60]. However, since we are more interested in a specific instance of reading comprehension—topic comprehension—we believe the simple model provided by [25] suffices.

4.2.2 Displays

According to Blei's probabilistic topic modelling review [11], one open problem is how to display the topics. He argues that either choosing different terms or displaying the chosen words differently could be more *effective*. However *effectiveness* is not defined. Let us now focus on *display*, and its relation to the reading comprehension model and *topic comprehension* in an example using topics by [45], shown in Table 4.1.

To the left of Table 4.1 we see one topic, whose terms were truncated to fit the table (as we said before, tens of thousands of terms are contained in the topic example with associated weights

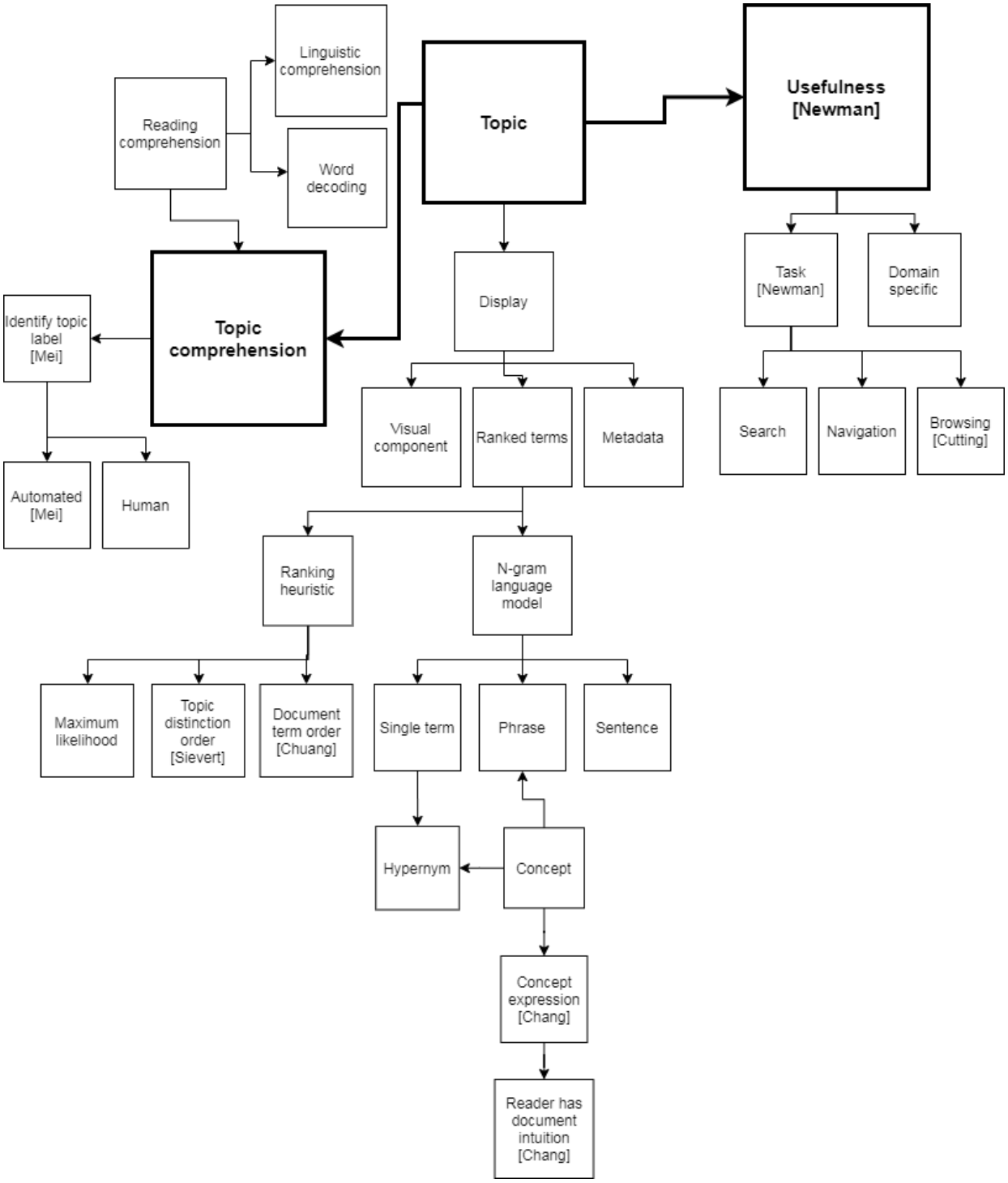


Figure 4.1: Topic Construct.

Table 4.1: One topic and possible displays [45].

Topic Example		**Topic Display**
views	0.10	*Top Terms:* views, view, materialized,
view	0.10	maintenance, warehouse, tables
materialized	0.05	*Human:* materialized view, data warehouse
maintenance	0.05	
warehouse	0.03	*Single Term:* view, maintenance;
tables	0.02	*Phrase:* data warehouse, view maintenance
summary	0.02	*Sentence:* Materialized view selection and
updates	0.02	maintenance using multi-query optimization

for every term in the corpus). To the right, we see some common kinds of *displays*.

One choice of display is presenting top terms using the maximum likelihood *ranking heuristic* (i.e. the terms are presented by order of weight, usually bounded by an arbitrary number (e.g. top 10 terms) [10]. The rationale behind this display is that topic modeling tends to place high probability on terms that represent concepts [13]. What is meant by "concept" is that the terms with high probability are hypernyms [57]. For example, color is a hypernym of red. Reading the inferred topics is used to ensure it captures the user's intuition about the documents, therefore serving as a human-interpretable decomposition of the text [13]. Here we can see *linguistic comprehension* is severely harder than *word decoding* due to the choice of display.

Different *displays* have been proposed other than *ranking terms* with *maximum likelihood*. For *visual components*, these include providing multiple topics arranged for a grid [16], or by displaying the topics based on their term distribution similarity in a plane [70]. These two authors have also proposed different *ranking heuristics*, such that the terms are ranked to facilitate topic distinction [70], or to preserve the original term orderings [16].

For displays which utilize metadata we have observed the following combinations: 1) Use of *ranked terms* with *maximum likelihood* and document sentences [17]; 2) *ranked terms* with related documents and co-occurring topics [48]. Ultimately, the goal of *topic display* is to provide a meaningful *topic label* for a reader. That is, a good *topic label* should be understandable to the user, should capture the meaning of the topic, and will distinguish a topic from all others [45].

An attempt to comprehend the *top terms* is shown under *'Human:'*. This attempt combines the terms 'materialized' and 'view' as a single concept. However, for 'warehouse' the term 'data' is inferred (the term is not shown on the left). This exemplifies why familiarity of the corpus which is summarized can be helpful, if not essential to achieve *linguistic comprehension* [45].

We borrow from [45] a more precise notion of topic comprehension here. Specifically, we say a topic was successfully comprehended if the reader is able to identify a *topic label*. In the example shown, a human *selected* terms from the 'pool of terms' with the intent to identify a topic label, but

also chose one *not* from the displayed pool of terms. An automated approach is also proposed in [45]. To identify a label, *linguistic comprehension* must be possible of the displayed pool of terms.

It is easier to understand what we mean by considering when a topic's top terms are *not* comprehensible.

In [48], the authors identified the following display of top term patterns which lead to poor choices of *topic label*: **Chained**. When every term is connected to every other term through some pairwise term chain, but not all term pairs make sense. **Intruded.** Either two or more unrelated sets of related terms, joined arbitrarily, or an otherwise good topic with a few "intruder" terms (based on [13]). **Random.** No clear, sensible connections between more than a few pairs of words. **Unbalanced.** The top words are all logically connected to each other, but the topic combines general and specific terms (e.g., "signal transduction" versus "notch signaling"). In all these cases, *linguistic comprehension is hard*, and consequently so is deciding on a *topic label*.

Because the *top terms*, and more generally *ranked terms*, are often presented as a fixed number, much smaller than the actual terms in the topic, the chosen *display* plays an essential role in *topic comprehension*, despite not being part of the optimization criteria of topic modeling. This is why authors have also presented different different *ranking heuristics* to use the weights, which included additional *visual components* [70, 16] and *metadata* [48].

Let us consider the remaining information in Table 4.1: the choice of display of *single term* (which top terms use), *phrase*, or *sentence* [45]. According to [45], *single terms* are usually too general and it may not be easy for a user to interpret the combined meaning of the terms. A *sentence*, on the other hand, may be too specific, thus it could not accurately capture the general meaning of a topic. In between these two extremes, a *phrase* is coherent and concise enough for a user to understand, while at the same time, it is also broad enough to capture the overall meaning of a topic. We can also see using Gough's [25] *reading comprehension* model that both *phrase* and *sentence* provide more guidance on *linguistic comprehension* than *single term*. Indeed, [45] notes related work in which people are required to manually perform topic labelling often prefers the use of phrases.

In summary we understand the following about *topics*:

- The goal of choosing a *display* for topics in a topic-term matrix is to improve *topic comprehension*. Specifically, a good display will allow a reader to identify a *topic label* from a pool of terms or from the person's own vocabulary, be it a single-term hypernym or phrase.

- There are various forms of *display*, but their choice in the literature does not discuss their differences, which makes it hard to assess topic comprehension as a whole. Our topic construct serves as a way to reason about the related literature from a topic comprehension standpoint.

- It is necessary, but not sufficient that a topic is comprehended through the terms. The comprehended meaning must correctly represent the underlying documents. That is, if the

topic label is seen as a concept, then the documents it is assigned are seen as the concept's expression.

As we noted at the beginning of the introduction, topics are not the goal, but rather we use topics to improve an end-user task [57]. We have yet to explore topic *usefulness*, which we address next.

4.2.3 Topic Usefulness

In ASRS, writing synopses of incoming reports is part of the existing workflow. We depict this process in Figure 4.2. The process is manual and performed per report. We emphasize here 3 steps: First a team of 3 analysts (Analysts A) during screening have to comprehend the text in a single report. Analysts may remove identifying information from the narrative and clearly indicate when information has been removed with the use of standardized codes but do not change the report in any other way.

Second, Analysts A have to summarize the entire report in a one-sentence synopsis. Third, Analyst B (which may be one of the original set of Analysts A) can then use the sentence synopsis along with the reports to quickly navigate through reports when performing information retrieval tasks, such as creating report sets or preparing newsletter issues (Callback).

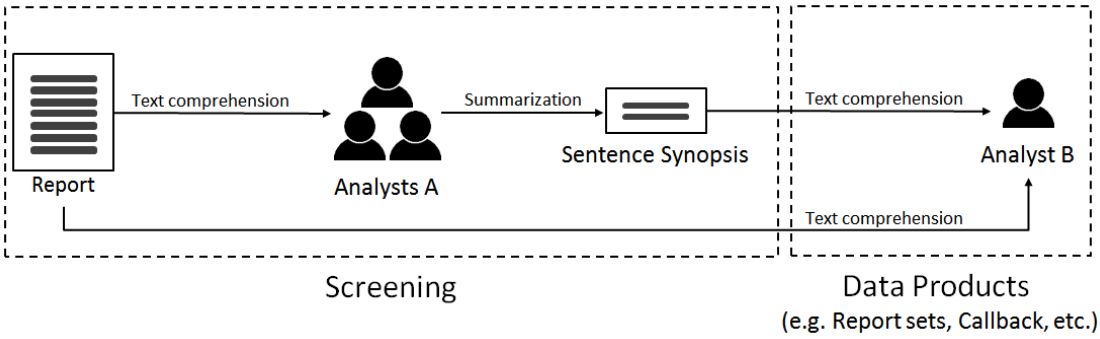


Figure 4.2: Current ASRS Report Synopsis and Interpretation Process.

For the existing synopsis writing in ASRS, we define *usefulness* as follows: Generally, the goal is to write a short concise restatement of the primary safety concern and *if* clearly stated by the reporter, contributing factors might be noted. Typically starts with the reporter's job function "Air Traffic Controller reported that".

Therefore, to be *useful*, the ranked terms provided by topic modeling would ideally include terms which suggest a) a safety threat and, b) contributing factors. The reporter's job function can be provided separately as metadata.

In Figure 4.3, we present how the *document-topic matrix*, and the *topic-term matrix* can be embedded in the ASRS process of Figure 4.2. First, the 3 Analysts A are replaced by a computer

to reflect that this step is automated. Observe also that instead of a single report, a report corpus is shown instead, as the input to the topic modelling step is a group of documents. This is useful, as the number of documents a computer can process simultaneously is arbitrarily larger than in the existing process. Moreover, this is consistent with the motivation of this thesis in Chapter 1, where it was noted that the analysis of a large collection of reports would be desirable.

The second step also changes the ASRS process. From the topic modelling step, we obtain a document-topic matrix, and a topic-term matrix. We depict the document-topic matrix in Figure 4.3 by showing Reports 1 through m (the report corpus), and the associated probabilities (w_1 through $w_{n!/(m!(n-m)!)}$) of each document to each topic in the topic-term matrix, depict as ranked terms. We emphasize here the *ranked terms* differ from the existing ASRS process in Figure 4.2, where *sentence synopses* are generated instead. This difference can decrease text comprehension of the underlying documents, indicated by the question mark in the Figure.

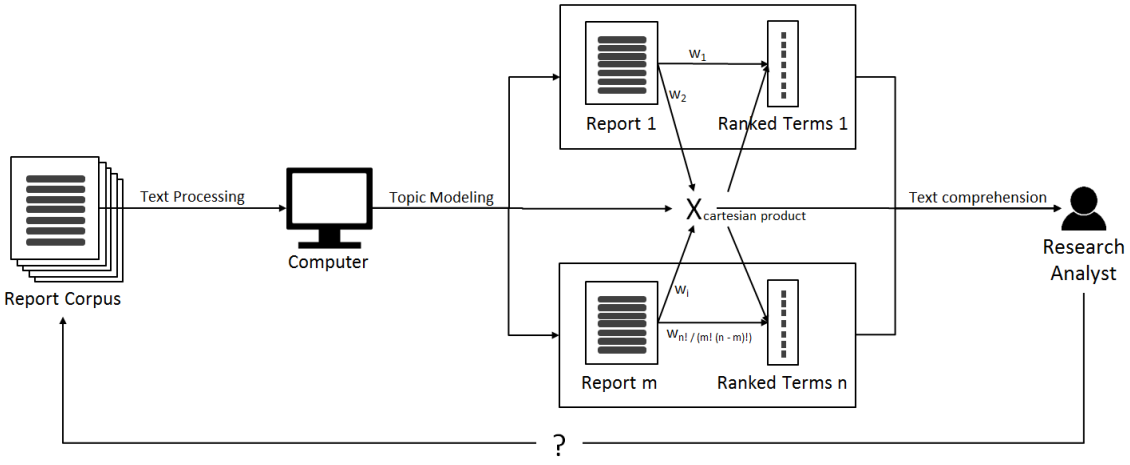


Figure 4.3: Topic Modelling output embedded in ASRS Process.

A practical limitation of using topic modeling *as-is* in Figure 4.3 occurs during the text comprehension step by the analyst. Contrary to the existing ASRS process in Figure 4.2, for each report an analyst must now evaluate several ranked terms instead of one sentence synopsis. It would be preferred if the automated step assigned one set of ranked terms for each report, or better, one set of ranked terms *per group of similar reports*. The later is shown on Figure 4.4.

As in the previous chapter, in the maximum likelihood approach each document is assigned to a single topic. Therefore, we can visualize each topic's ranked terms as summarizing a group of documents with the highest topical distribution, which is more efficient to an analyst. Other heuristics can be used to obtain the same representation, such as grouping documents of the same topical distribution. Regardless, for navigation to be possible, the analyst must be capable to comprehend the ranked terms associated to each report set.

The usefulness of topic modelling can also be seen in research on *tagging*. Indeed, the *display*

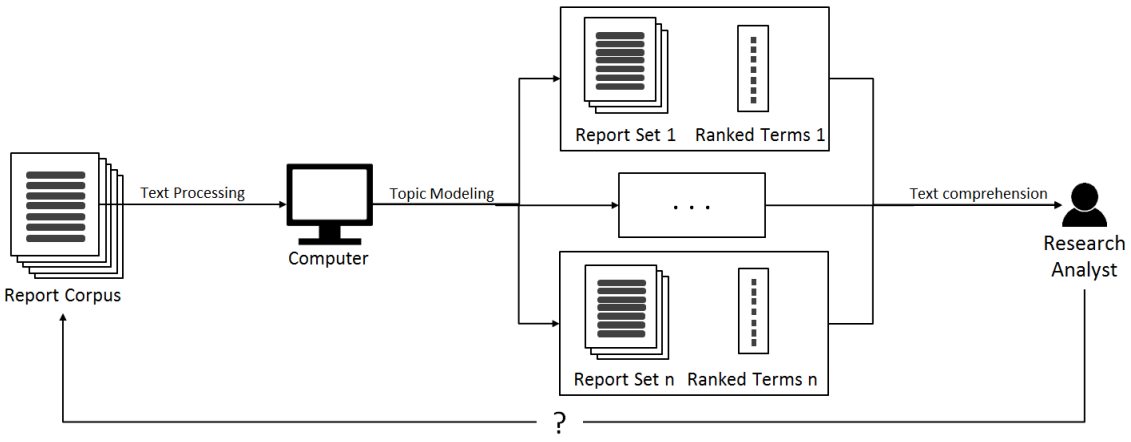


Figure 4.4: Topic Modelling output using maximum likelihood in ASRS Process.

of topics closely resembles content-based and factual tag kinds, and are also motivated by future retrieval [27]. Tags also supports similar tasks, such as navigation [47], and topic modeling has been previously proposed as an automated approach for tagging [2]. Interestingly, [2] highlighted the algorithm's poor choice of tags, similar to topic intrusion [13]. 'Bad' tags have also been attributed to polysemy, i.e. the coexistence of many possible meanings for a word or phrase [6], and a *ranking heuristic* based on relevance [38].

Alternatively the usefulness of a topic can be seen closer to *summarization*, when combining information from both topic-term and document-topic matrices to choose sentences, it is possible to create heuristics to identify representative sentences or documents. This choice of display more closely approximates literature in *summarization* [19], which is currently employed in ASRS.

We have presented what it is meant by *meaningful* and *useful* topics, by defining a *topic construct* from the existing literature, represented in Figure 4.1. We now define our survey using this construct.

4.3 Method

A fundamental limitation among the works presented is that they do not provide the rationale for the chosen questions. Because of that we do not know if they are sufficiently comprehensive when analyzing the responses. For example, in [13] two tasks, word intrusion and term intrusion, are provided but it is not explained how the authors derived the tasks. This makes it unclear if they are sufficient to assess topic interpretation, or if more tasks needed. Considering later work in [48] found intrusion to be one among other reasons experts judged topics as being bad (i.e. being chained, random, etc.), this suggests that other tasks are needed. In [48], however, various *metadata* are provided in addition to the *ranked terms* without rationale alongside the *ranked terms* display, and again it is not clear if the identified results are or not comprehensive. Similar to [57, 45] there

is also no clear set of questions that could be reused. In, [57] the authors provide some of the introductory explanation in the questionnaire, but they differ from prior work, and are more vague. In the end, none of the 4 most related works provide the actual survey used, or a construct.

In the following section 4.3.1, we address these limitations by using Groves' survey methodology [26], popularly known as the *total survey error*. In it, survey design is posed as an optimization aimed at minimizing error given the available resources (e.g. budget, people). Below we elaborate on each step of the survey design.

Because the topics are the output of a machine learning algorithm, (referred to as <set of words > in the questionnaire in the following subsection), and the survey is used to assess topics with respect to *meaningfulness* and *usefulness*, we must also provide sufficient detail of the *model setup* from which the <set of words > are derived. Figure 4.5 describes the full methodology to construct the survey.

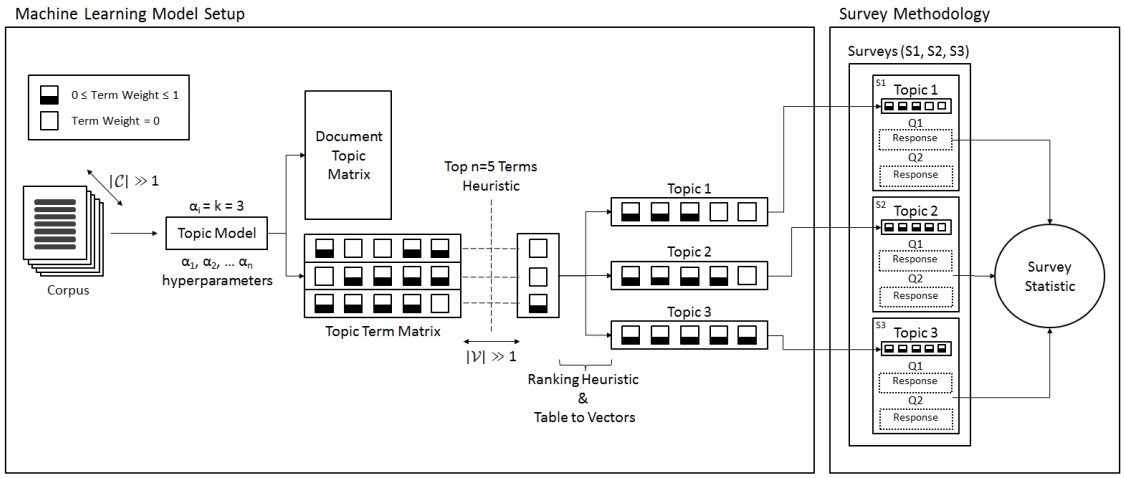


Figure 4.5: To assess topic comprehension, we present in a questionnaire the topics through different questions by applying different ranking heuristics. The choice of k = 3, n = 5, and displayed ranking heuristic (maximum likelihood) is for example purposes only.

4.3.1 Survey Design

We begin by explaining the questionnaire design (left side of Figure 4.6) using the survey method in [26], followed by the process to choose the participants (right side of Figure 4.6). Because the topics (set of words) presented to participants in the questionnaire are themselves part of the construction of the survey, as shown in Figure 4.5, we explain them in the subsequent section.

Motivation, Goal and Related Literature

Our motivation is to enable faster exploration of emerging safety threats. Ideally, comprehension only requires a set of words (as opposed to reading a large number of reports associated with the set

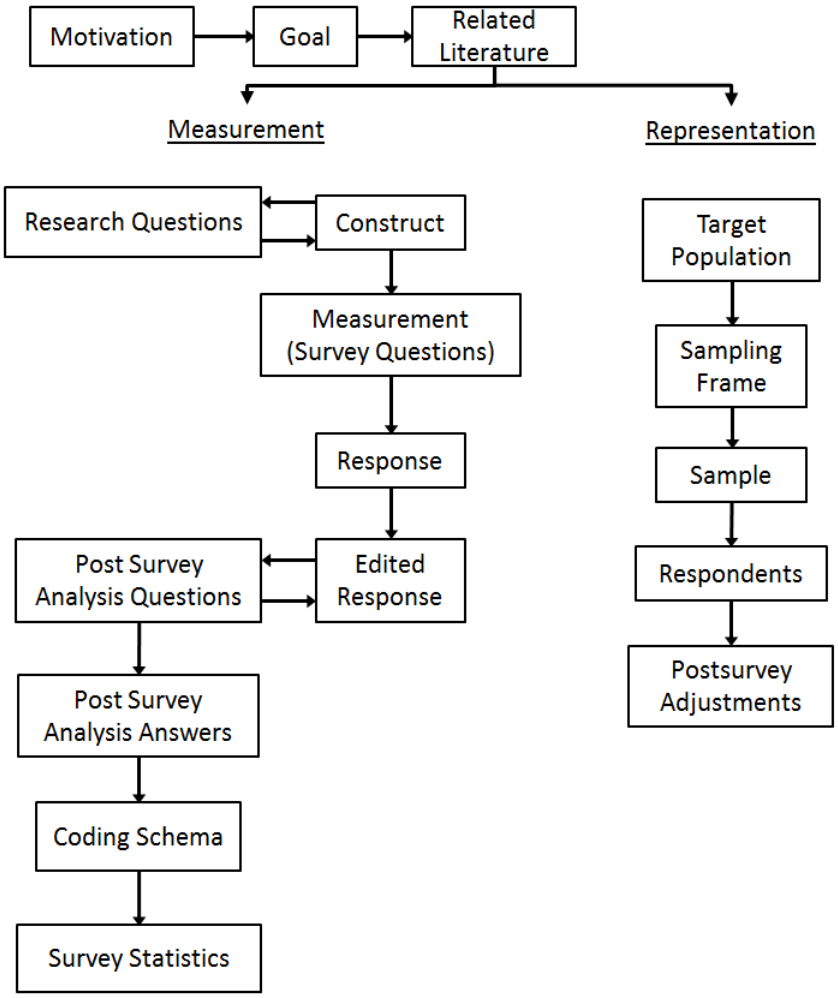


Figure 4.6: Adapted from the Total Error Survey Design Method [26].

of words). This would enable readers to select or skip entire groups of documents when searching for safety threats. Our goal is therefore to assess if the set of words are both *meaningful* and *useful* to readers. To realize our goal through a set of research questions, defined at the beginning of this chapter, which can be empirically answered through our survey, we conducted a literature review to inform our survey decisions, in particular the construct.

Research Questions and Construct

We described our construct, the elements of information that are sought by us [26, p.41], in the related work Section 4.2 of this chapter and depicted it in Figure 4.1. Specifically, we seek to understand topics from the perspective of their *meaningfulness* and *usefulness* to improve ASRS task performance. As stated in our motivation, this would ideally enable faster identification of safety report threats, by enabling readers to select or avoid, using the set of words, groups of reports for further reading of safety threat concerns.

Measurement (Survey Questions)

For our survey measures, we chose questions using only words. Specifically, we defer to future work the assessment of displays using *visual components* [71] and/or the use of *metadata* given the limited highly specialized human resources (pilots). Our measurement focuses on *ranked terms*, and the associated *ranked heuristics* reported in the literature but not empirically validated and being the more common choice of display in the literature.

Before we describe the questions to participants, we first present some context on the origin of the set of words to the survey participants. In providing context, it is important our survey does not introduce systematic biases in the responses when we attempt to explain what we mean by topic *meaningfulness*. To do so, we borrow the explanation from [57], with some modifications. The following shows our changes compared to the original (strike through represents removed text, bold represents added text, and normal text represents original text in the cited work):

> The ~~topics learned~~ **set of words displayed** by a topic model **(a computer program which automatically groups similar documents based on their content and provide set of words to describe them)**, are usually sensible, meaningful, interpretable and coherent. But some ~~topics learned~~**displayed sets of words** (while statistically reasonable) are not particularly ~~useful~~**meaningful** for human use. To evaluate our methods, we would like your judgment on how "~~useful~~**meaningful**" some ~~learned topics~~ **sets of terms** are. Here, we are purposefully vague about what is "~~useful~~**meaningful**" ... it is some combination of coherent~~, meaningful~~, interpretable, words-are-related, subject-heading-like, something-you-could-easily-label, etc.

First, we replaced the term 'useful' from the original question to 'meaningful'. We made this change, as we believe it more accurately capture the intent of the question and because we have a more precise definition of useful leveraging the domain. Second, we explained the *display* in simpler

terminology, as it is not necessarily the case that our target population is familiar with the method, nor is such knowledge relevant to our goal.

Provided with this context, we asked participants to write, if possible, in their response to the first question one or more topic labels out of the set of words provided (right side of Figure 4.5). In the second question, we ask participants to provide a mapping from the topic labels to the original set of words, so we are able to assess consensus between topic label choices, even if the choice of words are not the exact same. Because our questions and response formats are closely related, we provide the exact question after describing the response type next.

Response

The questions and responses presented here are based on the version presented to the participants. In the prototype Section 4.3.4 we discuss earlier versions.

As we motivated earlier in Table 4.1, the choice of topic label display *using text* can vary (e.g. single term, phrase, sentence). In the Table, the 'Human:' topic labels included an inferred word, *data*. And the topic label itself could be an entire sentence. The various topic labels presented in the Table may be derived from a subset of the set of words provided, in which case some of the words in the topic label end up being "noise". Ideally, per the *ranking heuristic* of maximum likelihood, these words would not occur among the top n terms.

What we see from this example is that the choice of topic label is a process rich with decisions and implicit assumptions, which could be missed on an overly structured response, such as multiple choice. We also hypothesize that the collection of those choices also relates to a participant's ability to derive meaning from the set of words presented, and also challenges existing assumptions in the related literature. For example, [45] assumes that the use of *phrases* leads to more meaningful topic labels, which in turn is used as a replacement to present topics to users instead of a set of words. When we chose to present a set of words instead of, for example, a set of randomly selected report snippets [17], we also made an assumption about topic display, similar to one of the seminal works in topic modeling [10]. Since no single response type seem to suffice to capture these decisions, we decided to make our two questions open-ended.

While the survey response was open-ended, however the response was not fully unstructured. We felt that not providing any guidance (beyond what type of answer was expected) would severely limit the comparison of responses and subsequent analysis. To address this limitation, we defined the response similar to [45] and exemplified in Table 4.1. Specifically the **first survey question / response** is defined as follows:

1. Please **identify**, if possible, in the following answer 1 or more **topic labels** that you believe the set of words are about. For each topic label, you may a) use one word from the list of provided words, b) or infer one not in this list you believe better describes; c) or a combination of few words (phrase); d) Explain the meaning in one sentence; e) other method.

2. Please **organize** the topic labels in the following manner in your answer:

   (a) one topic label per line

   (b) ranked by relevance where the first line label is the most relevant to you, and the last line is the least relevant to you in describing the set of words.

3. If **unable** to choose at least **one** topic label, please explain why you are unable to choose a topic label instead.

4. The following example 1 contains 4 topic labels, and the following example 2 provides an example answer of why not a single topic label could be identified:

   - E.g. 1
     - word
     - word1 word4
     - word k word m (inferred words)
     - Sentence explaining the meaning.
   - E.g. 2
     - I was unable to choose a topic label because word1, word4, and word5 are about one topic, and word2, word6, and word10 were about a completely different topic. The rest of the words did not make any meaningful connection to me.

The **second survey question / response** is defined as follows:

> Please copy and paste your topic labels one per line and include the words they were based on in parenthesis. For example, assume topic labels 1 and 3 in your answer were based of words not in the list, and topic label 5 was explained as a sentence. Your answer would then be in the following format, e.g.:
>
> - 'topic label 1 (word1 word5 word7)'
> - 'topic label 3 (word2 word9 word10)'
> - 'I believe these set of words are also about this subject (word3 word9)'

For clarity, and similarly to [57], we also included an example response to the question above. However, to be conservative we decided to use template example answers, as shown above (i.e. word1 word5) instead of actual words to avoid bias. Finally, we asked participants to time the start and end of their responses, to assess the difficulty in interpreting the set of words as a proxy measure of topic comprehension.

Post Survey Analysis Question and Answers

Thus far, we have presented *how* we decided to instruct participants to answer our research questions, but we did not explain *why* these instructions suffice. We included two steps to the total survey error measurement methodology [26], the *post survey analysis question*, and the *post survey analysis answer*, to make our rationale more explicit.

If the Measurement (Survey's Questions Section) and response instructions (Response Section) serve as a process to generate data from the participants, i.e. the actual responses, then the post survey analysis question and post survey analysis answer sections explicitly define what we intend to analyze from the responses as a set of inquires to the responses (post survey analysis questions), and the answer to these inquires (post survey analysis answers). Indeed, the measurement (survey questions) and response sections were guided by the post survey analysis questions, which in turn were based of our related literature, explained earlier in the chapter. The post survey analysis answers will be encoded using the Coding Schema defined in the next section, and then evaluated with survey statistics to answer our research questions.

Our post survey analysis questions, answers and associated rationale of why they help us answer the research questions are as follows:

- RQ1. Are individual topics meaningful?
  - **Post Survey Analysis Question 1.** Did the participant chose to list at least one topic label for a given topic (set of words)?
    - **Post Survey Analysis Answer.** Yes or No.
    - **Rationale.** If at least one topic label can not be assigned, then the set of words are not meaningful, and participants are expected to justify why they can not choose one.
  - **Post Survey Analysis Question 2.** How many out of the set of ten words were used by the participant in topic labels, for a given topic (set of words)?
    - **Post Survey Analysis Answer.** An integer between 1 and 10.
    - **Rationale.** When evaluating if topics are useful, we hope all the words contribute to one or more meanings (one or more topic labels). We thus consider a topic more meaningful, if more of the presented words are used in at least one topic label. This is also indirectly an evaluation of the maximum likelihood assumption: The words that surface to the top are assumed to be meaningful when ranked in this manner.
  - **Post Survey Analysis Question 3.** How many topic labels of the total identified by a participant has least one inferred word for a given topic?
    - **Post Survey Analysis Answer.** A real number between 0 and 1.

    - * **Rationale.** If topic labels have to be inferred from past knowledge of the participants, then it is more likely other participants will be unable to identify the inferred topic labels (i.e. if they do not have said previous knowledge). Ideally, the set of words should be self-contained for topic labels to be inferred.

- **Post Survey Analysis Question 4.** What were the word ranks of the used set of words in topic labels for a given topic?
    - **Post Survey Analysis Answer.** A vector of integers ranging between 1 and 10.
    - **Rationale.** According to the maximum likelihood heuristic commonly used in the literature, the higher the probability weight (and therefore the rank of the word), the more descriptive of the word it is. We thus expect that participants derived topic labels from higher ranked words than lower ones.

- **Post Survey Analysis Question 5.** How long it takes participant to choose topic labels for a given topic?
    - **Post Survey Analysis Answer.** A measure of time per question, in minutes which includes both question and answer in the response.
    - **Rationale.** Manual labeling of topics (set of words) is said to require a lot of mental effort [45]. We expect *meaningful* topics to require less effort. While the first time of completion may be biased due to the complexity of instructions, we expect the remainder of the responses to consume only the time to assign topic labels, as the instructions are the same other than the different sets of words.

- RQ2. Are the topics useful?
    - **Post Survey Analysis Question 6.** Are **coded labels from participant's topic labels** associated with a safety threat and/or confounding factors that lead to the safety threat? (i.e. is the topic label useful by our construct definition?)
        - * **Post Survey Analysis Answer.** A binary vector for each topic label of Yes or No responses.
        - * **Rationale.** To be operationally relevant to ASRS, the identified topic labels should relate to safety concerns, as discussed earlier in our research question definition.

- RQ3. Is the assignment of topics to documents meaningful?
    - **Post Survey Analysis Question 7.** Are the **coded labels from participant's topic labels** semantically related to the report set title and description from which the set of words were derived?

    - **Post Survey Analysis Answer.** A vector of Yes/No responses, where the number of elements of the vector equals the number of topic labels.
    - **Rationale.** While a participant may infer any number of topic labels (meaningfulness), the inferred topic labels still have to correctly represent the set of underlying documents of the topic, which is reflected on the report set title and description the topics were derived from.

- RQ4. Is the assignment of topics to documents useful?
  - **Post Survey Analysis Question 8.** Are **coded labels from participant's topic labels** semantically related to the report set title and description from which the set of words were derived *and* safety threats?
    - **Post Survey Analysis Answer.** A vector of Yes/No responses, where the number of elements of the vector equals the number of topic labels.
    - **Rationale.** While a participant may infer any number of topic labels (meaningfulness), which are safety threats (usefulness), the inferred topic label still has to correctly represent the set of underlying documents of the topic (i.e. a participant inferring an incorrect safety threat would not be a meaningful assignment).

For responses in RQ1 which indicate no topic label can be chosen, we compare it to related literature of bad topics [48].

Coding Schema

For each of the post survey analysis questions, we specified 1 of 2 types of post survey analysis answers: Yes/No (1 or 0), or a number ranging from 1 to 10. In addition, the post survey analysis answer may be single-valued or a vector. Since the post survey analysis answers are already numeric or binary, no further coding is necessary.

Target Population

Our target population are ASRS Analysts and Pilots, as both are familiar with the ASRS database and reports. When explaining *usefulness* in Section 4.2.3, we described the process in which ASRS analysts could benefit by having topic modeling improve the process of summarization, and subsequently navigation. While pilots do not summarize, they would still benefit of a navigation capability if the topic labels are useful.

Sampling Frame

The frame population is the set of target population members that has a chance to be selected into the survey sample (e.g. the sampling frame of a target population of U.S. adults could be a

list of telephone numbers) [26, p.45]. In this study, they will all be either NASA employees, or affiliates on NASA contracts. We refer to them as subject-matter experts (SMEs). Many of them have experience across several domains of aviation, including air traffic control, dispatch, etc.

Sample

We use convenience sampling on the sampling frame. We have asked contacts who have helped us in the past to volunteer as respondents.

Respondents

A total of 13 people volunteered to participate in the survey, two of which did not follow-up. We first presented two participants with the survey for prototyping, and and the remainder nine to take the final version of the survey.

Postsurvey Adjustments

We manually transcribed all responses to the two surveys questions to a tabular format, modifying only grammar errors on words. The tables created contained the following columns:

- Raw Table 1
  - Participant ID
  - Survey ID
  - Participant Topic Label
  - Term

Participants could have multiple topic labels and multiple terms per topic label. Topic labels which did not include set of words associated to them were eliminated from the table.

The information for time was annotated as follows:

- Raw Table 2
  - Participant ID
  - Survey ID
  - Start time of Q1 (Hour:Minute)
  - End time of Q1 (Hour:Minute)
  - Start time of Q2 (Hour:Minute)
  - End time of Q2 (Hour:Minute)

From the two tables above, the following tables were constructed for analysis:

- Analysis Table 1
  - Participant ID
  - Survey ID
  - Participant Topic Label
  - Does topic label contain at least one inferred word? (PSAQ3)
  - Is topic label useful? (PSAQ6)
  - Is assignment of topic label to documents meaningful? (PSAQ7)
  - Is assignment of topic labels to documents useful? (PSAQ8)
- Analysis Table 2
  - Participant ID
  - Survey ID
  - Duration to answer Q1 (PSAQ5)
  - Duration to answer Q2 (PSAQ5)
  - Difference in minutes between answering Q1 and Q2
- Analysis Table 3
  - Participant ID
  - Survey ID
  - Participant Topic Label (PSAQ1)
  - Term (PSAQ2)
  - Term Ranking (1 to 10) (PSAQ4)
  - Coded Labels

Survey Statistics

We used summary statistics (mean, median, standard deviation) to look for inter-rater agreement among the participants.

4.3.2 Topic Model Setup

.

We now discuss how the topics, indicated in the survey as <set of words>, are generated by the machine learning model setup in Figure 4.5. As shown in the Figure, there are many decision steps, starting from what documents are input to the algorithm, to the number of words which will be displayed. We decided to adopt the same evaluation model setup as in the previous chapter, section 3.3.2.

We made one modification to pre-processing, however. Stemming is not usually used in topic modeling when there is ample data, because the unstemmed terms generally result in topics that are more easily interpreted [57], which is our goal in this chapter.

Because we have 435 possible pair combinations of the 30 report sets (30C2), and in the previous chapter we also chose k = 2 topics per report set, we have a total of 435*2 = 870 topics. This means that, for the same report set (e.g. fire and fumes), we will have multiple topics (sets of words), in decreasing adjusted rand index, indicating how well the topic model was able to recover the original report sets. What is left then is to decide which of the 870 topics we should choose the set of words to present to participants.

In the related literature of this chapter, no other work used a model setup with ground truth (as topic modeling is an unsupervised learning algorithm). Therefore, we use for comparison another criterion, which is the document-topic probability assignment as basis for our rationale.

We have found different criteria for selection of documents based on their assignments to topics. These consisted of random selection [57], highest probability set of words being compared against lowest probability [13] and subject matter expert convenience sampling [48] (i.e. according to the authors topics were first pre-filtered by subject matter experts based on their areas of expertise).

Since there was no apparent consensus we decided to use a 'highest to lowest' criteria of ARI. Our rationale was that, in set of words with higher ARI there is less 'leaking' of words from a different report set into the topic. Therefore, if participants are unable to comprehend the set of words, we can more confidently attribute it to the words produced by the algorithm to represent the topic, instead of poor grouping performance. A drawback of this approach, is that extremely high values of ARI (e.g. 0.96%) are unlikely to be observed in practice. We defer to future work understanding the implications of lower ARI scores, as this would require a larger number of subject matter experts than what we have available for this study.

Having decided the criteria for the sets of words, a second decision is the choice of words (top n criteria), and more generally whether or not *only the set of words* is to be presented or additional metadata is required. For example [10], where LDA was introduced, used only the set of words. Cutting et al. [17] used a combination of set of words and first sentences of a randomly selected document. Minmo et al. [48] presented the set of words, and additional metadata for each topic, namely a) the most common sequence of two or more consecutive words assigned to the topic, b)

the four topics that co-occur most with that topic, the most common IDF-weighted words from document titles, thesaurus terms, organization names associated with the corpus, journal titles associated with the corpus, and a list of the highest probability documents for the topic. Recall other non-textual displays also exist, as previously discussed in the literature review section of this chapter. We decided to adopt only the set of 10 words, as this is the most popular representation of topics in applied studies (see [1] for a table containing several studies). Finally, we decided to only use the set of words, and no additional data, as the less data users have to process, the more efficiently they can browse through topics.

Lastly, since each participant was expected to spend no more than one hour in the questionnaire, a total of 5 set of words were presented to each participant. Using the criteria above, the report sets, and the set of words derived from them were:

1. **Cabin Smoke, Fire, Fumes, or Odor Incidents.** smoke — cabin — fire — odor — captain — passengers — fumes — cockpit — maintenance — attendant

2. **Controlled Flight Toward Terrain.** pilot — terrain — atc — feet — runway — airport — captain — flying — turn — visual

3. **General Aviation Flight Training Incidents.** tower — pilot — feet — traffic — plane — student — pattern — turn — airport — downwind

4. **Penetration of Prohibited Airspace Incidents.** tfr — airspace — approach — controller — pilot — airport — departure — turn — heading — call

5. **Passenger Electronic Devices.** passenger — fire — smoke — cabin — seat — battery — passengers — attendant — phone — captain

4.3.3 Response Diagrams

To answer Research Questions 2, 3 and 4, through the post survey analysis questions (PSAQ 6 through 8) we required subjective interpretations of the topic labels. To provide sufficient evidence so that others may assess our subject's interpretations, we represented the participant topic labels and their assignments to the set of words, as provided by the participants, in Figures 4.8,4.9,4.10,4.11,4.12. Figure 4.7 provides the starting point of the method used to color the diagrams, resulting in Figure 4.7. In this section we will use both figures to explain the method, and discuss the results of the PSAQs in the Results section.

Recall all five surveys contain the same set of instructions, and only the set of words provided are different. Therefore, the five diagrams follow the same structure. At the bottom, the set of 10 words are provided in *the same order* participants read them (and thus capture the notion of 'top terms' towards left, and 'bottom terms' towards the right of the set of words). Preserving the order

allow us to visualize how participant topic labels use the top and bottom terms. In the middle, *all* participant topic labels are shown, with the exception of sentences. We chose to exclude sentences because participants would at most use one sentence, and often the sentence would attempt to explain the chosen terms and topic labels. They also included most of the words. Between the set of words and topic labels are the mapping between the topic labels and words *provided by the participants*. We manually colored the connections which the set of words overlapped the most between topic labels to facilitate examination. Finally at the top, we provide *our* coded labels, and also our assignment to the participant topic labels. Our coded labels use both our subjective interpretation of the participant topic labels, and the mapping to the set of words the participant provided. Here, we leverage our coloring of connections to aid in sense making of the inferred underlying concept of the topic labels, and minimize our own assumptions biasing our coded topic labels. For example, in Figure 4.8, one participant chose as topic label 'May be a topic'. We found said topic label to be 'Fire' related not because we understood that from 'May be a topic', but because the participant indicated the term 'Fire' influenced its choice.

To the left of the figure, we included the report set's title and description the set of words at the bottom were 'derived from' (recall as per our method, we chose the highest Adjust Rand Index partitions for the given set of words). We only use the report set and description information for RQ3 and RQ4.

To begin the coloring, our first step is to start from the set of words at the bottom, and look across all the uncolored dashed edges in the constructed mapping between topic labels and set of words provided by the participants. In particular, we look for the highest number of words that have topic labels in common. In Figure 4.8, we see that is the case for the words 'Smoke', 'Fire' and 'Odor'. We then replace the dashed lines touching these three words by a solid colored line (red in this case). Subsequently, we color either the topic label they touch outline or filling (but not both) with the associated color. This means that in Figure 4.8 topic labels with a red outline *all share the same set of words in common, 'Smoke', 'Fire' and 'Odor').*

One could ask, why not also include 'Cabin' in the list of words to color with a red solid line in this case. Observe that doing so would decrease the number of topic labels with red outline, as there are fewer topic labels that share these terms. Since our goal with coloring is a means to an end to identify related topic labels that may be ambiguous to comprehend from their meaning, but which share words, we strike a balance between the most words in common and the most topic labels shared.

We again tried to perform step 1 to other words. For example, take the word 'Fumes', which instead of having three words in common, is by itself. By following the same method, we again made the outgoing dashed lines from Fumes into solid orange lines, and colored all topic labels it touches, this time the filling. The choice of filling or outline is irrelevant. Our interest is mainly to facilitate the human eye to quickly capture overlap. With the affordance we have available

here, boxes, that allow us to only use the outline and filling color. We can see in Figure 4.8 that associating 'Fumes' with a different color, allow us to see a topic label that was fire related ('Fire problem'), but did not share 'Fumes'.

Once most overlaps were manually identified, and topic labels we then moved to Step 2. We started from the now colored topic labels, and colored the remaining dashed lines touching them the color of their box. If the box has two colors, we subjectively chose the color we believed represented the main theme based on the topic label meaning and the words used. Our goal with step 2 was to identify missing connections were completely overlooked. Once this process was completed, we re-evaluated if the participant topic labels were properly shuffled in a way they are adjacent to related themes. Steps 1 and 2 continued until we believed we achieved a reasonable comprehension of the topic labels and the coded labels properly reflected them.

The process described above is not an algorithm; it is intended to facilitate the final evaluation of our PSAQ answers, which in turn relies on the coded labels. It is entirely possible in reproducing the method that some other coloring is obtained, but we expect the coded labels to be similar. In future work we hope to evaluate via multiple inter-rate agreement of other coders. In the end we hope, through the coloring, it is easier to evaluate whether the coded labels are indeed related to the participant topic labels they derive from, and subsequently the words, as **only the coded labels are used to answer RQ2, RQ3, and RQ4, and not the coloring method)**.

4.3.4 Prototype

In this section, we discuss the changes made and not made on the survey based on the prototype phase. In total, there were 4 versions of the surveys. We present 3 surveys discussed in this Section on Appendix C. The differences between the 3rd and 4th versions are minimal and also discussed in this Section.

Version 1 - Open-ended and Multiple Choice Format

The first proposed version of the survey sought to assess participants understanding of the set of words both in an open-ended and multiple-choice format, in addition to their confidence in doing so. This survey was then reviewed by three senior researchers, two of whom were aviation subject matter experts. The following shortcomings were identified:

1. It was not clear what would constitute the "right answer" to the open-ended question of the set of words meaning, nor the results could be interpreted. For example, if the responses were a range of different descriptions like "Fume Event", "Smoke in the Cabin" and "Captain is reporting a fume event in which the passengers detect an odor in the cabin", it would be hard to even compare responses for consensus without avoiding extensive subjective interpretation.

2. While the rationale behind including a confidence question was to serve as a proxy if participants were making informed interpretations or randomly guessing the description of meaning, "confidence" itself is a participant specific subjective metric. If one respondent has high confidence that 'smoke in the cabin' is the true meaning, and another has high confidence that 'fume event' is the true meaning, given said subjectivity, it was not clear how these results would be interpreted, as both are partially correct.

3. To get a more quantitative response, we considered asking participants to rate the relevance of each key word or rank order them. We factored **the mapping of the answers to the original set of words as an important factor to keep in the next version of the survey**, however, an additional step was still necessary in the instructions to account when participants understood multiple meanings from the set of words.

4. Finally, a problem with the multiple choice question is that it could taint the answers to the following set of words.

Version 2 - Task Format

Based on the lessons learned from Version 1, the second proposal removed the questions asking for confidence and providing multiple choices due to the aforementioned limitations.

To facilitate the evaluation of research question 1, that of the meaningfulness of the set of words for the participants, we wanted to use open-ended questions while providing more structure to the responses, so a "right answer" and "comparable answer" could be viable. An additional requirement is that this structured answer acknowledged that any number, from one to multiple meanings, were acceptable. Based on related literature [45, 57], we chose to ask participants to choose **topic labels**, instead of fully open-ended answers.

Topic labels, similar to *tagging*, served to provide a succinct representation of one more meanings to respondents. In addition, the related literature [45] acknowledged, and even claimed certain representations of topic labels (phrases), were superior to others (word, sentences). We chose to provide respondents with the option of choosing the way they preferred to define their topic labels, and considered that one of the objects of our analysis, or post survey analysis question, was to answer if the set of words were or not meaningful (RQ1).

Having defined a format to obtain multiple interpretations in Question 1 as topic labels, we then requested, in Question 2, that participants provide the **mapping to the set of 10 words** for each question label, so they could be comparable. In this version, the full traceability of the survey, from research questions to summary statistics was reviewed once again, as presented in Figure 4.6.

Version 3 - Task Format Test Run

A major concern of Version 2 of the survey was the complexity of its instructions. We considered three tools to present our survey: Google Forms, Survey Monkey[1] and Qualtrics[2]. Unfortunately, only Google Forms allowed us to download the collected responses without a paid license, and the only suitable format for the answers were open-ended questions. For example, if Google Forms allowed for a question format of "connecting two words with a line", then the burden of instructions in Question 2 could be lessened.

To evaluate if the survey instructions could be comprehended the instructions provided, we performed a test with two participants (we emphasize that the participants in this survey is a highly specialized cohort, and hence the number of participants was very limited).

We asked the two participants, provided with the consent form and survey and without further instructions, to answer the questions so we could validate the clarity of the instructions. The test-run participants were presented with only one of the five surveys, as the instructions were the same, other than the sets of words presented for the assignment of topic labels.

The conclusions are separated based on a) the feedback provided directly from the participants, and b) conclusions derived based on their responses. Moreover, to identify gaps in the identified limitations of the prototype and obtained feedback, we created the following checklist to each identified problem:

- Participant X - Problem i
    1. What was the observed inconsistency between survey instruction and participant response?
    2. Why did the participant not follow the instruction / did something different than expected?
    3. What portion of the survey led to the confusion?
    4. How can we, the ones applying the survey, can address it?
    5. Should we incur the risk of the change, or keep the survey as is? Why?

The following is the list of identified problems using the template above. Note the template was not presented to the participants to avoid restricting our biasing their feedback:

- **Problem 1** - Test Participant 2 Feedback
    1. Participant did not answer question 2.
    2. Participant understood to only answer question 2 if all words in the topic label were inferred. If only partial or none were inferred, then skip.

[1] `https://www.surveymonkey.com/`
[2] `https://www.qualtrics.com/core-xm/survey-software/`

3. The survey instruction “If you infer a topic label using words not listed in the word set above (See 1 item b above)” references item 1b, which states “b) infer one word that is not in this list”.
4. 1) Make question 2 mandatory, 2) reword the instruction from “b) infer one word that is not in this list” to “b) infer one word one or more words that is not in this list”.
5. We accepted change 1) to make the question 2 mandatory, however we rejected change 2), as Problem 3 solution also resolves change 2).

- **Problem 2** - Test Participant 2 Feedback

1. Participant answered one topic label in Question 2 with 4 words in parenthesis, but also commented it could have used 10 words.
2. Participant was not clear whether to ”only specify the dominant words between the parentheses or list all the words”.
3. In question 2, the sentence ”and the words they were based on in parenthesis” does not specify any criteria for word dominance.
4. 1) Specify only dominant words should be included.
5. We rejected 1) as it would be hard to specify a threshold for when a word becomes dominant, and potential risk of biasing since the choice of words in question 2 is also one object of analysis in this work.

- **Problem 3** - Review of Test Participant 2 Answers

1. Participant did not answer question 2.
2. If topic label only includes non-inferred words, participant will assume question 2 should not be answered. The interpretation of the instruction is correct, but the instruction itself is incorrect. Another test participant included more words in parenthesis than those in the topic label even if no inferred words occur.
3. Question 2 says “If you infer a topic label... then...”, which excludes the case “If no words are inferred”.
4. The sentence “If you infer a topic label using words not listed in the word set above (See 1 item b above), or used a sentence to explain (See 1 item d)” was removed. The subsequent sentence was also edited “Please ~~specify these~~**copy and paste your** topic labels one per line and **include** the words they were based on in parenthesis” (words in **bold** are new).
5. We decided to replace the sentence, as it also addresses Problem 1.

- **Problem 4** - Review of Test Participant 2 Answers

1. Participant submitted one topic label of question 1 per response for a total of 5 responses.
2. Google Forms display long answer text boxes as a single line. Participant did not realize keyboard key "enter" could be pressed to obtain more lines. In addition, Google Forms option "Show link to submit another response" was enabled by default, suggesting to the participant more responses were encouraged (and hence one topic label per response).
3. The instruction "1) place one topic label per line" was deemed impossible by the participant due to the Google Forms interface.
4. The Google Forms option "Show link to submit another response" can be disabled. In addition, a "Sample Response Template PDF" can be included to participants. To avoid bias, the sample answer still uses the format 'word1 word3' etc instead of being concrete.
5. The Google Forms option to show links was disabled, and the Sample Response Template PDF" included. The PDF file responses are included in the Appendix. We note it was not possible to enforce one response per participant in Google Forms, as that requires identifying information of the participant (e-mail), which would violate the anonymity specified in the IRB protocol.

- **Problem 5** - Test Participant 1 Feedback

1. All followed, but instructions were claimed to be ambiguous to participant.
2. The participant claimed it would be helpful to give more concrete examples.
3. No specific sentence was pointed out by the participant.
4. A handout with actual examples was proposed by the participant, with set of words and responses.
5. We chose as a compromise to provide a handout but with template answers, as it helped address Problem 4. However, we did not introduce an actual example due to potential risk of biasing the participants in our post survey analysis questions.

In addition to the above, one of the two test participants remarked they had previously read one of our papers [63]. While this did not pose a threat, as the related work did not concern the survey, we found it important to account for the level of familiarity of the participants on both the method (topic modeling), and the domain (aerospace) and data (ASRS) used in the experiment, to assess if a higher level of familiarity with them could influence the responses.

Version 4 - Time Questions

The final survey was presented to an additional senior researcher for commentary. We decided to add an additional instruction for participants to measure how long they took to answer each question, to gauge comprehension difficulty.

A strong threat to validity to the measure of time is that the participant response would include both the time to read and comprehend the instructions and to choose topic labels, whereas the latter was the true object of interest. However, this threat to validity is alleviated given that the same instructions were presented to participants for each of the 5 sets of words. We expect that the later responses will more accurately reflect the time for participants to assign labels due to instruction familiarity.

4.4 Results

We now answer our four research questions, through our post survey analysis questions. We re-emphasize that the goal of the four research questions is to *incrementally* evaluate the participants' understanding of the set of words. In RQ1 and RQ2, our goal is to assess participant comprehension, but we do *not* assess if said comprehension matches the reports' narratives from where the set of words were generated. In RQ3 and RQ4 we *do* evaluate if the comprehension reflects the narratives, indirectly via the report sets.

RQ1. Are individual topics meaningful?

Post Survey Analysis Question 1. Did the participant chose to list at least one topic label for a given topic (set of words)?

Only one participant in the second survey concerning controlled flight towards terrain stated it was not possible to identify a topic label: "This set could lead to an infinite number of possible reports, it does not identify a clear single topic". Therefore, overall participants were able to draw from experience to suggest topic label meanings.

Post Survey Analysis Question 2. How many out of the set of ten words were used by the participant in topic labels, for a given topic (set of words)?

Survey ID	Mean	SD
1	8.78	1.64
2	7.37	3.11
3	7.33	3.31
4	7.89	3
5	8.11	2.47

Table 4.2: PSAQ2 - Average number of words from the set of words used in participants topic labels.

We can see in table 4.2 that on average participants used 7 or 8 out of the 10 words made available across all surveys, with a small standard deviation. Ideally, all words would be of use of the participants, but there is a consistent use of their majority.

Post Survey Analysis Question 3. How many topic labels of the total identified by a participant has least one inferred word for a given topic?

The average number of topic labels which contained at least one infer term per participant is shown in Table 4.3, which quantifies PSAQ3.

Survey ID	Average Percent of Topic Labels
1	68%
2	58%
3	50%
4	77%
5	46%

Table 4.3: PSAQ3 - Average percent of topic labels containing at least one inferred term.

Ideally, the set of words provided should be sufficient to explain meaning, or otherwise tacit knowledge may be require to relay or even identify what the underlying set of reports may be.

Post Survey Analysis Question 4. What were the word ranks of the used set of words in topic labels for a given topic?

Survey ID	Survey Average of Term Rank Medians
1	4.89
2	5.64
3	6
4	5.1
5	5.18

Table 4.4: PSAQ4 - Survey Average of Term Rank Medians.

In table 4.4, we can see the median ranking of the terms chosen on each topic label, averaged out across all participants in the survey. Ideally, we would expect the index to be closer to 1. The index of the terms nonetheless ranges within the set of 10 words, which suggests top 10 terms is appropriate to the display of topics, or potentially even less for inference of the underlying documents.

Post Survey Analysis Question 5. How long it takes participant to choose topic labels for a given topic?

In table 4.5, survey 1, as expected, has a longer time duration to answer the questions compared to the remaining surveys, as participants are learning the instructions, with an average difference of two minutes more. The remaining questionnaires indicate an average between 4 and half minutes to identify different topic labels of every question.

Overall, we consider the answer to RQ1 to be yes, topics are meaningful. However, we can see there is room for improvement as participants have to tap into tacit knowledge by inferring terms to convey meaning, the words used are not among the top terms of the 10 words, but in the middle,

Survey ID	Mean Time Q1	Mean Time Q2	Time Between Q1 and Q2
1	7	4.5	44
2	4.44	3.3	0.77
3	5	2.7	0.44
4	4.9	2.5	0.44
5	4.33	2.66	0.33

Table 4.5: PSAQ5 - Participant mean time to answer survey questions and time between survey questions.

and it takes at least 4 minutes to interpret just 10 words.

RQ2. Are the topics useful?

Post Survey Analysis Question 6. Are **coded labels from participants' topic labels** associated with a safety threat and/or confounding factors that lead to the safety threat? (i.e. is the topic label useful by our construct definition?)

Using the coded labels we found participants were able to identify safety threats in three of the five surveys. Specifically, Figures 4.8 (coded labels fumes and fire) and 4.12 (coded labels fire and cabin smoke) were related to fire safety threats. Figure 4.9 captures safety threats with the coded labels terrain and runaway excursion. In Figures 4.10 and 4.11, no codes are associated to safety threats.

We note some words in the diagrams led to more different coded labels than others. In Figure 4.9, the term 'runway' was associated both to the coded labels 'terrain' and 'runway excursion', but also 'go around', 'landing operations', and 'phase of flight'. In Figure 4.10, the term 'traffic' was associated to the coded labels of 'conflict', 'tower' and 'airport traffic'.

These terms are interesting from the perspective of terms generated by an algorithm, because they help provide context to the meaning of coded labels when combined to other terms.

Therefore, the answer to this question is mostly (3 out of 5) yes.

RQ3. Is the assignment of topics to documents meaningful?

Post Survey Analysis Question 7. Are the **coded labels from participants' topic labels** semantically related to the report set title and description from which the set of words were derived?

In this and the following RQ, we evaluate if the identified coded labels reflect the underlying reports from which the set of words were derived. Since examining every report in ASRS would be time prohibitive, we compared the coded labels instead to the report sets title and abstract (shown

in every Figure to the left). Note in this RQ our interest is to see if the coded labels are related to title and/or abstract, *regardless* of being safety threat related.

We can see across all report sets, at least some of the coded labels relate exactly to the main theme of the report set, which is highlighted by the colors in the coded labels.

In Figure 4.8, the coded labels "fire" and "fumes" relate to the title and description of the report set, Cabin Smoke, Fire, Fumes or Odor Incidents. This is expected, given the set of words generated by the algorithm also contain these words and others (e.g. odor). In Figure 4.9, terrain and runway excursion to controlled flight towards terrain.

Figures 4.10 and 4.11 coded labels, although not related to safety threats, properly relate to the report set meaning. In Figure 4.10, the coded label "instructional flight", is associated to the report set of training reports, and Figure 4.11 matches exactly one of the two types of incidents covered by the report set ("tfr", which stands for temporary flight restriction).

Finally, in Figure 4.12, although the coded label associated to safety threat was "fire" (RQ3, PSAQ6), the title and description of the report sets are matched by the coded label of "phone battery".

We can see in particular to the report sets of instructional flight in Figure 4.10 and in phone battery 4.12 that a single word led to the proper inference of the report set, respectively 'student' and 'battery'.

The answer to RQ3 is therefore yes, participants were able to interpret, from the set of words, the meaning of the report sets.

RQ4. Is the assignment of topics to documents useful?

Post Survey Analysis Question 8. Are **coded labels from participants' topic labels** semantically related to the report set title and description from which the set of words were derived *and* safety threats?

In this research question, we combine the answers to RQ3 and RQ4, i.e. are there report sets in which the coded labels are safety related and also relate to the report set?

Based on the answer to RQ3, we know the coded labels from Figures 4.8, 4.9, and 4.12 convey safety threats. Here, we are interested in knowing if they also convey the safety threat of the report set. As we noted in RQ3, Figure 4.12 conveys a safety threat, but is not the main theme of the report set. Therefore, only the report sets of Figures 4.8 and 4.9 convey safety threats associated to the report set.

Since the minority of the surveys satisfy PSAQ 8, the answer to RQ4 is no. While one of the report sets does not actually convey safety threats (General Aviation Flight Training Reports), which would even the surveys in half, we had hope that participants would be able to identify from the set of words incidents concerning electronic devices, and incidents associated to penetration of

prohibited air space. The fault, however, seems more associated to the set of words displayed than participants being unable to derive meaning associated to the report sets (as shown in RQ3).

4.5 Threats to Validity

In the process of tabulating the responses, we observed the following threats to validity: Out of the X topic labels assigned to participants, 12 topic labels were removed.

Some responses seem to follow the example template response too closely. For instance, the response would have the format topic label 1, topic label 2, ..., topic label n, followed by a sentence.

Participants sometimes included in their response additional judgments of the potential benefit or lack of for the method (briefly explained at the start of the survey for context). However, participants judged their value in respect to the method in identifying keywords to search for reports, which was not the intended goal of the study.

We also received requests for clarification for some of the instructions, despite not observing confusion during the prototyping. However, we were careful to not provide any example responses in providing clarification to avoid biasing participant answers.

Finally, the majority of the 9 participants required one reminder to complete the survey, which introduces a risk that the responses may have been rushed.

4.6 Conclusion and Future Work

In this chapter, we assessed the *meaningfulness* and *usefulness* of topics, and their assignments [57]. We concluded that participants were not only able to assign topic labels (RQ1), whether safety threat related or not (RQ2), but also that these topic labels were in general related to the report set title and description they were derived from (RQ3). We believe these results are promising, given participants guessed the topic label only from 10 terms instead of having to read the 50 report narratives for each report set (which the set of words were derived from). While navigation of report sets by theme using words seems promising, understand whether the underlying reports are associated or not is unclear (RQ4).

In future work can explore other useful properties of topic modeling, such as whether topics are *discriminative* and have *high coverage* [45]. And we can include other metadata to evaluate if more information can be derived from reports without requiring reading the narratives.

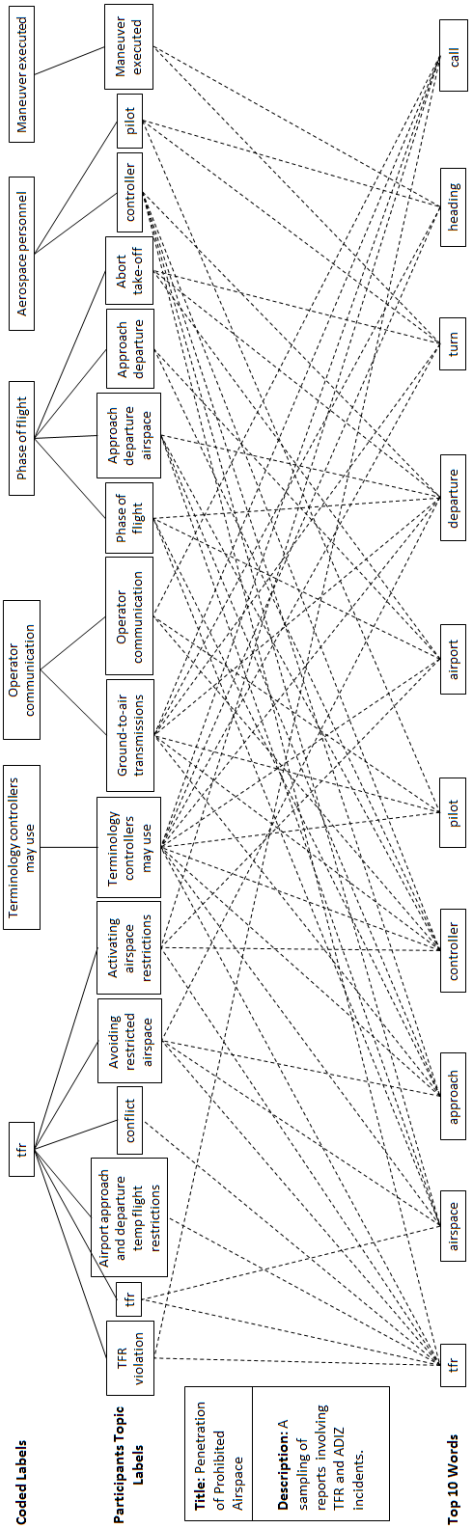


Figure 4.7: Report Set: Controlled Flight Toward Terrain before coloring step.

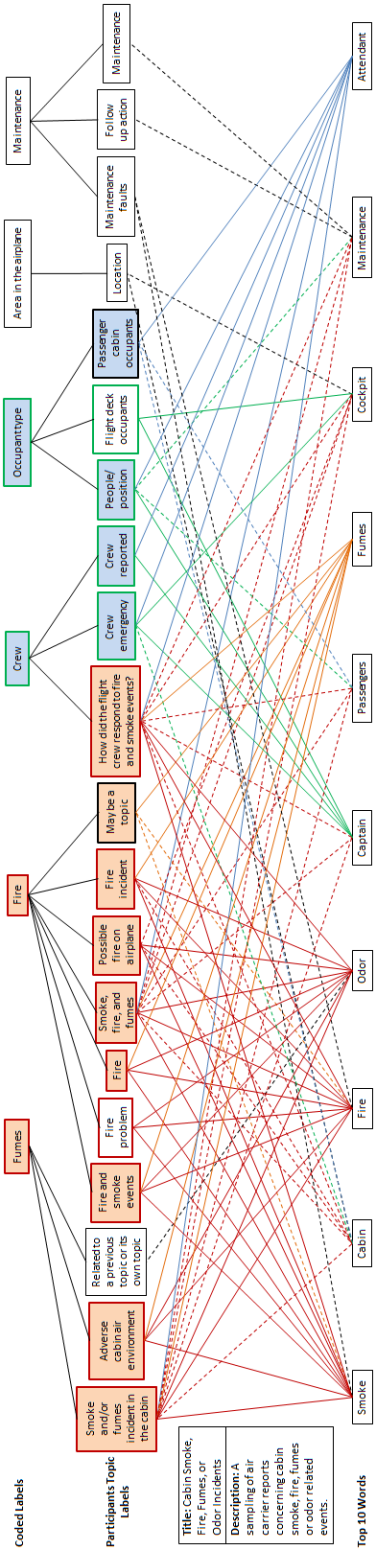


Figure 4.8: Report Set: Cabin Smoke, Fire, Fumes, or Odor Incidents.

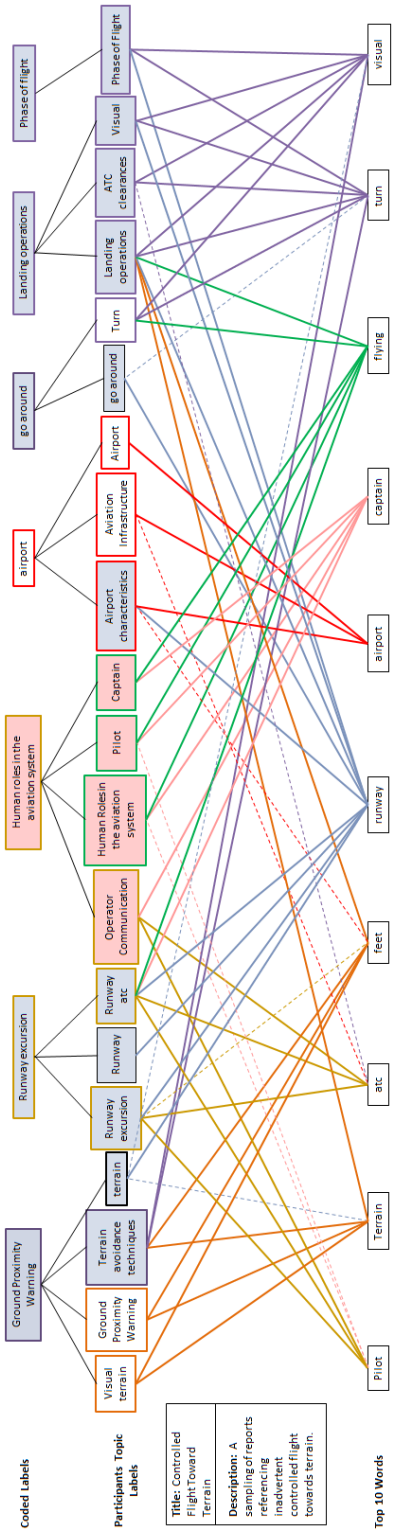


Figure 4.9: Report Set: Controlled Flight Toward Terrain.

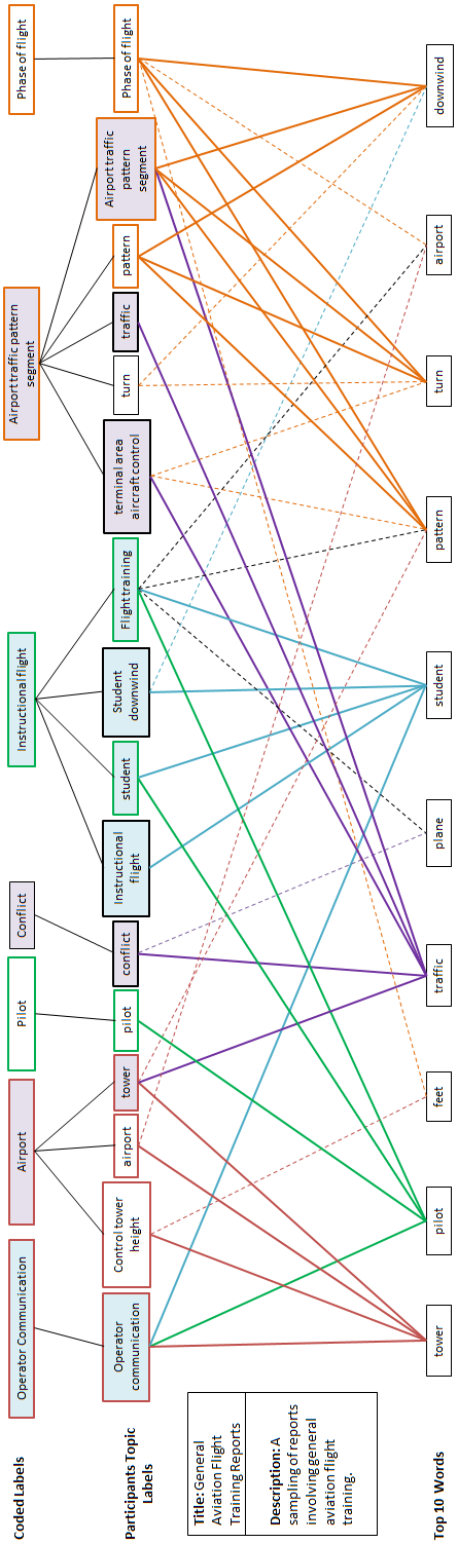


Figure 4.10: Report Set: General Aviation Flight Training Incidents.

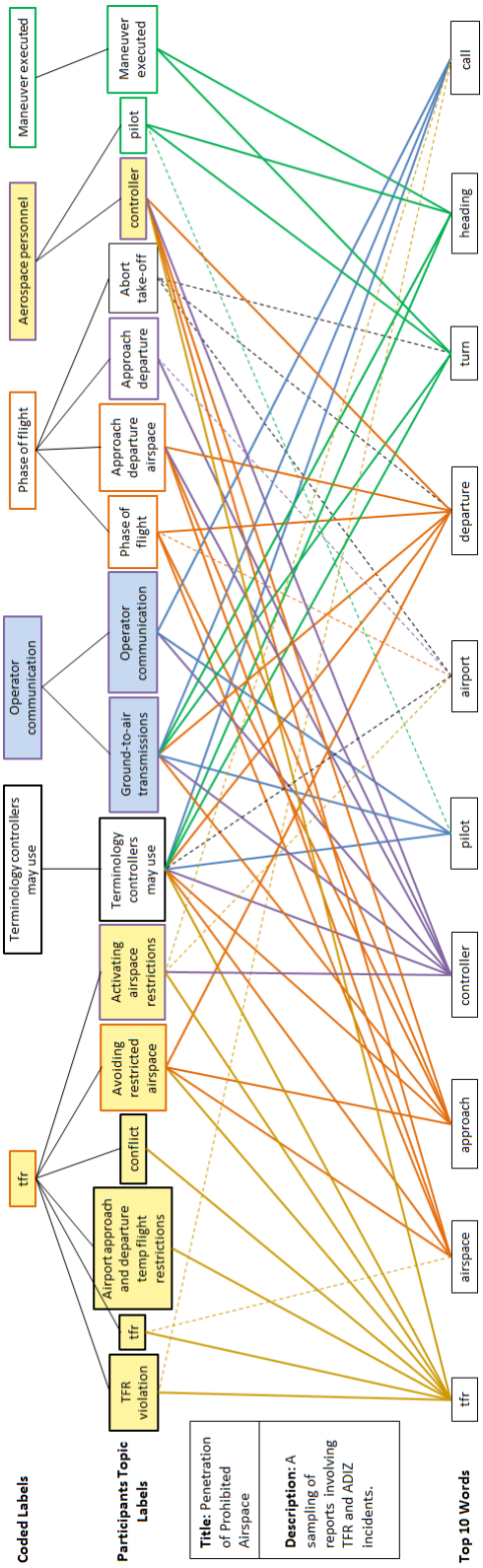


Figure 4.11: Report Set: Penetration of Prohibited Airspace Incidents.

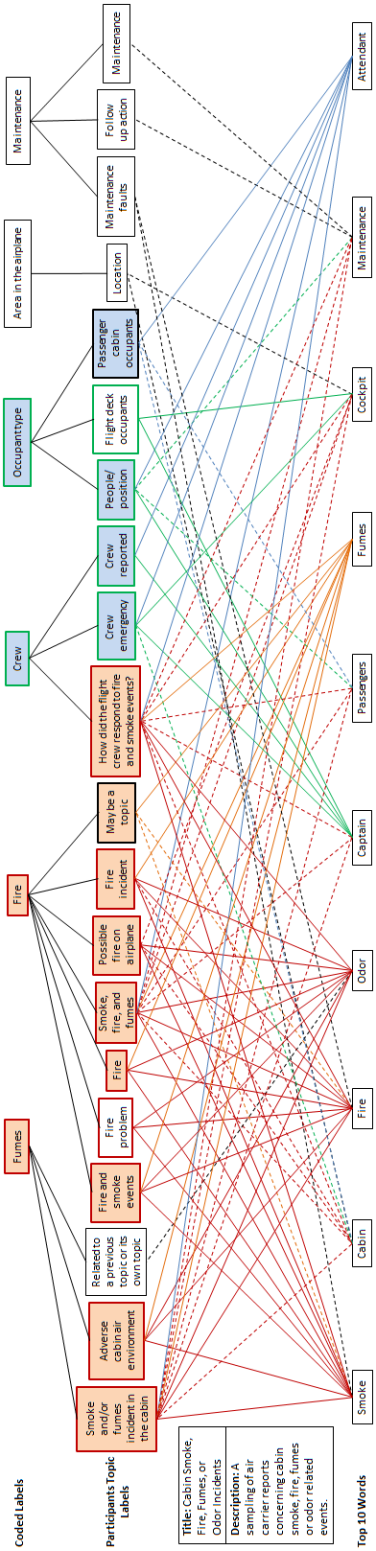


Figure 4.12: Report Set: Passenger Electronic Devices.

CHAPTER 5
IDENTIFYING EMERGING THREATS THROUGH FREE TEXT: GROUPING OVER TIME

In the previous Chapters 3 and 4, we evaluated the performance, meaningfulness and usefulness of groupings using topic modeling for *point in time* datasets. We improved on the state-of-the-art by leveraging external data as ground truth, whether generated by ASRS (Report Sets), CVE, or via subject matter experts (Questionnaire Responses). The different methods also evaluated the two different parts of a topic: the document-topic matrix, which defines the groupings, and the topic-term matrix, which defines the terms to comprehend the groupings. In this last chapter, we leverage both matrices and methods using COVID-19 as a use case to identify if emerging safety threats can be identified *over time*. We do so by using *topic flow*, introduced in this chapter, to construct *timelines*, that separate the evolution of different topics over time.

5.1 Introduction

Since March/April 2020, COVID-related reports in ASRS screening increased dramatically, suggesting 'waves' of safety concerns that have changed week-by-week from March to the present. We use COVID-19 as a use case study to evaluate if topic flow can identify themes that have changed over time. Little metadata is available on these reports, making a strong case for leveraging narratives and thus the application of topic flow. Specifically, we pose the following overarching research question:

> Can *topic flow* detect emerging COVID-19 topics?

We chose COVID-19 Reports for a few reasons. First, this is a very relevant and current theme to facilitate the identification of reports. Second, the vocabulary surrounding COVID-19 is not as specialized as other topics within ASRS, making it a good candidate for evaluation in the scope of this work, as non-specialist readers can relate to the identified themes. Lastly, ASRS has created a large report set for this theme, which reflects its importance to aviation safety. This report set contains 1213 COVID-19 related reports to date, all of which are in 2020, making it a strong candidate for evaluating our method. To empirically evaluate our overarching research question, we define the following research questions:

> RQ1: Are there timelines whose terms clearly suggest the prevalence of COVID-19 reports on its topic's top terms?

Timelines constitute the evolution of themes in text over time. Ideally, stakeholders should be capable of navigating through a set of terms alone to identify the emergence of COVID-19 topics.

RQ2: If there exist timelines where the top terms suggest a prevalence of COVID-19 reports, what proportion of the Report Set's COVID-19 reports do they account for?

Our goal here is to understand how many COVID-19 reports in the *COVID-19 Report Set* a user would have detected via the top terms in a given timeline, i.e., the precision and recall of the *report set* in a given timeline. We emphasize the Report Set here, because the COVID-19 Report Sets are not necessarily all COVID-19 reports in the year 2020. We find the comparison pertinent nonetheless, as our goal is to automate the identification of emerging safety threats, and thus can serve to compare if the Report Set did not exist, how many of the reports upon further inspection of the timelines would have been uncovered.

RQ3: Are there timelines that consist exclusively of reports from the COVID-19 Report Set?

Here we ask a slightly different question than RQ2: Rather than focusing on the proportion of reports from the COVID-19 Report Set, we focus on the proportion of said COVID-19 Reports per Timeline, i.e., how "saturated" is each timeline with COVID-19 reports from the Report Set. Ideally, we would like to know how many of *all COVID-19 reports* in 2020 each timeline has, but this information is not available (as it would have required the exhaustive manual inspection by ASRS of each incoming report). Nonetheless, if we can identify timelines that contain close to or 100% of the COVID 19 reports from the report set, the effect is the same. Unique COVID-19 timelines are highly valuable from a user standpoint, as the user will have a higher precision of identifying COVID-19 reports upon further inspection.

5.2 Method

In Figure 5.1, what we mean by *analyzing points in time*, is the independent evaluation of each month's topics. In the Figure, this is equivalent to evaluating, in each experiment, a vertical line of rectangles. Each rectangle represents a topic and therefore contains a document-topic matrix and a topic-term matrix. We attempted to understand in previous chapters if each month could be grouped correctly and if the set of terms were meaningful and useful using report sets and subject matter experts through a questionnaire.

In this chapter, our focus is on empirically deriving the *edges* between the rectangles shown in Figure 5.1, and devising a method of evaluation to assess if the evolution of a topic can indeed be observed.

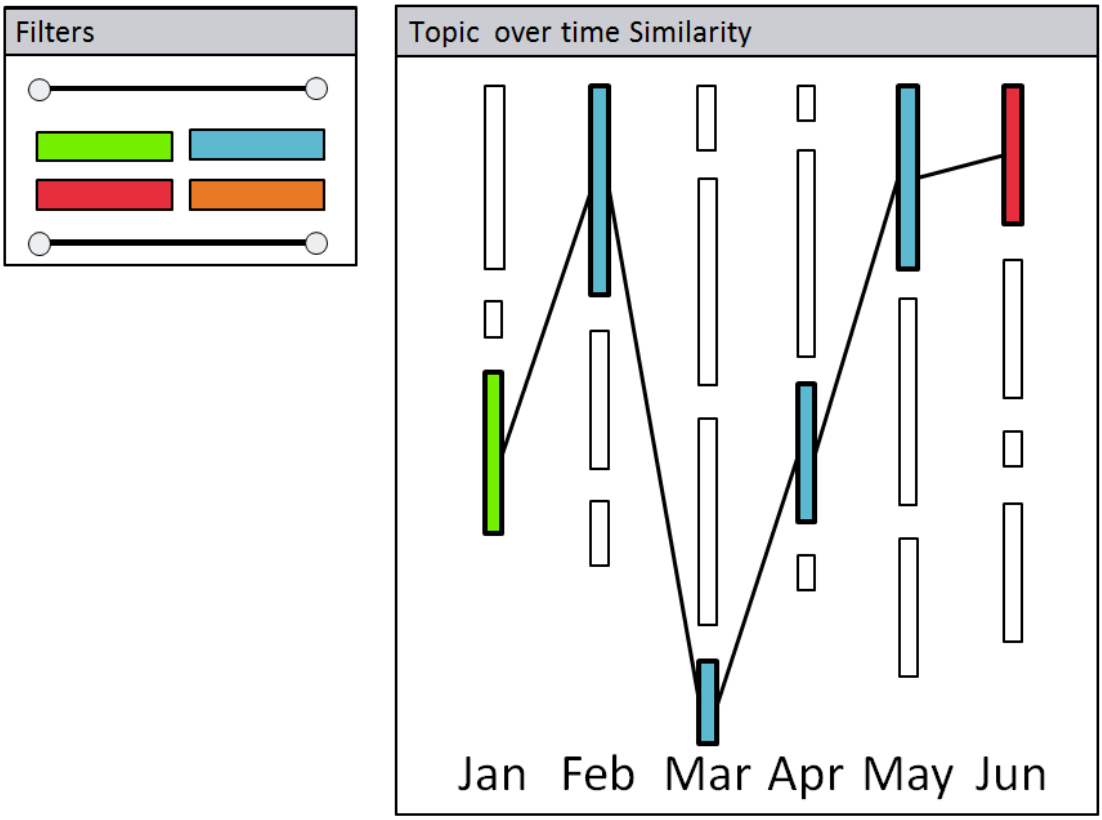


Figure 5.1: Topic Flow Model, adapted from [71].

5.2.1 Constructing Monthly Topics

To obtain multiple topics *per month*, as shown in Figure 5.2, we downloaded all ASRS 2020 reports publicly available via the ASRS Database Online [1] in tabular format.

Each ASRS report contains a timestamp of the format year-month (day or time information is not available). We used this timestamp to partition the entire collection of 2020 reports into months and then applied WarpLDA to each month *separately*. Therefore, in the example shown in the Figure which displays six months, WarpLDA would have been applied six times. ASRS has, at the time of the writing, eleven months worth of reports (January through November), and therefore WarpLDA was applied eleven times.

An important consideration here is the number of topics k, which in turn reflects the number of rectangles shown in Figure 5.1 per month. Since the algorithm is executed independently per month, any number of topics ranging from 2 and higher is a candidate number of k, and we chose $k = 10$ for all months. Our rationale is as follows:

In Chapter 3, we used the standard approach in topic modeling with perplexity to identify the

[1] `https://asrs.arc.nasa.gov/search/database.html`

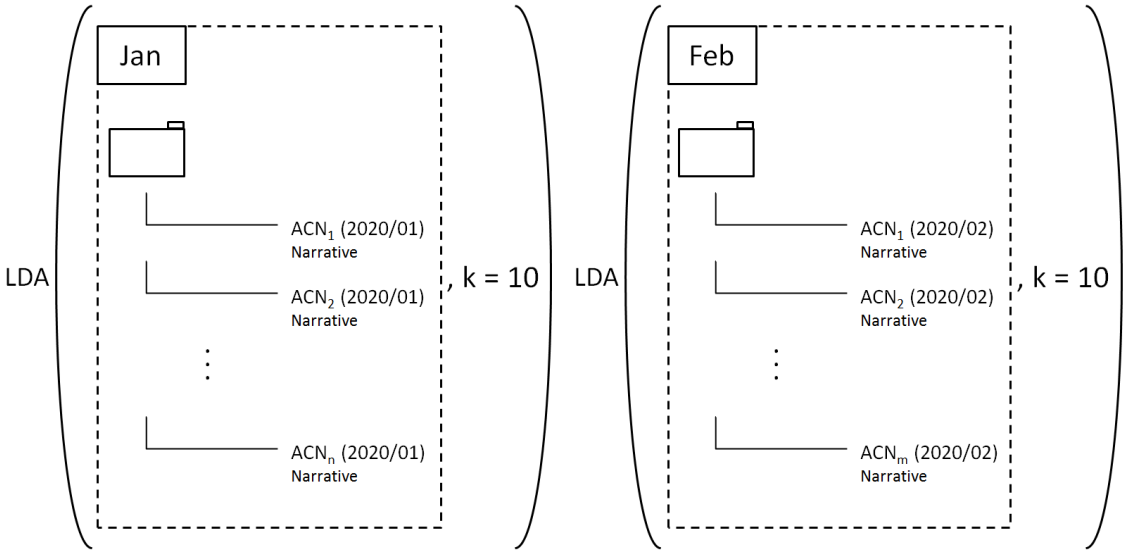


Figure 5.2: ASRS Report Partition for Monthly Topics.

ideal number of topics. However, our emphasis was on optimal grouping performance. While this is also true for topic flow, here we share the concerns of Chapter 4: the presentation of terms should be meaningful and useful to users. According to [13], both approaches are at odds: The authors even concluded that optimizing for perplexity groupings led to less meaningful sets of terms presented to users. A method to address the conflict, however, is not proposed. This is consistent with the closest work related to ours in [37] which applies topic modeling to identify trends in a NASA dataset. The authors state that the process of selecting the topic modeling parameters *lacks* definitive guidance.

We, therefore, decided to emphasize performance instead: The number of topics is reasonably sized to not be too small or too computationally expensive to be able to identify candidate edges among the resulting 11 months * 10 = 110 topics.

We chose this number for performance purposes only, as the construction of the edges, as we will discuss next, requires the Cartesian product of every pair of consecutive month's topics to assess their similarity and to construct the entire year's timeline requires the construction of 110 topics.

Finally, topic flow will compensate, to some degree, for any underestimation or overestimation of the "true" number of topics. If a given month has too many redundant topics, we would expect them to converge to a single topic in a subsequent month. Conversely, we expect them to split into different topics in the following month.

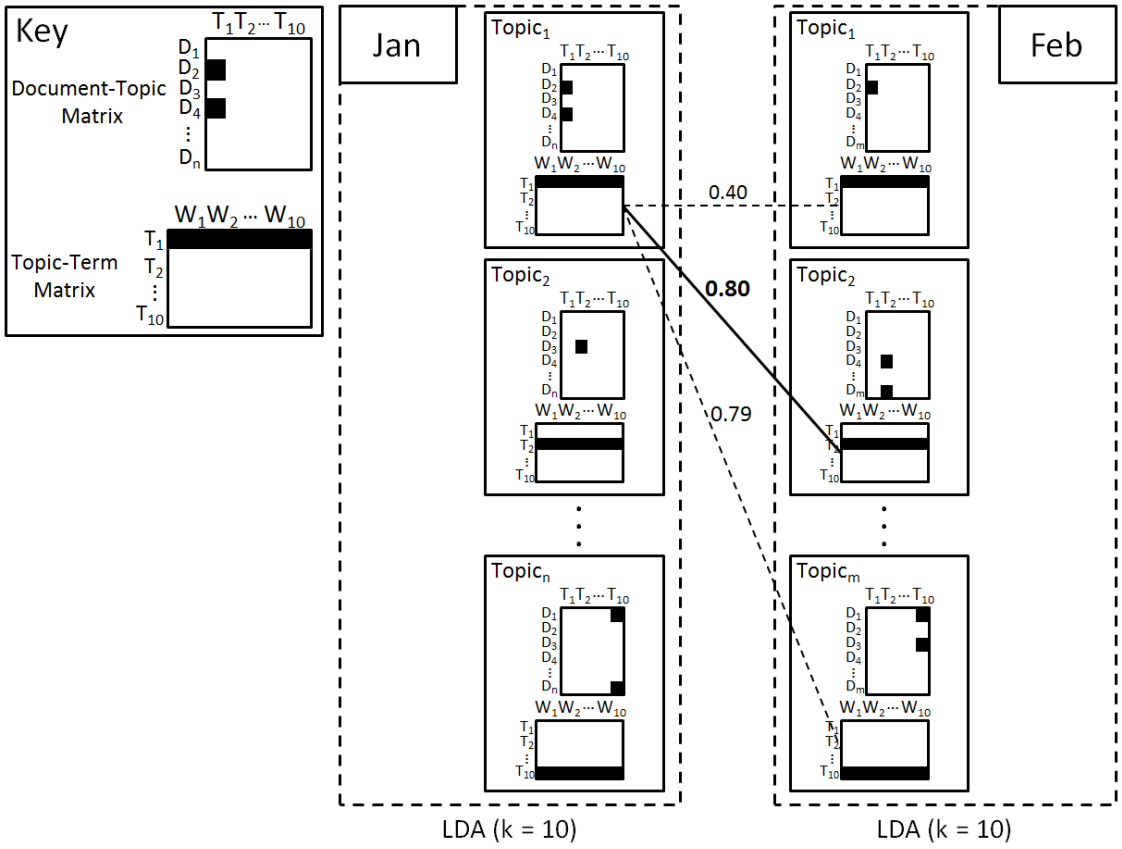


Figure 5.3: Connecting topics over time using topic term matrices.

5.2.2 Connecting Topics over Time

As we noted earlier, each rectangle in Figure 5.1 consists of a topic-term matrix and document-topic matrix. To create the *edges* shown, we use a similar approach to [71], which exclusively uses only the topic-term matrix.

Specifically, after we construct the monthly topics in Figure 5.2, all topic's (rectangles) topic-term matrices in month i are paired (via cartesian product) with month $i+1$ (e.g. Jan-Feb, Feb-Mar, etc.). For each pair we then compute the cosine similarity of the topic-term matrices, as defined in 5.1. The cartesian product is shown in Figure 5.3.

$$\cos(\theta) = \frac{z_i \cdot z_j}{\|z_i\|_i \, \|z_j\|_j} \tag{5.1}$$

Where in equation 5.1, z_i and z_j represent topic-term matrices, consistent with the notation introduced in previous chapters. The intuition here is that if the distribution of the set of words is similar between topics, then they are considered similar.

For example, for topic 1 in January, its similarity to all ten topics in February is calculated. What remains then is to decide which candidate edges to keep, resulting in a diagram similar to

the one shown in Figure 5.1.

In [71], the authors use a global threshold for all topics that a user can select via a slider (top left of the Figure 5.1) to adjust what threshold should be used between all topics to preserve the lines. We see some limitations with this approach: The decision of threshold is entirely up to the user by trial and error, and it is not necessarily clear how the user would have achieved the correct parameter. Moreover, we find it unlikely that a single global threshold can effectively reflect all pairs of topics and believe a different approach to establish the line filtering criteria could more accurately reflect the trends. Specifically, we chose to use maximum likelihood instead and include alongside each edge the associated similarity weight (which ranges from 0 to 100%).

Consider again the previous example where January's topic 1 has ten candidate edges to the ten topics of February. Suppose also among the ten February topics, Topic 7 has the highest similarity to January's topic 1. Then, via maximum likelihood, we claim topic 1 is connected to topic 7, and the remaining candidate topic edges are deleted. A major threat to validity, in this case, is that topics may end up connected while having very low similarity. However, since we know the similarity weights, while inspecting the terms of January's topic 1, the terms of February's topic 7, and the similarity weights of other topic pairs of the two months, we may conclude that the topics are, in reality, disjoint. If this is the case, then all the remaining nine candidate edges would also be disjoint since the chosen edge had the highest similarity weight. Therefore, we find this an easier approach to derive the edges from the candidate edges in topic flow.

Note this does not necessarily put the burden on the user. The user is primarily looking at the set of words for sense-making. If there is a disconnect in meaning, the user can then "drill down" to make a more informed guess. This is different from the global threshold approach from [71], where, from the very start, users have to identify a global threshold for all of the datasets without any data evidence.

5.2.3 Paths and Timelines

To facilitate discussion in the following results section, we define the set of *single* topics (rectangles) shown in Figure 5.1 connected across all months as a *path*, in the sense sense of graph theory. A timeline is defined as a set of paths that contain at least one topic in common.

5.2.4 Types of Topics within Timelines

In *topic flow*'s original definition [71], the lines that connect the rectangles represent four types of relations: emerging, continuing, ending and standalone. In Figure 5.1, the green rectangle represents *emerging*, the blue rectangle *continuing*, and the red rectangle *ending*. A standalone orange rectangle would resemble the colorless rectangles shown in the Figure, although we left them colorless to emphasize the other colored three, which exemplify a *a timeline*. Specifically, emerging topics are those that have no earlier month's topics connecting to it. Conversely, ending

topics are those that have no later month's topics connecting to it. Continuing topics are those who have both prior and following months edges, and standalone those who have neither.

In the maximum likelihood criteria we adopted, topic flow will generate all but standalone topics. However, if users deem the similarity weight to be low for any pair of topics and/or the set of terms of the chosen month pair disjoint, then in this case the topic may become standalone.

5.2.5 Deterministic Mapping Criteria

To answer RQs 2 and 3, we must consider documents as belonging to a given topic, or not, and therefore to a given path and timeline. Similar to Chapter 3, we must provide a deterministic mapping to evaluate whether the topic assignment is accurate using our evaluation dataset. In both Chapter 3 studies, we used a maximum likelihood assumption, i.e., we assume the highest likelihood reflects the more appropriate, and importantly, single mapping.

5.3 Results

The result of performing topic flow in the entire 2020 ASRS Database is split into two Figures due to space constraints, and shown in Figures 5.4 and 5.5. We now answer our three research questions.

RQ1: Are there timelines whose terms clearly suggest the prevalence of COVID-19 reports on its topic's top terms?

We manually identified, using *only the terms of each topic*, a total of 19 COVID-19 topics out of the 110 topics, distributed across 6 paths (Path IDs 12, 4, 2, 6 1 and 32) and 4 timelines (1, 5, 6, 32). The list of terms associated to each of the 19 identified topics is shown in Tables 5.1 and 5.2. The identified COVID-19 topics are also highlighted in a red rectangle in Figures 5.4 and 5.5.

Therefore, the answer to RQ1 is *Yes*, it is possible to identify timelines in which terms clearly suggest the prevalence of COVID-19 reports. We note the first COVID-19 related topic through the set of terms dates from March, which is the first occurrence of COVID-19 reports in the ASRS COVID-19 Report Set (Timeline 1, Path ID 12 in Figures 5.4,5.5. We also observed from the Figure that at least one topic in every month of the timeline from March and onward contained terms associated to COVID-19, but in different timelines. We would expect therefore that they convey different meanings, or otherwise the topics should have been part of the same paths.

In *Path ID 12*, we see terms suggestive of the crew preparedness through company training to assist passengers due to the virus. *Path ID 4*, in turn emphasizes the use of face masks and row seats for passenger safety. The focus of *Path ID 6* is on the effect of COVID-19 on work days of the crew, and airport related flying safety issues. Common to *Path IDs 4, 6, and 2* are the topics in

Timeline ID	**Path ID**	**Topic ID**	**Month**	**Top 10 Terms**
5	2	2	Jan	traffic tower approach airport clearance pilot final departure turn frequency
5	2	4	Feb	traffic tower final cleared told left pattern pilot downwind made
5	2	4	Mar	tower turn airport miles fuel left landing approach flying frequency
5	2	6	Apr	tower class clearance traffic airspace vfr departure route called asked
5	2	5	May	tower taxiway taxi controller control cleared contact airspace plan short
5	2	3	Jun	tower controller clearance departure asked told ground called cleared frequency
5	2	3	Jul	tower taxi clearance ground frequency ramp taxiway turn controller cleared
5	2	3	Aug	departure clearance crew due pilots noticed area correct change takeoff
5	**2**	**3**	**Sep**	**due close pilots flying change company covid times missed atc**
5	**2**	**2**	**Oct**	**day due company covid hours captain event trip message flying**
5	**2**	**6**	**Nov**	**company training crew covid month due informed situation return pilots**

Table 5.1: Set of top 10 related words for the topics of Path ID 2, which contain COVID-19 related terms. These topics can be seen in the Figure. This displays all the topics in path ID 2, both COVID-19 related and non COVID-19 related words. Rows in bold indicate adjacent topics in which terms were deemed COVID-19 related.

Timeline ID	Path ID	Topic ID	Month	Top 10 Terms
1	**12**	**2**	**Mar**	**company safety supervisor told covid passengers training virus crew hand**
1	**12**	**9**	**Apr**	**crew covid company information due informed action day safety fact**
1	12	8	May	maintenance crew fuel made cockpit ramp position make situation dangerous
5	**4**	**1**	**Apr**	**mask passengers passenger safety masks covid fa rows people seat**
5	**4**	**9**	**May**	**passenger mask masks safety face crews service told wearing seat**
5	**4**	**9**	**Jun**	**taxi taxiway short line passenger hold ramp takeoff make mask**
5	4	3	Jul	tower taxi clearance ground frequency ramp taxiway turn controller cleared
5	**6**	**7**	**May**	**flying due part noticed day failed covid panel looked flown**
5	**6**	**6**	**Jun**	**day crew work days hours part event covid working company**
5	**6**	**2**	**Jul**	**airport working covid day area flying issue safety due information**
5	**6**	**6**	**Aug**	**day company crew told covid part number ramp work call**
5	**6**	**8**	**Sep**	**safety air day work made hours operations covid service make**
6	**1**	**9**	**Jul**	**mask passenger passengers asked fa seat told put gate policy**
6	**1**	**7**	**Aug**	**mask passenger passengers fa seat asked policy told face captain**
6	**1**	**5**	**Sep**	**mask passenger passengers asked fa told put policy wear attendant**
6	**1**	**1**	**Oct**	**mask passenger passengers attendant wear seat fa asked wearing boarding**
6	**1**	**7**	**Nov**	**passenger passengers policy mask fa seat attendant service distancing social**
10	**32**	**8**	**Nov**	**tower work flights shift cleaning windshear told controllers safety covid**

Table 5.2: Set of top 10 related words for the topics of Path IDs 1, 4, 6, 12, and 32, which contain COVID-19 related terms. These topics can be seen in the Figure surrounded by red rectangles. Rows in bold indicate adjacent topics in which terms were deemed COVID-19 related.

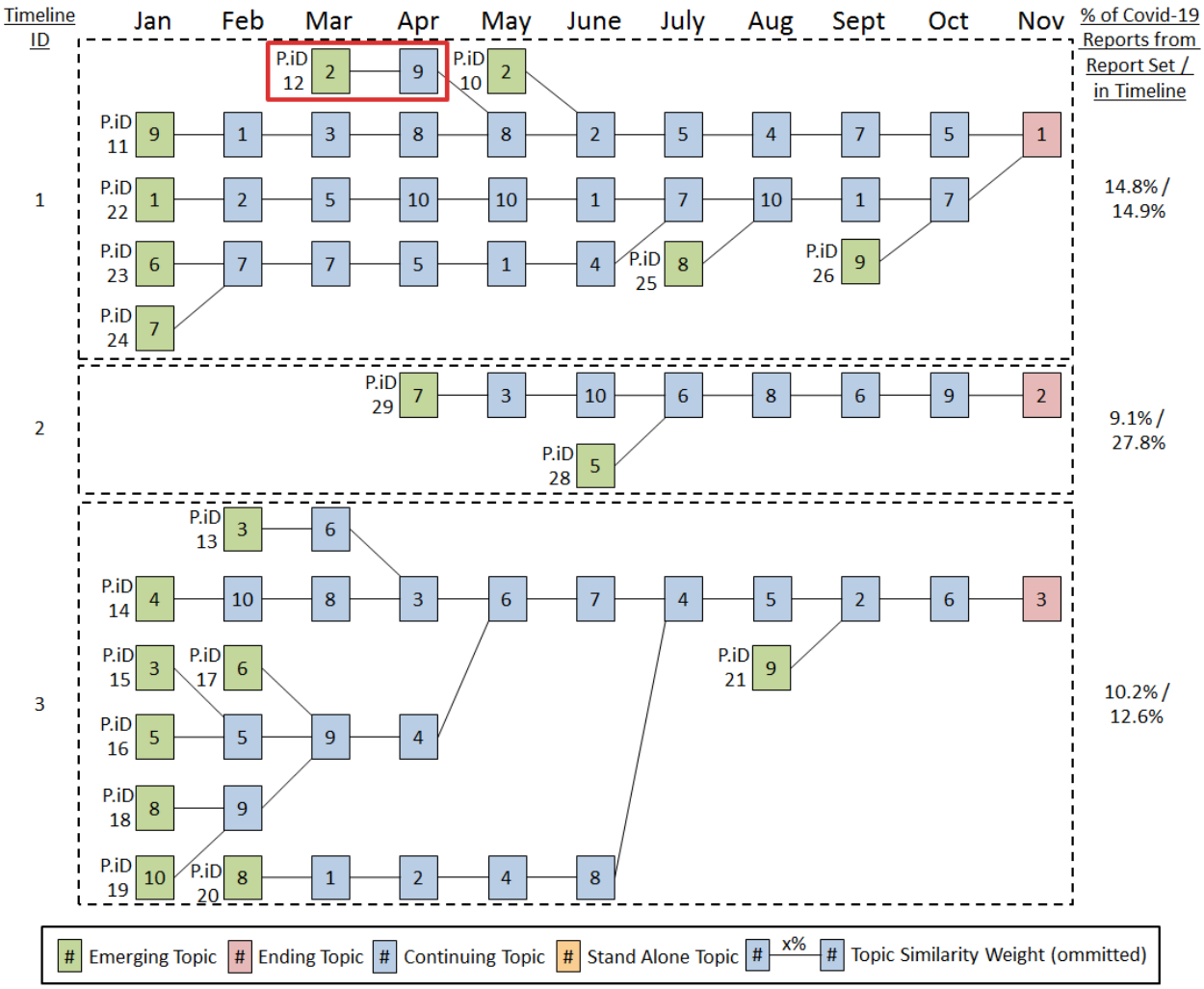


Figure 5.4: ASRS 2020 Topic Flow.

October and November, which is associated to company training for COVID-19 situations, which is a reasonable common ground for all three path ids to converge.

Path IDs 1 and 32 convey the use of masks and cleaning and shift procedures for COVID safety for controllers. Both *Path ID 1* and *Path ID 4* are associated with mask wearing, *Path ID 4* seems related to boarding and social distancing due to policy (terms: gate, policy, boarding, policy, social, distancing) whereas *Path ID 1* mask wearing concerns appear more associated to the positioning of passengers on the plane (terms: rows, seat, passengers, service, takeoff).

While the full interpretation of the terms is open for debate, we believe it is reasonable from the presented list of terms that they are suggestive of different COVID-19 themes in different path ids, and suggestive of similar themes, *within* path ids, as we would hope they would be.

We emphasize the COVID-19 related topics appear consecutively, instead of scattered across

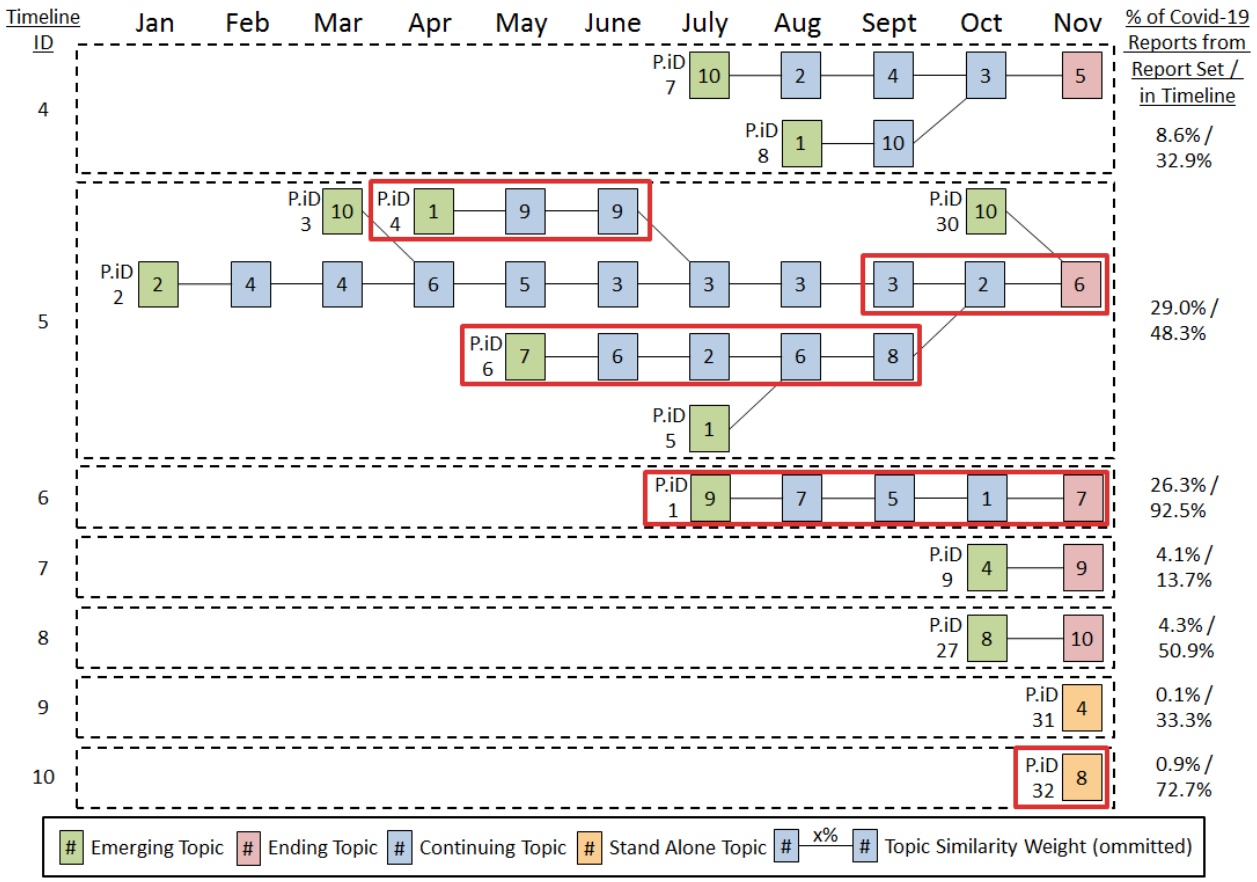


Figure 5.5: ASRS 2020 Topic Flow (Continued).

Figures 5.4,5.5, consistent with the intuition of emerging timelines, and with the effect of cosine similarity on topic-term matrices of consecutive months.

> RQ2: If there exist timelines where the top terms suggest prevalence of COVID-19 reports, what proportion of the Report Set's COVID-19 reports do they account for?

As we argued in Chapter 4, other studies limit the evaluation of topic modeling to the author's set of terms only. In contrast, similar to Chapter 3, we leverage report sets to empirically evaluate the results of topic modeling.

In Figure 5.4,5.5, if a user were to have inspected the *entire timeline* where at least one COVID-19 topic occurs (based only on the use of terms we chose), then they would have encountered approximately $14.8\% + 29.0\% + 26.3\% + 0.9\% = 71\%$ of the COVID-19 reports from the COVID-19 report set. In doing so they would have disregarded the timelines with the least number of the COVID-19 report set's reports (i.e. timelines 2, 3, 4, 7, 8 and 9, with the exception of timeline 10).

However, this is not the only approach a user may choose to navigate the reports. Instead,

the search can be constrained to only the paths where a COVID-19 topic occurs, or the topics themselves in the interest of time. The ability to choose a trade-off between precision and recall here, accounting for human effort, rather than only keyword searching, we believe, merits the benefit of our proposed method.

RQ3: Are there timelines which consist exclusively of COVID-19 reports from the Report Set?

In Figure 5.4,5.5, we can see a wide variety of saturation levels of COVID-19 reports from the report set, ranging from timelines 6 (92.5%) to timeline 3 (12.6%). However, there are no exclusive COVID-19 timelines, and hence the answer to RQ3 is no. However, contrary to Chapter 3 where groupings are thematically related, a lower saturation of COVID-19 reports here suggests non-related COVID-19 themes which may overlap in problem, equipment or solution to COVID-19.

For instance, in Topic 5 we can see path ids 4, 2, and 6 have COVID-19 related topics (per their display of terms). However, July's topic 3, and August's topic 3 do not. The overlap, however, occurs on issues during clearance and takeoff (terms: tower, clearance, departure), which is also the overarching topic for path id 2 (terms: tower, appoach, vfr, controller taxiway).

5.4 Conclusion and Future Work

In this chapter, we combined methods from previous chapters and related literature to define topic flow. We provided both qualitative and quantitative evaluation of the viability of topic flow for capturing emerging safety threats using a timely emerging theme: COVID-19. Contrary to related literature, we considered the results of topic flow in the context of a real-world system and the merits of its usage in operation.

Our construction of the topic flow method allows for the identification of emerging safety threats, as it does not attempt to fit the entire year of topics a priori. Rather, a timeline can be defined as soon as any time granularity (e.g., month, week) is available, and also allows for the existence of topics that didn't exist up to a given point to emerge, therefore fully realizing the goal of this dissertation.

Future work will focus on relaxing the restriction of hard assignments between document topics, therefore allowing multi-topic assignment and non-consecutive months edge creation, therefore identifying the reoccurrence of previously known safety threats.

We also intend to experiment with applying the method to other domains, including software vulnerability taxonomies such as CAPEC and CWE.

CHAPTER 6
CONCLUSION

In this proposal, I presented a holistic framework, PERCEIVE, to support organizations that collect reports by enabling over time analysis. The work motivation is that currently, most of the analysis is done manually by experts and on an individual record basis. As I exemplified in the introduction Chapter, not only can this incur damage, but it can also prove fatal.

To ensure the work is novel, I used what is considered state of the art in software engineering research literature review, extensively used and borrowed from the medical field, by performing a systematic mapping study. Systematic mapping studies provide a *reproducible* protocol by identifying, executing, and analyzing a priori research questions for a field of study to ensure coverage of the subject of interest.

I exemplified the methodology applied to vulnerabilities and safety threats by proposing two models: commit flow and topic flow, to account for emerging threats. In covering two domains, software vulnerabilities and safety threats, I expanded the framework to account for source code and free text. To the best of my knowledge, no prior work has attempted to account for this diverse type of data or scope, as presented in the mapping study Chapter.

For the content analysis chapter, I have also provided in the appendix the associated primary source references where each claim and interpretation made in the chapter can be assessed. In the current state of software engineering practice, supplemental material is uncommon, and source code a needle in the haystack.

Finally, while I chose to exemplify PERCEIVE in two domains, the entire dissertation could have been performed using vulnerabilities (ASRS does not have source code data associated with it). Specifically, I defer to future work the chapters that apply the metadata model in the interest of scope. Early work not included in this dissertation devoted considerable time to establish a mapping between the various metadata sources of vulnerabilities and linkages. I include this mapping in Appendix B with the hopes it may be useful to others performing related work. For Topic Flow, both CWE[1] and CAPEC[2] provide for its taxonomies a detailed comparison of what is changed over every new version, with references to vulnerabilities. This can be used to assess topic flow in a similar manner as I did with ASRS in the topic flow Chapter.

[1] `https://cwe.mitre.org/data/reports.html`
[2] `https://capec.mitre.org/data/reports.html`

APPENDIX A
CONTENT ANALYSIS SUPPLEMENTAL MATERIAL

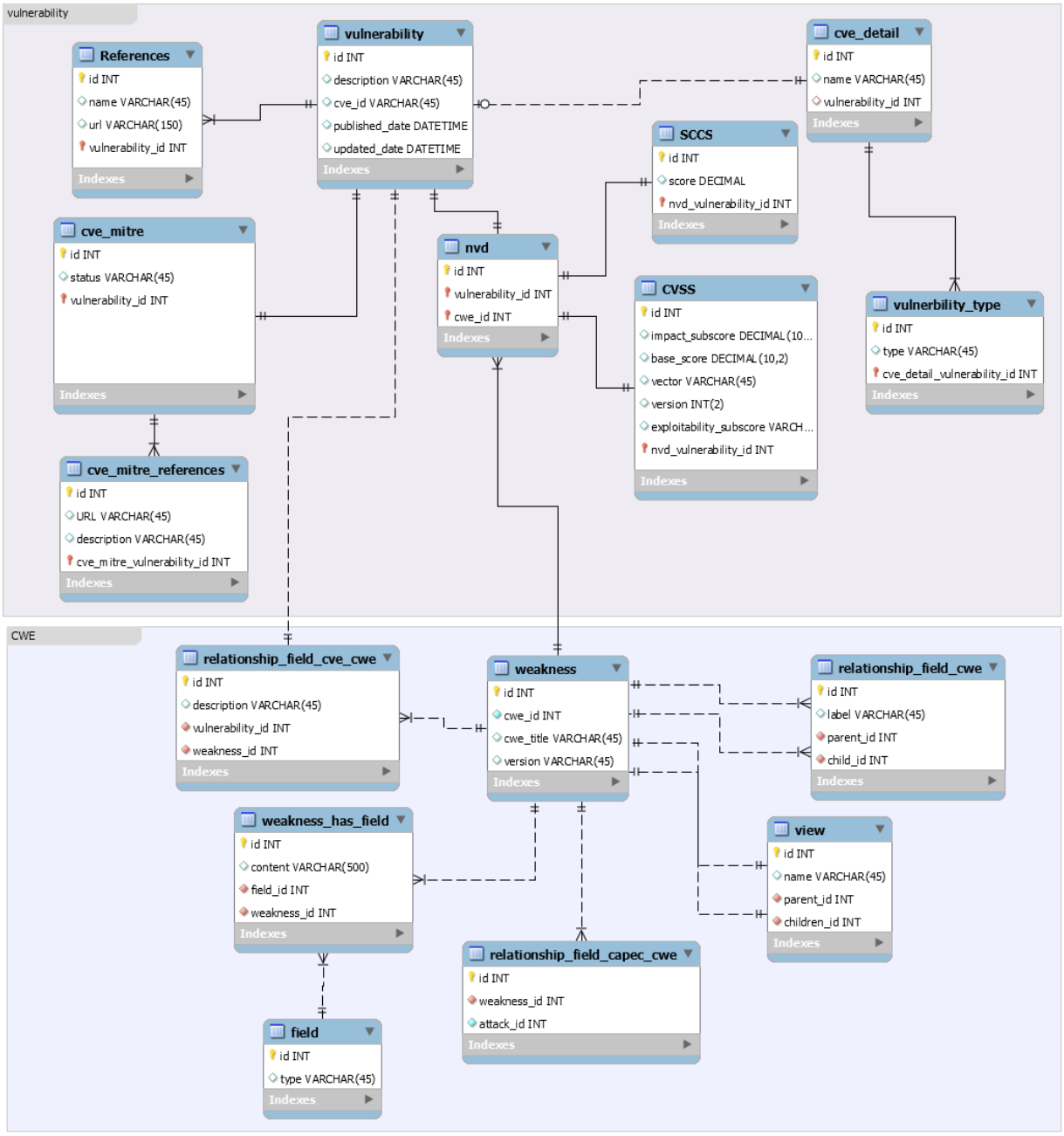
This Appendix provides supplemental material to the references used in Chapter 2.

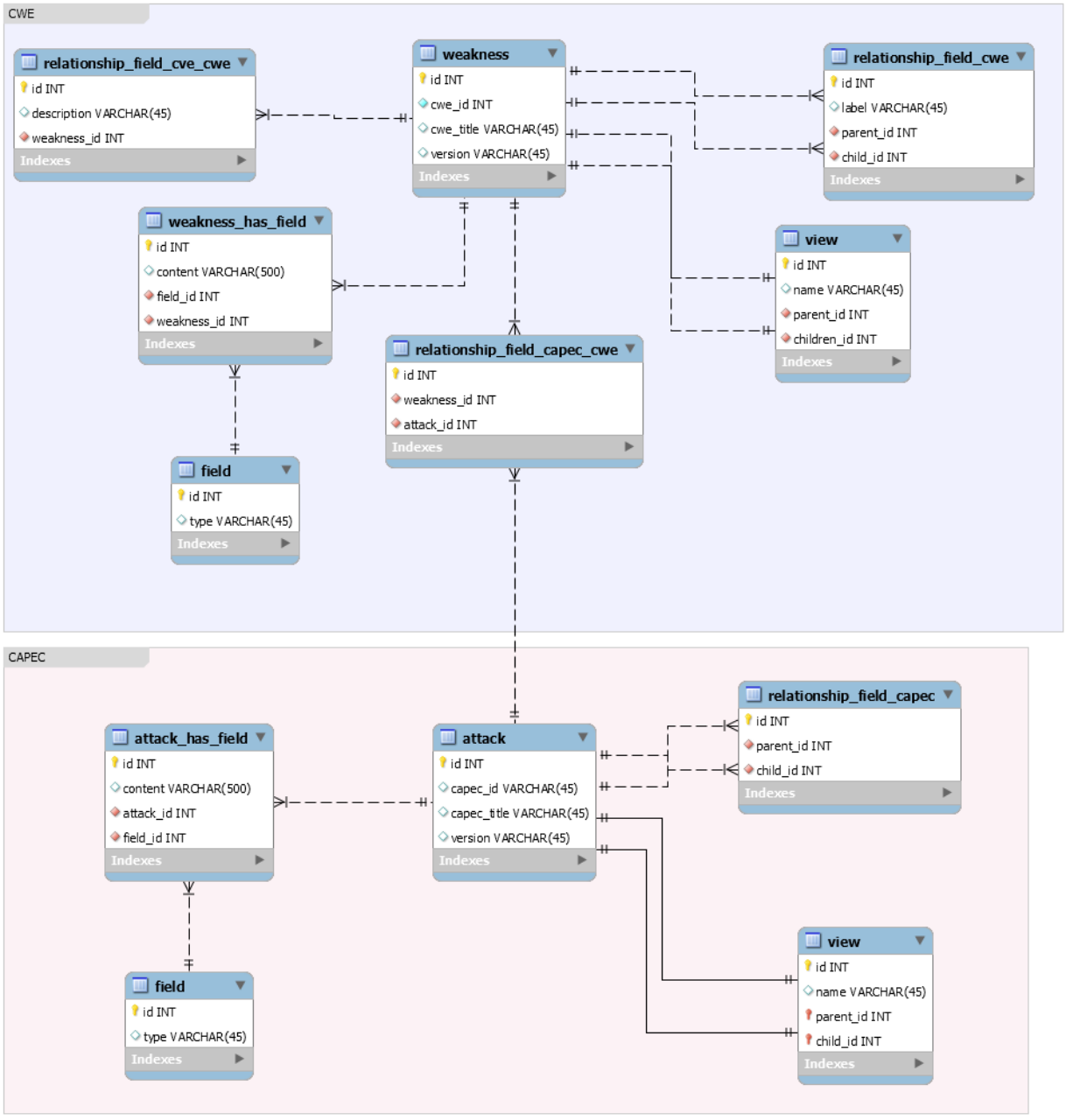
Table A.1: References for Content Analysis in OpenSSL.

ID	Reference
M1	https://marc.info/?l=openssl-dev\&m=100429666608071\&w=2
M2	https://marc.info/?l=openssl-dev\&m=102181324915607\&w=2
M3	https://marc.info/?l=openssl-dev\&m=147638421307039\&w=2
M4	https://marc.info/?l=openssl-dev\&m=147638473207215\&w=2
M5	https://marc.info/?l=openssl-dev\&m=97741948606308\&w=2
M6	https://marc.info/?l=openssl-dev\&m=96457413327658\&w=2
M7	https://marc.info/?l=openssl-dev\&m=96498092726092\&w=2
M8	https://marc.info/?q=about#Robots
M9	https://www.openssl.org/community/mailinglists.html
M10	https://web.archive.org/web/20060220165750/http://www.openssl.org/support/
M11	https://mta.openssl.org/pipermail/openssl-project/2018-November/001171.html
B1	https://www.openssl.org/blog/blog/2018/05/16/security-policy/
B2	https://www.openssl.org/blog/blog/2017/02/13/bylaws/
B3	https://www.openssl.org/blog/blog/2020/05/12/security-prenotifications/
B4	https://www.openssl.org/blog/blog/2014/12/23/the-new-release-strategy/
B5	https://www.openssl.org/policies/roadmap.html
B6	https://www.openssl.org/blog/blog/2014/12/19/hello/
B7	https://www.openssl.org/blog/blog/2016/10/12/f2f-rt-github/
B8	https://www.openssl.org/blog/blog/2014/12/28/website-redesign/
B9	https://www.openssl.org/blog/blog/2017/06/13/committers/
B10	https://www.openssl.org/blog/blog/2015/08/01/cla/
B11	https://www.openssl.org/blog/blog/2018/03/01/last-license/
B12	https://www.openssl.org/blog/blog/2017/06/17/code-removal/
B13	https://www.openssl.org/blog/blog/2015/01/05/source-code-reformat/
B14	https://www.openssl.org/blog/blog/2015/07/28/code-cleanup/
B15	https://www.openssl.org/blog/blog/2018/11/28/version/
B16	https://www.openssl.org/blog/blog/2018/12/20/20years/
B17	https://undeadly.org/cgi?action=article;sid=20140423045847
B18	https://www.openbsd.org/papers/bsdcan14-libressl/mgp00001.html
B19	https://opensslrampage.org/page/49
B20	https://www.openbsd.org/papers/bsdcan14-libressl/mgp00026.html
B21	https://www.openbsd.org/papers/bsdcan14-libressl/mgp00020.html
B22	https://www.theguardian.com/technology/2014/apr/11/heartbleed-developer-error-regrets-oversight
B23	https://www.openssl.org/news/vulnerabilities.html
C1	https://github.com/openssl/openssl/commit/bd6941cfaa31ee8a3f8661cb98227a5cbcc0f9f3
C2	https://github.com/openssl/openssl/commit/731f431497f463f3a2a97236fe0187b11c44aead

APPENDIX B
MAPPING OF CVE, CWE AND CAPEC

The following mapping provides the relevant fields and relationships between vulnerabilities (CVEs), weaknesses (CWEs), and attack patterns (CAPEC), including different instantiations of CVEs, the National Vulnerability Database (NVD) and CVE Details. This mapping was the result of extensive work from our group in examining available data in these resources, and to the best of our knowledge was not available at the time of this proposal. It is made available here with the hopes it may be useful in extending the metadata network and topic flow in vulnerabilities for related research in future work. This model and the contributing authors can be found in `https://github.com/sailuh/perceive/tree/master/Database`.


vulnerability
References
id INT
name VARCHAR(45)
url VARCHAR(150)
vulnerability_id INT
Indexes
vulnerability
id INT
description VARCHAR(45)
cve_id VARCHAR(45)
published_date DATETIME
updated_date DATETIME
Indexes
cve_detail
id INT
name VARCHAR(45)
vulnerability_id INT
Indexes
SCCS
id INT
score DECIMAL
nvd_vulnerability_id INT
Indexes
cve_mitre
id INT
status VARCHAR(45)
vulnerability_id INT
Indexes
nvd
id INT
vulnerability_id INT
cwe_id INT
Indexes
CVSS
id INT
impact_subscore DECIMAL(10...
base_score DECIMAL(10,2)
vector VARCHAR(45)
version INT(2)
exploitability_subscore VARCH...
nvd_vulnerability_id INT
Indexes
vulnerbility_type
id INT
type VARCHAR(45)
cve_detail_vulnerability_id INT
Indexes
cve_mitre_references
id INT
URL VARCHAR(45)
description VARCHAR(45)
cve_mitre_vulnerability_id INT
Indexes
CWE
relationship_field_cve_cwe
id INT
description VARCHAR(45)
vulnerability_id INT
weakness_id INT
Indexes
weakness
id INT
cwe_id INT
cwe_title VARCHAR(45)
version VARCHAR(45)
Indexes
relationship_field_cwe
id INT
label VARCHAR(45)
parent_id INT
child_id INT
Indexes
weakness_has_field
id INT
content VARCHAR(500)
field_id INT
weakness_id INT
Indexes
view
id INT
name VARCHAR(45)
parent_id INT
children_id INT
Indexes
relationship_field_capec_cwe
id INT
weakness_id INT
attack_id INT
Indexes
field
id INT
type VARCHAR(45)
Indexes


CWE
relationship_field_cve_cwe
id INT
description VARCHAR(45)
weakness_id INT
Indexes
weakness
id INT
cwe_id INT
cwe_title VARCHAR(45)
version VARCHAR(45)
Indexes
relationship_field_cwe
id INT
label VARCHAR(45)
parent_id INT
child_id INT
Indexes
weakness_has_field
id INT
content VARCHAR(500)
field_id INT
weakness_id INT
Indexes
view
id INT
name VARCHAR(45)
parent_id INT
children_id INT
Indexes
relationship_field_capec_cwe
id INT
weakness_id INT
attack_id INT
Indexes
field
id INT
type VARCHAR(45)
Indexes
CAPEC
relationship_field_capec
id INT
parent_id INT
child_id INT
Indexes
attack_has_field
id INT
content VARCHAR(500)
attack_id INT
field_id INT
Indexes
attack
id INT
capec_id VARCHAR(45)
capec_title VARCHAR(45)
version VARCHAR(45)
Indexes
view
id INT
name VARCHAR(45)
parent_id INT
child_id INT
Indexes
field
id INT
type VARCHAR(45)
Indexes

APPENDIX C
SURVEY AND PROTOTYPE

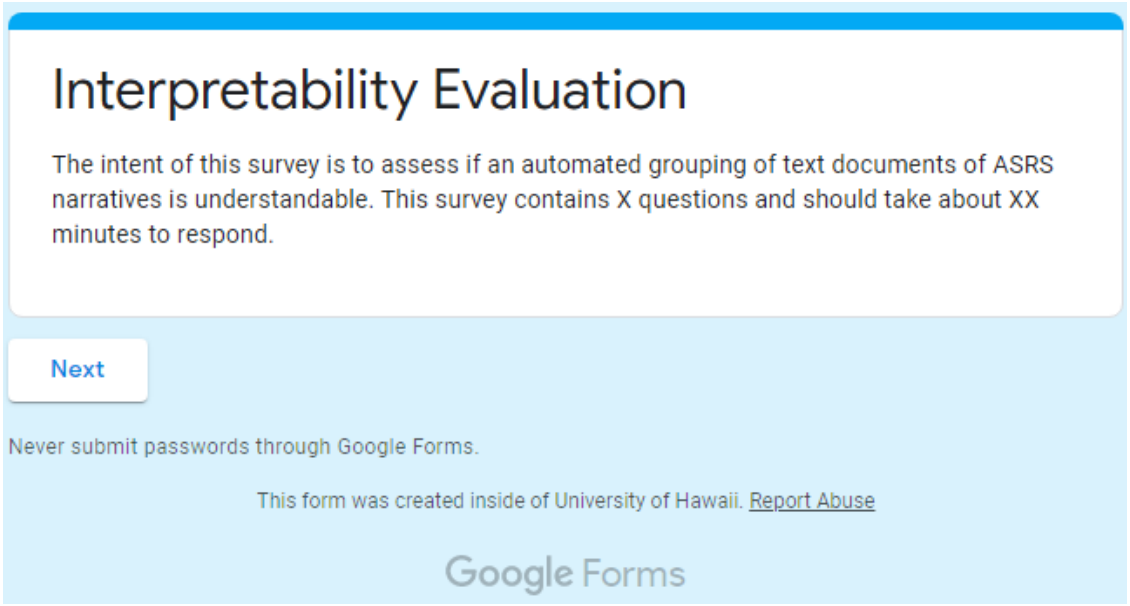


Figure C.1: Survey v1 p1.

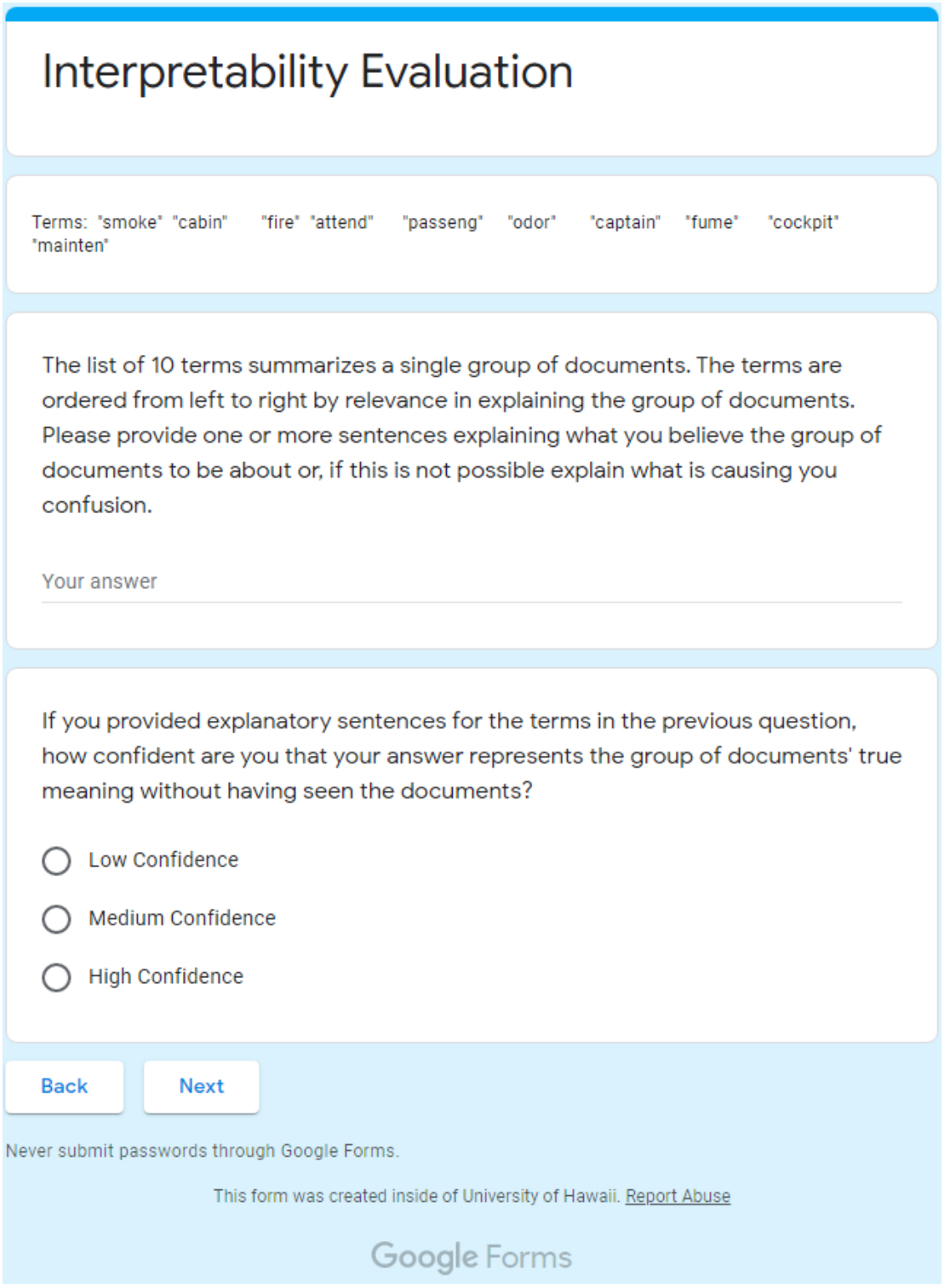


Figure C.2: Survey v1 p2.

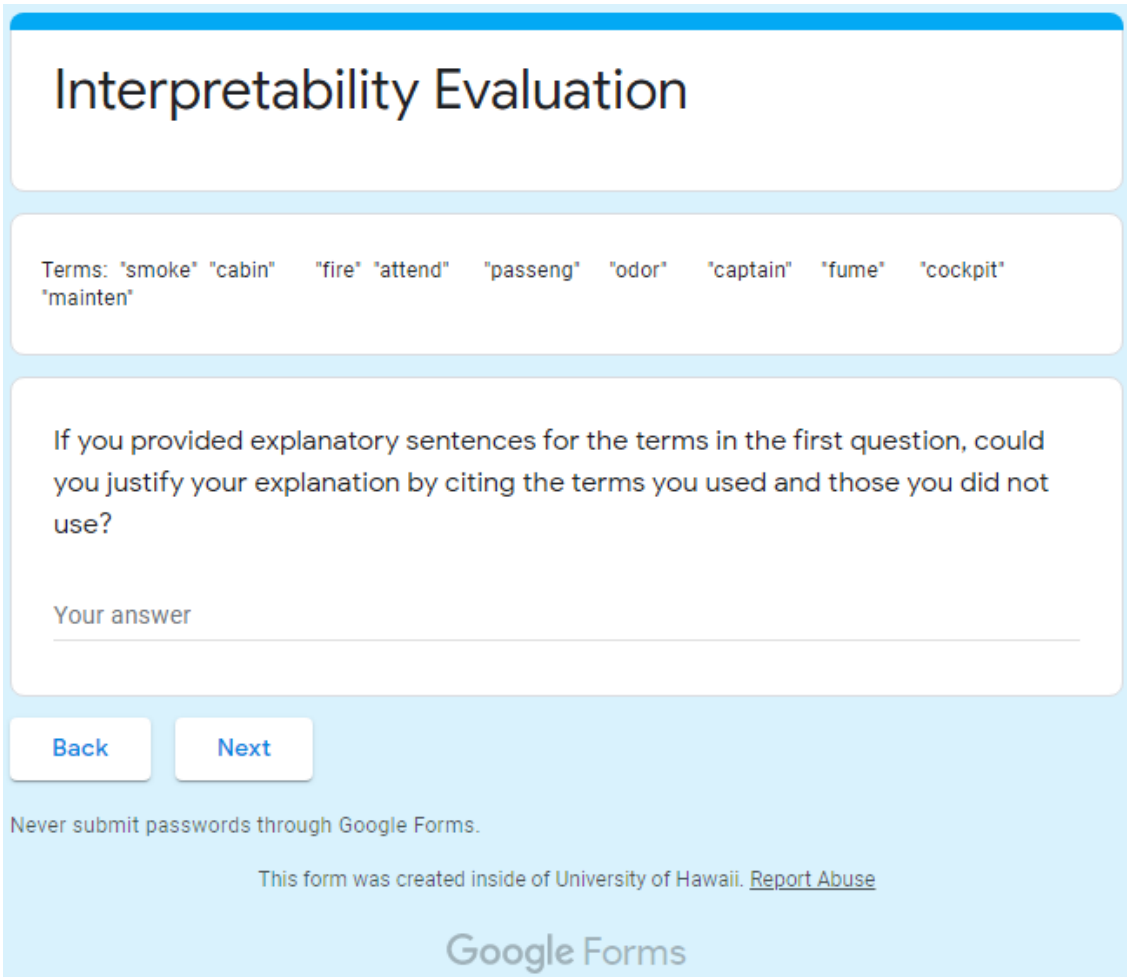


Figure C.3: Survey v1 p3.

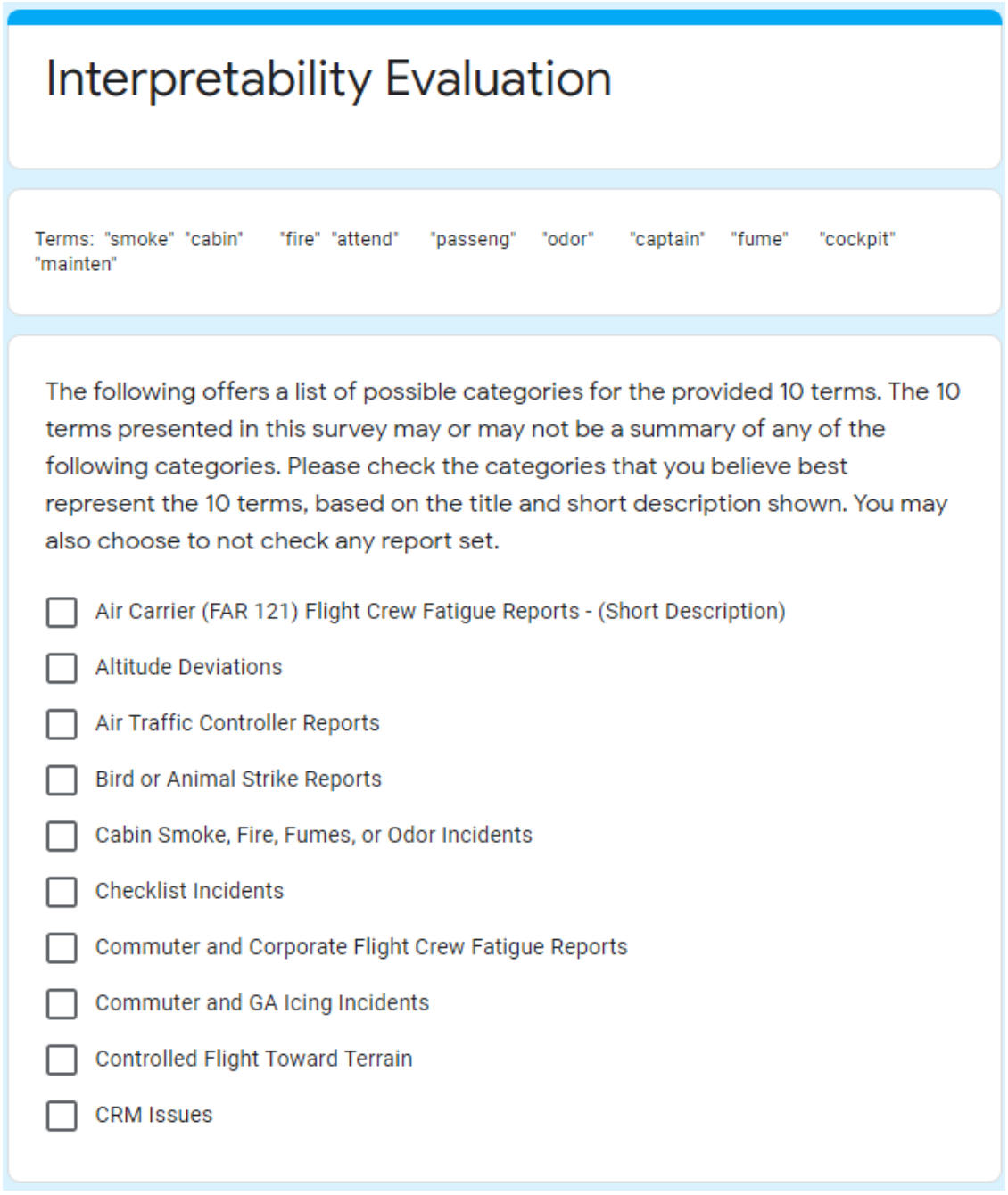


Figure C.4: Survey v1 p4 1.

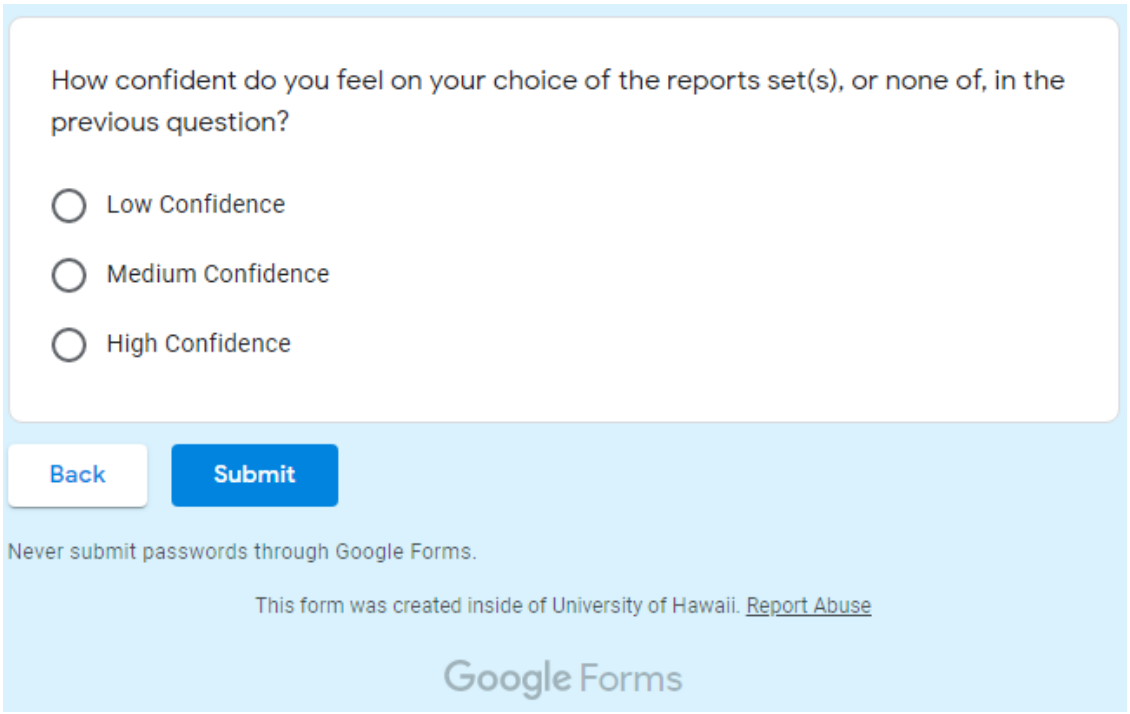
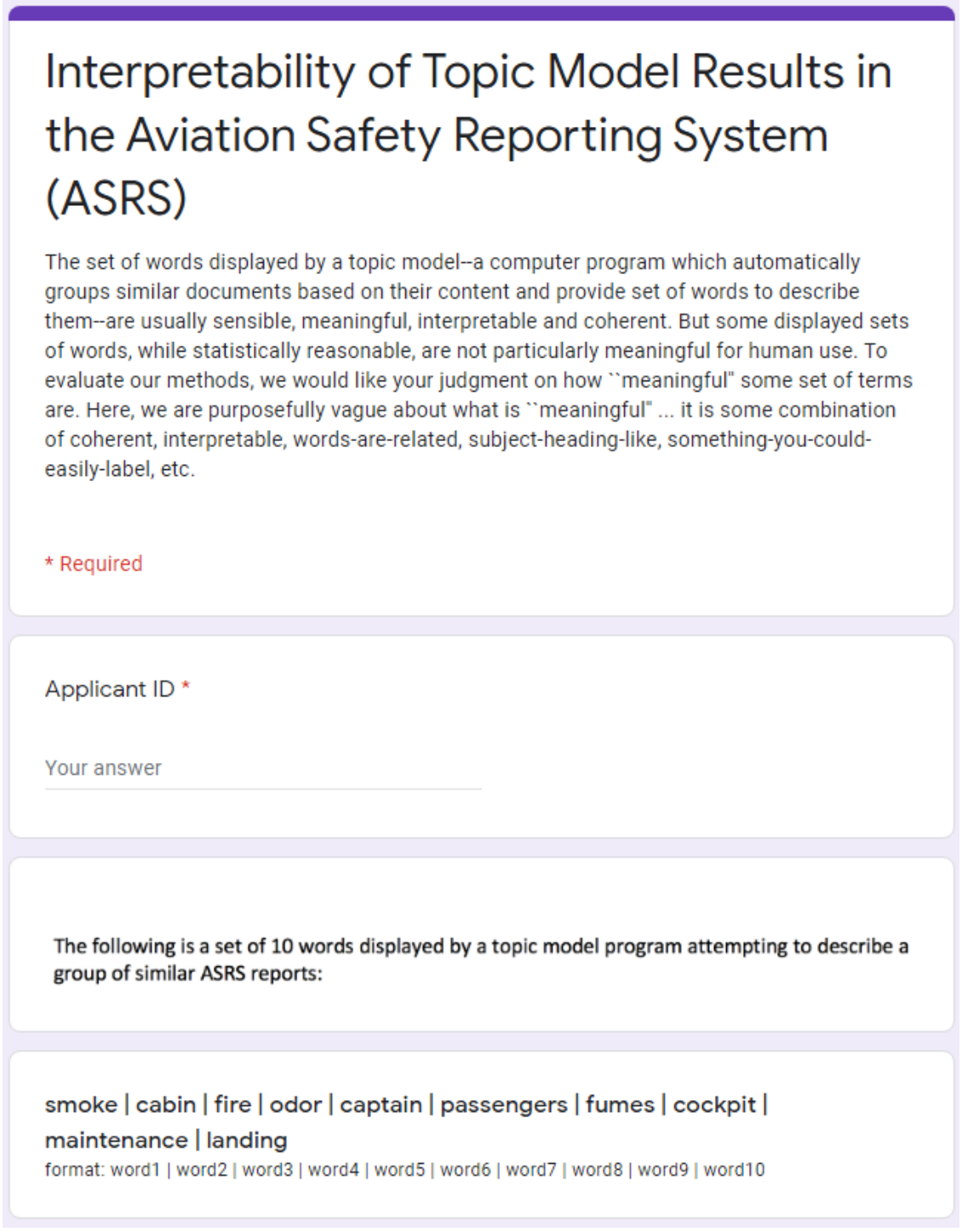


Figure C.5: Survey v1 p4 2.



Interpretability of Topic Model Results in the Aviation Safety Reporting System (ASRS)

The set of words displayed by a topic model--a computer program which automatically groups similar documents based on their content and provide set of words to describe them--are usually sensible, meaningful, interpretable and coherent. But some displayed sets of words, while statistically reasonable, are not particularly meaningful for human use. To evaluate our methods, we would like your judgment on how ``meaningful" some set of terms are. Here, we are purposefully vague about what is ``meaningful" ... it is some combination of coherent, interpretable, words-are-related, subject-heading-like, something-you-could-easily-label, etc.

* Required

Applicant ID *

Your answer

The following is a set of 10 words displayed by a topic model program attempting to describe a group of similar ASRS reports:

smoke | cabin | fire | odor | captain | passengers | fumes | cockpit | maintenance | landing

format: word1 | word2 | word3 | word4 | word5 | word6 | word7 | word8 | word9 | word10

Figure C.6: Survey v2 p1.

smoke | cabin | fire | odor | captain | passengers | fumes | cockpit | maintenance | landing

format: word1 | word2 | word3 | word4 | word5 | word6 | word7 | word8 | word9 | word10

1. Please **identify**, if possible, in the following answer one or more **topic labels** that you believe the set of words are about. For each topic label, you may: a) use one word from the list of provided words, b) infer one word that is not in this list but which you believe better describes the topic; c) use a combination of a few words (i.e., a phrase) that describes the topic; d) explain the meaning of the topic in a sentence; e) some other method.

2. Please **organize** the topic labels you have chosen in part 1 above in the following manner:

1) place one topic label per line
2) rank these topic labels by relevance, where the first line is the most relevant to you, and the last line is the least relevant.

3. If you are **unable** to choose at least **one** topic label, please explain why.

The following Example 1 contains 4 topic labels, and the following example 2 provides an example answer of why no topic label could be identified:

Example 1

"word3
word1 word4
word k word m (inferred words)
Sentence explaining the meaning."

Example 2

"I was unable to choose a topic label because word1, word4, and word5 are about one topic, and word2, word6, and word10 were about a completely different topic. The rest of the words did not make any meaningful connection to me"

*

Your answer

Figure C.7: Survey v2 p2.

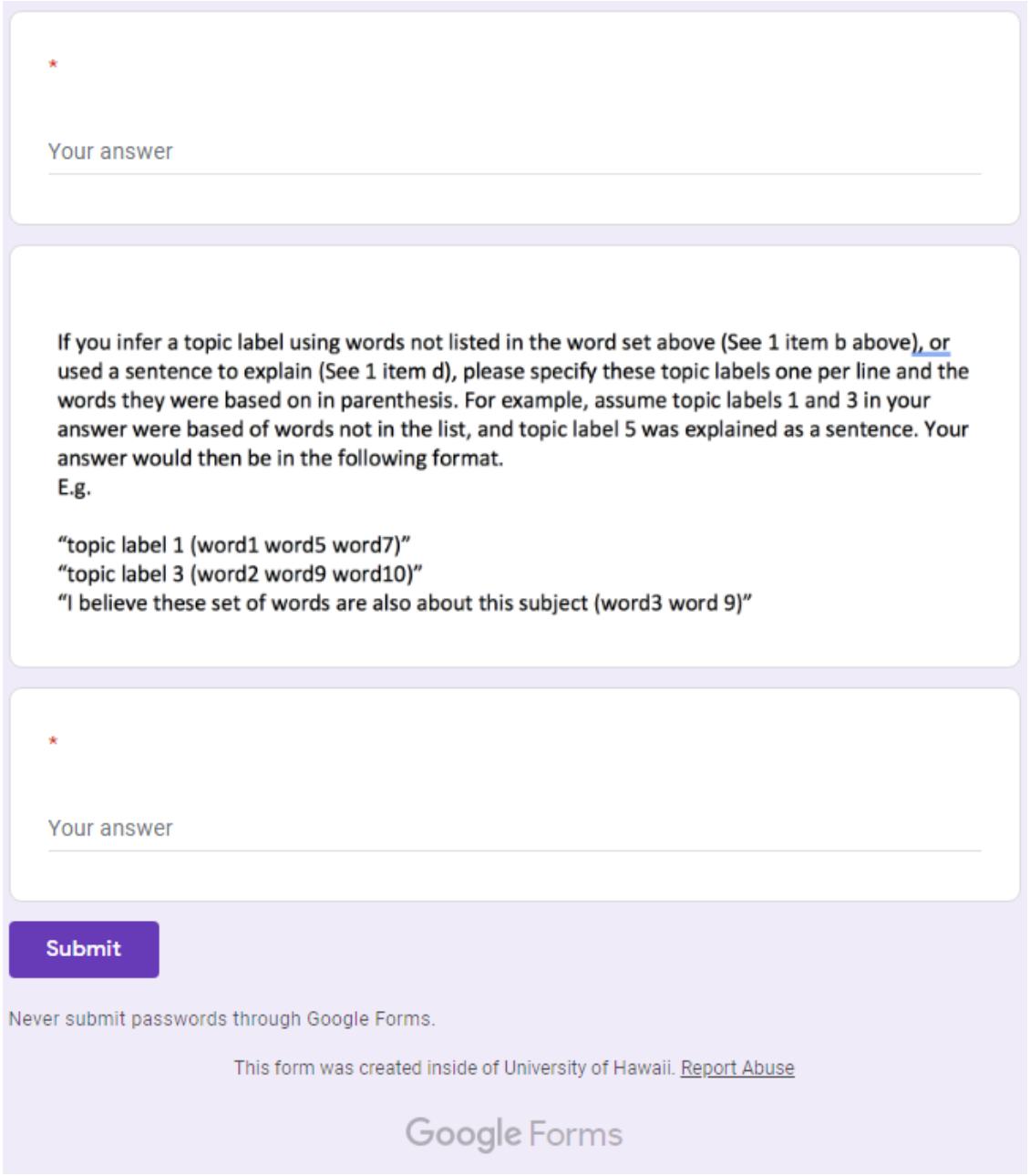


Figure C.8: Survey v2 p3.

Interpretability of Topic Model Results in the Aviation Safety Reporting System (ASRS)

The set of words displayed by a topic model--a computer program which automatically groups similar documents based on their content and provide set of words to describe them--are usually sensible, meaningful, interpretable and coherent. But some displayed sets of words, while statistically reasonable, are not particularly meaningful for human use. To evaluate our methods, we would like your judgment on how ``meaningful" some set of terms are. Here, we are purposefully vague about what is ``meaningful" ... it is some combination of coherent, interpretable, words-are-related, subject-heading-like, something-you-could-easily-label, etc.

We ask that you time your responses to this questionnaire. There will be questions within the questionnaire to indicate when to log the start and end times before and after each question.

* Required

Applicant ID *

Your answer

Please record, as accurately as possible, the time that you begin the portion of the questionnaire immediately below. *

Time

: AM

The following is a set of 10 words displayed by a topic model program attempting to describe a group of similar ASRS reports:

Figure C.9: Survey v4 p1.

The following is a set of 10 words displayed by a topic model program attempting to describe a group of similar ASRS reports:

smoke | cabin | fire | odor | captain | passengers | fumes | cockpit | maintenance | attendant

format: word1 | word2 | word3 | word4 | word5 | word6 | word7 | word8 | word9 | word10

1. Please **identify**, if possible, in the following answer one or more **topic labels** that you believe the set of words are about. For each topic label, you may: a) use one word from the list of provided words, b) infer one word that is not in this list but which you believe better describes the topic; c) use a combination of a few words (i.e., a phrase) that describes the topic; d) explain the meaning of the topic in a sentence; e) some other method.

2. Please **organize** the topic labels you have chosen in part 1 above in the following manner:

1) place one topic label per line
2) rank these topic labels by relevance, where the first line is the most relevant to you, and the last line is the least relevant.

3. If you are **unable** to choose at least **one** topic label, please explain why.

The following Example 1 contains 4 topic labels, and the following example 2 provides an example answer of why no topic label could be identified:

Example 1	Example 2
"word3 word1 word4 word k word m (inferred words) Sentence explaining the meaning."	"I was unable to choose a topic label because word1, word4, and word5 are about one topic, and word2, word6, and word10 were about a completely different topic. The rest of the words did not make any meaningful connection to me"

*

Your answer

Figure C.10: Survey v4 p2.

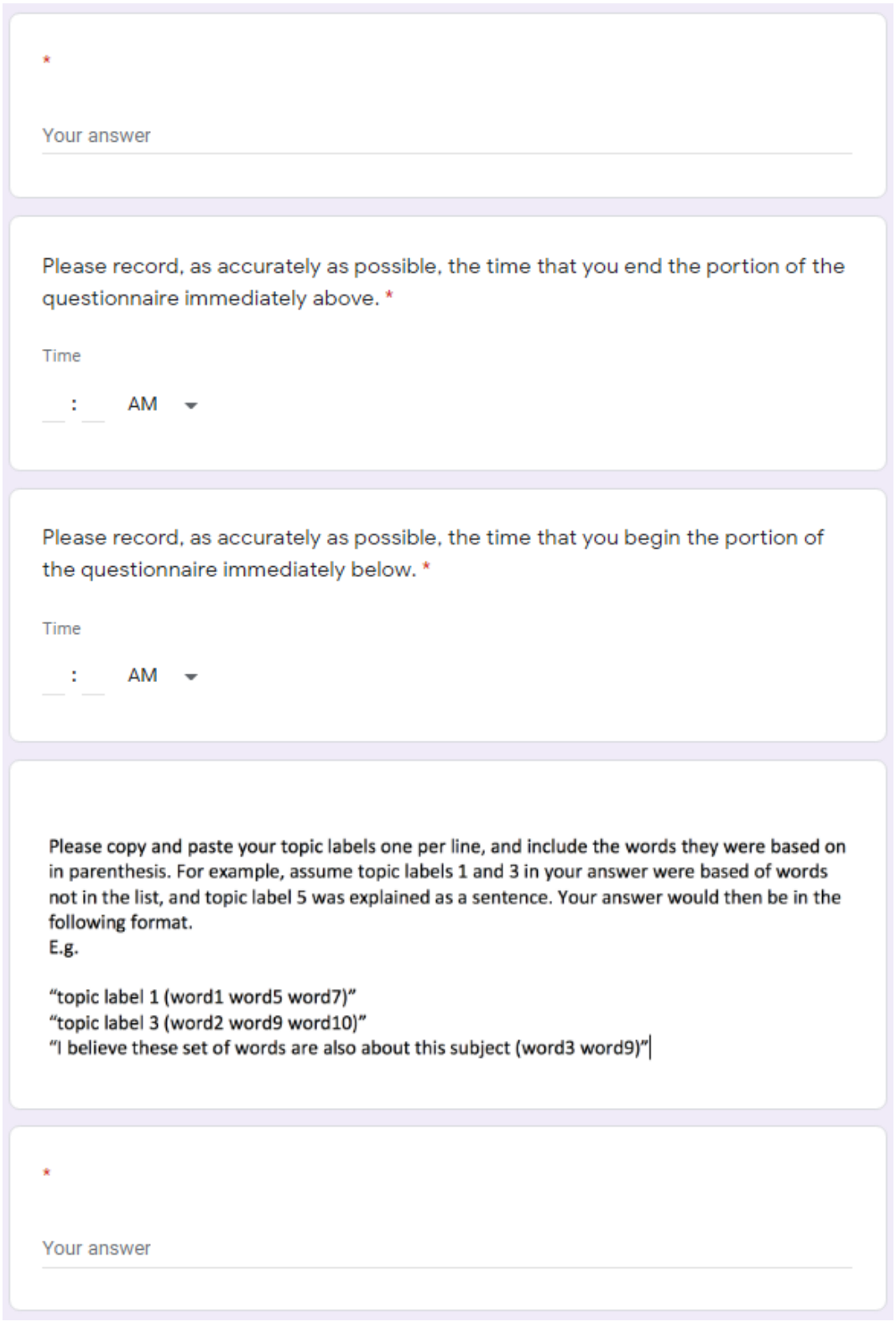


Figure C.11: Survey v4 p3.

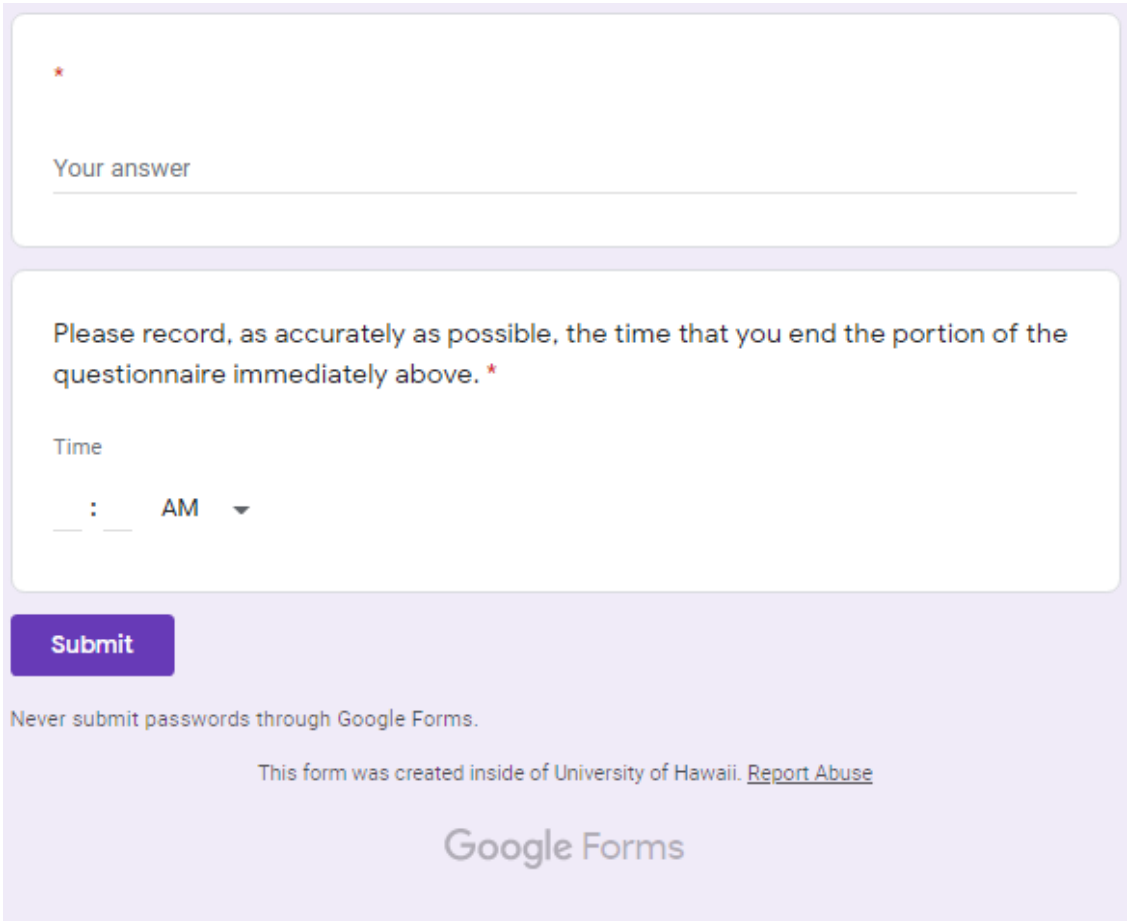


Figure C.12: Survey v4 p4.

Background Questionnaire: Interpretability of Topic Model Results in the Aviation Safety Reporting System (ASRS)

To better understand your responses to the questionnaire, it would be helpful for us to understand your level of familiarity with both the method and domain which we are applying to this research. Please let us know the following about yourself and ensure that you use the same application id that you received with the consent form (and which you used to respond to the study questionnaire):

* Required

Applicant ID *

Your answer

In the **last year**, have you (please select all that apply): *

- [ ] Read an article or presentation about topic modeling?
- [ ] Ever used a tool to perform topic modeling?
- [ ] Wrote code to perform topic modeling?
- [ ] Co-authored an article on topic modeling?
- [ ] None of the above

Figure C.13: Survey Background p1.

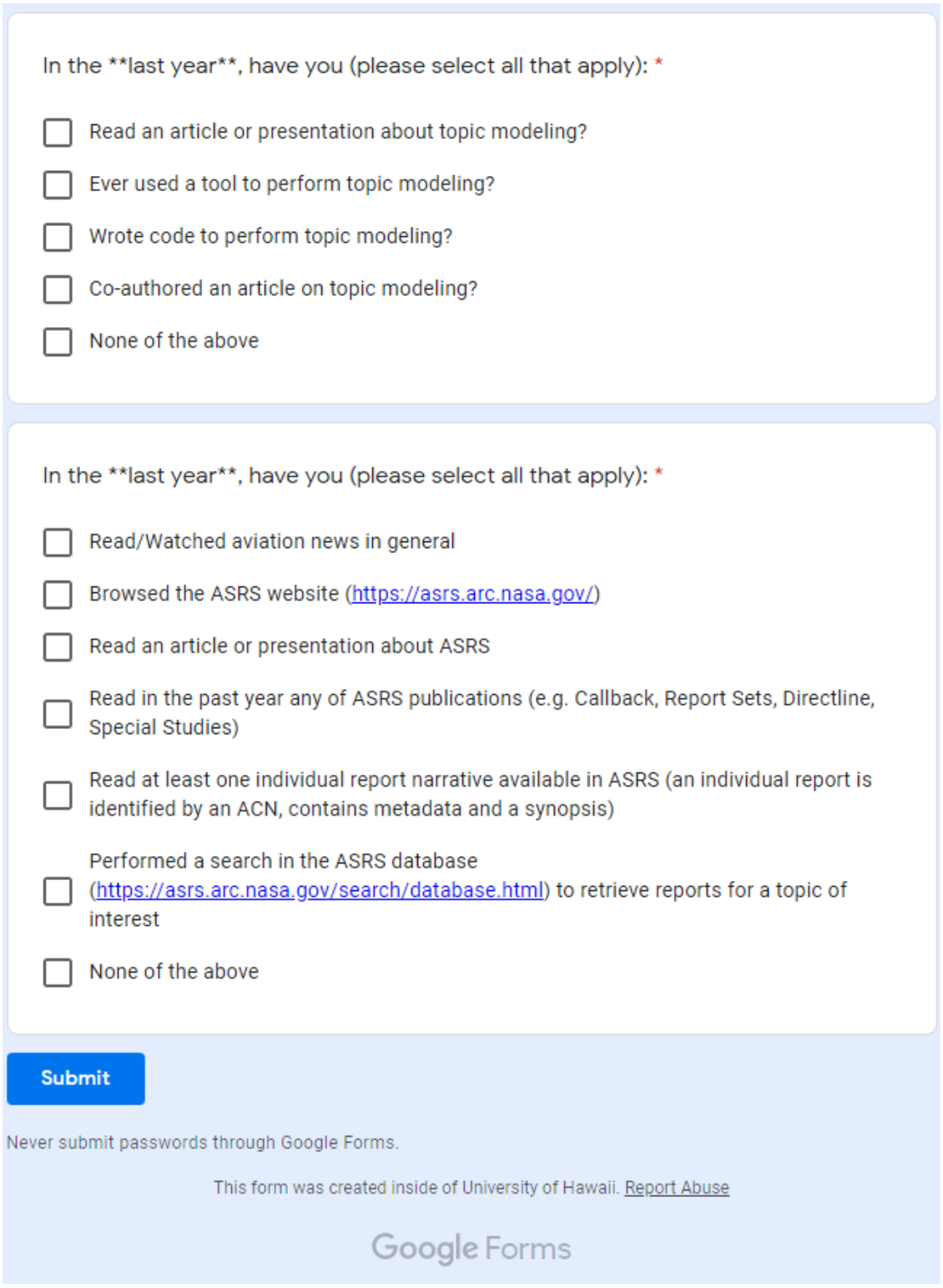


Figure C.14: Survey Background p2.

BIBLIOGRAPHY

[1] Amritanshu Agrawal, Wei Fu, and Tim Menzies. What is wrong with topic modeling? and how to fix it using search-based software engineering. Information and Software Technology, 98:74–88, 2018.

[2] M. Allahyari and K. Kochut. Semantic tagging using topic models exploiting wikipedia category network. In 2016 IEEE Tenth International Conference on Semantic Computing (ICSC), pages 63–70, 2016.

[3] P.D. Allison. Missing Data. Number no. 136 in Missing Data. SAGE Publications, 2001.

[4] H. Alves, B. Fonseca, and N. Antunes. Software metrics and security vulnerabilities: Dataset and exploratory study. In 2016 12th European Dependable Computing Conf, pages 37–44, Sept 2016.

[5] ASRS. Callback Number 317 March/April 2006. `https://asrs.arc.nasa.gov/publications/callback/cb_317.html`, 2006. [Online; accessed 18-June-2020].

[6] Ching-man Au Yeung, Nicholas Gibbins, and Nigel Shadbolt. Contextualising tags in collaborative tagging systems. In Proceedings of the 20th ACM Conference on Hypertext and Hypermedia, HT '09, page 251–260, New York, NY, USA, 2009. Association for Computing Machinery.

[7] L. Bilge, Y. Han, and M. Dell'Amico. Riskteller: Predicting the risk of cyber incidents. In Proc. 2017 ACM SIGSAC Conf on Computer and Communications Security, pages 1299–1311. ACM, 2017.

[8] CE Billings, JK Lauber, H Funkhouser, EG Lyman, and EM Huff. Nasa aviation safety reporting system. 1976.

[9] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan. Mining email social networks. In Proceedings of the 2006 International Workshop on Mining Software Repositories, MSR '06, page 137–143, New York, NY, USA, 2006. Association for Computing Machinery.

[10] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty. Latent dirichlet allocation. Journal of Machine Learning Research, 3, 2003.

[11] David M. Blei. Probabilistic topic models. Commun. ACM, 55(4):77–84, April 2012.

[12] M. Carvalho, J. DeMott, R. Ford, and D. A. Wheeler. Heartbleed 101. IEEE Security Privacy, 12(4):63–67, 2014.

[13] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L. Boyd-graber, and David M. Blei. Reading tea leaves: How humans interpret topic models. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, Advances in Neural Information Processing Systems 22, pages 288–296. Curran Associates, Inc., 2009.

[14] Hong-Mei Chen, Rick Kazman, Ira Monarch, and Ping Wang. Can cybersecurity be proactive? A big data approach and challenges. In 50th Hawaii International Conference on System Sciences, HICSS 2017, Hilton Waikoloa Village, Hawaii, USA, January 4-7, 2017, 2017.

[15] Jianfei Chen, Kaiwei Li, Jun Zhu, and Wenguang Chen. Warplda: a simple and efficient o(1) algorithm for latent dirichlet allocation. CoRR, abs/1510.08628, 2015.

[16] Jason Chuang, Christopher D. Manning, and Jeffrey Heer. Termite: Visualization techniques for assessing textual topic models. In Proceedings of the International Working Conference on Advanced Visual Interfaces, AVI '12, page 74–77, New York, NY, USA, 2012. Association for Computing Machinery.

[17] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '92, page 318–329, New York, NY, USA, 1992. Association for Computing Machinery.

[18] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, D. Adrian, Vern Paxson, M. Bailey, and J. A. Halderman. The matter of heartbleed. In Proceedings of the 2014 Conference on Internet Measurement Conference, page 475–488, 2014.

[19] Brigitte Endres-Niggemeyer, Jerry Hobbs, and K Sparck Jones. Summarizing text for intelligent communication. Internationales Begegnungs-und Forschungszentrum für Informatik, 1993.

[20] Mica Endsley. A taxonomy of situation awareness errors, human factors in aviation operations;. Proceedings of the 21st Conference of the European Association for Aviation Psychology (EAAP), 3:287–292, 01 1995.

[21] Q. Feng, Y. Cai, R. Kazman, Di Cui, Ting Liu, and Hongzhou Fang. Active hotspot: An issue-oriented model to monitor software evolution and degradation. In Proceedings of the 34th IEEE/ACM International Conference on Automated Software Engineering, ASE '19, page 986–997. IEEE Press, 2019.

[22] Q. Feng, R. Kazman, Y. Cai, R. Mo, and L. Xiao. Towards an architecture-centric approach to security analysis. In 2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA), pages 221–230, April 2016.

[23] Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. Frontiers in Genetics, 10:524, 2019.

[24] Muhammad Waleed Gondal, Manuel Wuthrich, Djordje Miladinovic, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Scholkopf, and Stefan Bauer. On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alche-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 15740–15751. Curran Associates, Inc., 2019.

[25] Philip B. Gough and William E. Tunmer. Decoding, reading, and reading disability. Remedial and Special Education, 7(1):6–10, 1986.

[26] Robert M. Groves, Floyd J. Fowler Jr., Mick P. Couper, James M. Lepkowski, Eleanor Singer, and Roger Tourangeau. Survey Methodology. Wiley, 2 edition, 07 2009.

[27] Manish Gupta, Rui Li, Zhijun Yin, and Jiawei Han. Survey on social tagging techniques. SIGKDD Explor. Newsl., 12(1):58–72, November 2010.

[28] M. Hafiz and M. Fang. Game of detections: how are security vulnerabilities discovered in the wild? Empirical Software Engineering, 21(5):1920–1959, Oct 2016.

[29] Robins JM Hernán MA. Causal Inference: What If. Handbooks of Sociology and Social Research. Boca Raton: Chapman & Hall/CRC, 2020.

[30] A. Hira, J. Alstad, and M. Konrad A. Investigating causal effects of software and systems engineering effort. In Joint Software and IT-Cost Forum 2020, 2020.

[31] H-F Hsieh and S. Shannon. Three approaches to qualitative content analysis. Qualitative Health Research, 15(9):1277–1288, 2005.

[32] M. Joblin, W. Mauerer, S. Apel, J. Siegmund, and D. Riehle. From developer networks to verified communities: A fine-grained approach. In 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, volume 1, pages 563–573, 2015.

[33] Mitchell Joblin. Structural and Evolutionary Analysis of Developer Networks. PhD thesis, Universitat Passau, 2017.

[34] Debra Jones and Mica Endsley. Sources of situation awareness errors in aviation. Aviation, space, and environmental medicine, 67:507–12, 07 1996.

[35] K. Krippendorff. Content Analysis: An Introduction to Methodology. Sage Publications, Inc., Beverly Hills, CA, 1980.

[36] R. Lagerström, C. Baldwin, A. MacCormack, D. Sturtevant, and L. Doolan. Exploring the relationship between architecture coupling and software vulnerabilities. In Engineering Secure Software and Systems, pages 53–69, 06 2017.

[37] Lucas Layman, Allen P. Nikora, Joshua Meek, and Tim Menzies. Topic modeling of nasa space system problem reports: Research in practice. In Proceedings of the 13th International Conference on Mining Software Repositories, MSR '16, page 303–314, New York, NY, USA, 2016. Association for Computing Machinery.

[38] Zhen Liao, Maoqiang Xie, Hao Cao, and Yalou Huang. A probabilistic ranking approach for tag recommendation. ECML PKDD Discovery Challenge 2009 (DC09), page 143, 2009.

[39] Y. Liu, A. Sarabi, J. Zhang, P. Naghizadeh, M. Karir, M. Bailey, and M. Liu. Cloudy with a chance of breach: Forecasting cyber security incidents. In Proc. 24th USENIX Conf on Security Symposium, SEC'15, pages 1009–1024, Berkeley, CA, USA, 2015. USENIX Association.

[40] Simone Magnoni. An approach to measure Community Smells in software development communities. PhD thesis, Politecnico Milano, 2016.

[41] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to Information Retrieval. Cambridge University Press, USA, 2008.

[42] Mika V. Mantyla, Maelick Claes, and Umar Farooq. Measuring lda topic stability from clusters of replicated runs. In Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM '18, New York, NY, USA, 2018. Association for Computing Machinery.

[43] R. Martin, S. Christey, and D. Baker. A progress report on the cve initiative, June 2002.

[44] N. McNeil, R. A. Bridges, M. D. Iannacone, B. D. Czejdo, N. Perez, and J. R. Goodall. Pace: Pattern accurate computationally efficient bootstrapping for timely discovery of cyber-security concepts. 12th Intl Conf on Machine Learning and Applications, 2:60–65, 2013.

[45] Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. Automatic labeling of multinomial topic models. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '07, page 490–499, New York, NY, USA, 2007. Association for Computing Machinery.

[46] T. Menzies, L. Williams, and T. Zimmermann. Perspectives on Data Science for Software Engineering. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2016.

[47] Cédric S. Mesnage and Mark J. Carman. Tag navigation. In Proceedings of the 2nd International Workshop on Social Software Engineering and Applications, SoSEA '09, page 29–32, New York, NY, USA, 2009. Association for Computing Machinery.

[48] David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11, page 262–272, USA, 2011. Association for Computational Linguistics.

[49] S. Mittal, P. K. Das, V. Mulwad, A. Joshi, and T. Finin. Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities. In 2016 IEEE/ACM Intl Conf on Advances in Social Networks Analysis and Mining, pages 860–867, Aug 2016.

[50] R. Mo, Y. Cai, R. Kazman, L. Xiao, and Q. Feng. Architecture anti-patterns: Automatically detectable violations of design principles. IEEE Transactions on Software Engineering, 2019.

[51] R. Mo, Y. Cai, R. Kazman, Lu Xiao, and Q. Feng. Decoupling level: a new metric for architectural maintenance complexity. In Proceedings of the 38th International Conference on Software Engineering, pages 499–510, 2016.

[52] S.L. Morgan. Handbook of Causal Analysis for Social Research. Handbooks of Sociology and Social Research. Springer Netherlands, 2013.

[53] Katharine Mullen, David Ardia, David Gil, Donald Windover, and James Cline. Deoptim: An r package for global optimization by differential evolution. Journal of Statistical Software, Articles, 40(6):1–26, 2011.

[54] S. Murtaza, W. Khreich, A. Hamou-Lhadj, and A. Bener. Mining trends and patterns of software vulnerabilities. J. Syst. Softw., 117(C):218–228, July 2016.

[55] S. Neuhaus and T. Zimmermann. Security trend analysis with cve topic models. In IEEE 21st Intl Symposium on Software Reliability Engineering, pages 111–120, 2010.

[56] S. Neuhaus and T. Zimmermann. Security trend analysis with cve topic models. In 2010 IEEE 21st International Symposium on Software Reliability Engineering, pages 111–120, 2010.

[57] David Newman, Youn Noh, Edmund Talley, Sarvnaz Karimi, and Timothy Baldwin. Evaluating topic models for digital libraries. In Proceedings of the 10th Annual Joint Conference on Digital Libraries, JCDL '10, page 215–224, New York, NY, USA, 2010. Association for Computing Machinery.

[58] Jessica Lang Nowinski, Jon B. Holbrook, and R. Key Dismukes. Human memory and cockpit operations: An asrs study. Technical report, NASA Ames Research Center, 2003.

[59] E. Nunes, A. Diab, A. Gunn, E. Marin, V. Mishra, V. Paliath, J. Robertson, J. Shakarian, A. Thart, and P. Shakarian. Darknet and deepnet mining for proactive cybersecurity threat intelligence. In 2016 IEEE Conf on Intelligence and Security Informatics, pages 7–12, Sept 2016.

[60] J.V. Oakhill, K. Cain, and P.E. Bryant. The dissociation of word reading and text comprehension: Evidence from component skills. Language and Cognitive Processes, 18(4):443–468, 2003.

[61] N. Oza, J. P. Castle, and J. Stutz. Classification of aeronautics system health and safety documents. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 39(6):670–680, 2009.

[62] F. Palomba, D. A. A. Tamburri, F. Arcelli Fontana, R. Oliveto, A. Zaidman, and A. Serebrenik. Beyond technical aspects: How do community smells influence the intensity of code smells? IEEE Transactions on Software Engineering, pages 1–1, 2018.

[63] Carlos Paradis, Rick Kazman, Misty Davies, and Becky Hooey. Augmenting Topic Finding in the NASA Aviation Safety Reporting System using Topic Modeling.

[64] Carlos V. Paradis, Rick Kazman, and Ping Wang. Indexing text related to software vulnerabilities in noisy communities through topic modelling. In 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018, Orlando, FL, USA, December 17-20, 2018, pages 763–768, 2018.

[65] Judea Pearl. Causal inference in the health sciences: A conceptual introduction. Health Services and Outcomes Research Methodology, 2(3):189–220, 2001.

[66] Joseph Ramsey, Madelyn Glymour, Ruben Sanchez-Romero, and Clark Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. International Journal of Data Science and Analytics, 3(2):121–129, 2017.

[67] Saul D. Robinson. Multi-label classification of contributing causal factors in self-reported safety narratives. Safety, 4(3), 2018.

[68] C. Sabottke, Suciu O, and T. Dumitras. Vulnerability disclosure in the age of social media: Exploiting twitter for predicting real-world exploits. In 24th USENIX Security Symposium, pages 1041–1056, 2015.

[69] Nadine B. Sarter and Heather M. Alexander. Error types and related error detection mechanisms in the aviation domain: An analysis of aviation safety reporting system incident reports. The International Journal of Aviation Psychology, 10(2):189–206, 2000.

[70] C. Sievert and K. Shirley. Ldavis: A method for visualizing and interpreting topics, 06 2014.

[71] A. Smith, S. Malik, and B. Shneiderman. Visual Analysis of Topical Evolution in Unstructured Text: Design and Evaluation of TopicFlow, pages 159–175. Springer, 2015.

[72] K. Soska and N. Christin. Automatically detecting vulnerable websites before they turn malicious. In 23rd USENIX Security Symposium, pages 625–640, San Diego, CA, 2014.

[73] Peter Spirtes. Introduction to causal inference. J. Mach. Learn. Res., 11:1643–1662, August 2010.

[74] Rainer Storn and Kenneth Price. Differential evolution –a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11(4):341–359, 1997.

[75] T.C. Summers, K.J. Lyytinen, T Lingham, and E Pierce. How hackers think: A study of cybersecurity experts and their mental models, 2013.

[76] Aviation Safety Report System. Program Briefing. `https://asrs.arc.nasa.gov/overview/summary.html`, 2019. [Online; accessed 18-June-2020].

[77] D. Tamburri, K. Blincoe, F. Palomba, and R. Kazman. "the canary in the coal mine...": A cautionary tale from the decline of sourceforge. Software: Practice and Experience, 2020.

[78] D. Tamburri, F. Palomba, and R. Kazman. Exploring community smells in open-source: An automated approach. IEEE Transactions on Software Engineering, PP:1–1, 02 2019.

[79] D. A. Tamburri, R. Kazman, and Willem-Jan van den Heuvel. Splicing community and software architecture smells in agile teams: An industrial study. In Proc. HICSS, pages 1–11, 2019.

[80] D. A. Tamburri, P. Lago, and Hans van Vliet. Uncovering latent social communities in software development. IEEE Software, 30(1):29 –36, jan.-feb. 2013.

[81] D. Andrew Tamburri, P. Kruchten, P. Lago, and Hans van Vliet. Social debt in software engineering: insights from industry. J. Internet Services and Applications, 6(1):10:1–10:17, 2015.

[82] S. Trabelsi, H. Plate, A. Abida, M. M. Ben Aoun, A. Zouaoui, C. Missaoui, S. Gharbi, and A. Ayari. Mining social networks for software vulnerabilities monitoring. In 2015 7th Intl Conf on New Technologies, Mobility and Security, pages 1–7, July 2015.

[83] F. Tsai and K. L. Chan. Blog Data Mining for Cyber Security Threats, pages 169–182. Springer, 2009.

[84] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. J. Mach. Learn. Res., 11:2837–2854, December 2010.

[85] D. Votipka, R. Stevens, E. Redmiles, J. Hu, and M. Mazurek. Hackers vs. testers: A comparison of software vulnerability discovery processes. In 2018 IEEE Symposium on Security and Privacy, pages 134–151, 2018.

[86] J. Walden, J. Stuckman, and R. Scandariato. Predicting vulnerable components: Software metrics vs text mining. In IEEE 25th Intl Symposium on Software Reliability Engineering, pages 23–33, 2014.

[87] S. Wasserman and K. Faust. Social network analysis : methods and applications. University Press, Cambridge, 1995.

[88] S. Weerawardhana, S. Mukherjee, I. Ray, and A. Howe. Automated Extraction of Vulnerability Information for Home Computer Security, pages 356–366. Springer, 2015.

[89] D. A. Wheeler. Preventing heartbleed. Computer, 47(08):80–83, aug 2014.

[90] S. Wong, Y. Cai, G. Valetto, G. Simeonov, and K. Sethi. Design rule hierarchies and parallelism in software development tasks. In 2009 IEEE/ACM International Conference on Automated Software Engineering, pages 197–208, 2009.

[91] Xiaoge Zhang and Sankaran Mahadevan. Ensemble machine learning models for aviation incident risk prediction. Decision Support Systems, 116:48 – 63, 2019.

[92] J. Zhu and J. Wei. An empirical study of multiple names and email addresses in oss version control repositories. In 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR), pages 409–420, 2019.