ote technical note technica

Analysis of Technical Operations Job Tasks
Final Report

Andrew J. Abbate and Ellen J. Bass

December 28, 2017

DOT/FAA/AR-TN

This document is available to the U.S. public through the National Technical Information Services (NTIS), Springfield, Virginia 22161.

This document is also available from the Federal Aviation Administration William J. Hughes Technical Center at actlibrary.tc.faa.gov.

U.S. Department of Transportation

Federal Aviation Administration

NOTICE

This document is disseminated under the sponsorship of the U.S. Department of Transportation in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof. The U.S. Government does not endorse products or manufacturers. Trade or manufacturers' names appear herein solely because they are considered essential to the objective of this report. The findings and conclusions in this report are those of the author(s) and do not necessarily represent the views of the funding agency. This document does not constitute FAA policy. Consult the FAA sponsoring organization listed on the Technical Documentation page as to its use.

This report is available at the Federal Aviation Administration William J. Hughes Technical Center's Full-Text Technical Reports page: actlibrary.tc.faa.gov in Adobe Acrobat portable document format (PDF).

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
DOT/FAA/AR-xx/xx		

4. Title and Subtitle	5. Report Date
Analysis of Technical Operations Job Tasks: Final Report	December 2017
	6. Performing Organization Code

7. Authors	8. Performing Organization Report No.
Andrew J. Abbate and Ellen J. Bass	

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS).
Drexel University 3141 Chestnut Street Philadelphia, PA 19104	
	11. Contract or Grant No. AN002

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
	14. Sponsoring Agency Code AJI-2400

15. Supplementary Notes

16. Abstract

The target outcome of this research is to develop recommendations for an approach to analyzing the alignment of existing technical operations (Tech Ops) courses and job task analysis (JTA) data. Two main tasks support this outcome:

1. Task identification and analysis
2. Training alignment analysis

This document supports the target outcome of this research. We accomplish this by integrating results of the first two tasks, identifying potentially useful tools and algorithms from the text search literature, and recommending a text search approach to alignment.

In support of Task 1, we reviewed the JTA data and materials for eight Tech Ops courses of interest selected by the FAA. Our analyses identified what JTA data and what types of files within the Tech Ops course curriculum would be applicable to Task 2.

In support of Task 2, we identified what capabilities an approach to alignment should support. The analyst should be able to compare all job task statements to all of the curriculum and to specify analyses by identifying the job task statement or statements as well as the Tech Ops curriculum courses of interest. Based on the analyst's selections, the system should compare full-sentence job task statements from JTA workbooks and sentences from text-based training documents. Matching can include exact word-for-word text matches and partial matches, where partial matches support differences such as words with different letter case, words with different stems, acronyms, initialisms, abbreviations, and synonyms. The system should identify what partial matches are needed for a text-based document to be considered aligned. Alignment reports should identify what training documents are aligned with what task statements, what training documents are not aligned, and where these documents are in the corpus.

To support target outcome of this research, we reviewed text search literature that is relevant to the files specified in Task 1 and the approach described in Task 2. Our review focused on tools and algorithms for getting text from text-based training documents and JTA task statements, preparing text from the task statements and training documents so that they can be compared, comparing the text to identify what training documents are aligned/not aligned, and rendering outputs that the user can analyze. We leveraged the identified tools and algorithms to recommend a text search approach to alignment. We identified what text search operations should be performed by the analyst and by the system, when each operation should be performed (i.e., before, during, and after the search), what extant tools and algorithms support each operation, and how each operation maps to an element of the approach specified in Task 2. We then identified future work that is needed to address other file types identified in Task 1 and additional alignment-related considerations. The AN003 report extends this research by reviewing output-presentation techniques from the human-computer interaction (HCI) literature, identifying additional searching capabilities (*keyword search* and *refined search*), and recommending a visual presentation of results.

17. Key Words	18. Distribution Statement
	This document is available to the U.S. public through the National Technical Information Service (NTIS), Springfield, Virginia 22161. This document is also available from the Federal Aviation Administration William J. Hughes Technical Center at actlibrary.tc.faa.gov.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. Pages	22. Price
Unclassified	Unclassified	60	

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

EXECUTIVE SUMMARY

The target outcome of this research is to develop recommendations for an approach to analyzing the alignment of existing technical operations (Tech Ops) courses and job task analysis (JTA) data. Two main tasks support this outcome:

1. Task identification and analysis
2. Training alignment analysis

This document supports the target outcome of this research. We accomplish this by integrating results of the first two tasks, identifying potentially useful tools and algorithms from the text search literature, and recommending a text search approach to alignment.

In support of Task 1, we reviewed the JTA data and materials for eight Tech Ops courses of interest selected by the FAA. Our analyses identified what JTA data and what types of files within the Tech Ops course curriculum would be applicable to Task 2.

In support of Task 2, we identified what capabilities an approach to alignment should support. The analyst should be able to compare all job task statements to all of the curriculum and to specify analyses by identifying the job task statement or statements as well as the Tech Ops curriculum courses of interest. Based on the analyst's selections, the system should compare full-sentence job task statements from JTA workbooks and sentences from text-based training documents. Matching can include exact word-for-word text matches and partial matches, where partial matches support differences such as words with different letter case, words with different stems, acronyms, initialisms, abbreviations, and synonyms. The system should identify what partial matches are needed for a text-based document to be considered aligned. Alignment reports should identify what training documents are aligned with what task statements, what training documents are not aligned, and where these documents are in the corpus.

To support target outcome of this research, we reviewed text search literature that is relevant to the files specified in Task 1 and the approach described in Task 2. Our review focused on tools and algorithms for getting text from text-based training documents and JTA task statements, preparing text from the task statements and training documents so that they can be compared, comparing the text to identify what training documents are aligned/not aligned, and rendering outputs that the user can analyze. We leveraged the identified tools and algorithms to recommend a text search approach to alignment. We identified what text search operations should be performed by the analyst and by the system, when each operation should be performed (i.e., before, during, and after the search), what extant tools and algorithms support each operation, and how each operation maps to an element of the approach specified in Task 2. We then identified future work that is needed to address other file types identified in Task 1 and additional alignment-related considerations. The AN003 report extends this research by reviewing output-presentation techniques from the human-computer interaction (HCI) literature, identifying additional searching capabilities (*keyword search* and *refined search*), and recommending a visual presentation of results.

INTRODUCTION

The training curriculum for Federal Aviation Administration (FAA) Technical Operations (Tech Ops) must be updated to maintain alignment with evolving job tasks. Currently, the FAA has job task analysis (JTA) data that identify what job tasks the training curriculum should support. Job tasks in the JTA are full-sentence statements (e.g. "Test the MDS," "Replace the CPME power supply"). The Tech Ops training curriculum contains text-based documents (e.g. 40289_SG_2-09_VSCS_BK_02_VoiceChannelTesting.doc) as well as other types of files (e.g. Adobe Flash). In order to maintain alignment, task statements in the JTA should be analyzed with respect to text-based documents in the training curriculum. One way to accomplish this is by identifying what text-based documents are aligned and not aligned with respect to the job task statements.

A text-based training document could be considered aligned if its text is relevant to one or more job task statements. One way of determining alignment could be relatively easy: search through a document for text that matches one or more task statements exactly (i.e., character for character). If a document has text matching at least one task statement, then it is relevant, and the document can be considered aligned. Otherwise, if no text matches *any* task statements, then the document might reflect an alignment problem.

We compared all 13,129 task statements in the FAA-supplied Tech Ops JTA workbooks to text-based documents from eight FAA-supplied Tech Ops curriculum courses. No exact matches were found. This indicates that a different approach is necessary.

It is unlikely that all of the FAA-supplied curriculum is irrelevant to the complete FAA-supplied JTA. Thus, it was important to identify what was missed: documents having relevant text that matches a task statement partially.

We hypothesized that some documents having relevant text were missed because they have partially matching sentences, such as sentences that have some exactly matching words, but not all of them; all exactly matching words, but in a different order; or some combination of both. These sentence level issues are further complicated if some exactly matching words are more important than others, such as verbs and nouns that are relatively more important than articles (e.g. "the"), propositions (e.g. "of"), and conjunctions (e.g. "and").

We also hypothesized that some documents having relevant text were missed because they have partially matching words (i.e., *word level* partial matches), such as words that

- Have punctuation characters inserted
- Begin with a different letter case
- Have the same root word, but a different stem (e.g. "Monitoring" instead of "Monitor," where "Monitor" is the root and "ing" is the stem)
- Are the spelled-out form of an acronym, initialism, or abbreviation
- Are different, but capture the same concept (e.g. "Fix" instead of "Repair")
- Are different, but capture a related concept (e.g. "Observe" instead of "Interpret")

Partially matching text can include combinations of the matches above. For example, a partially matching word could have the same root, but with punctuation inserted. Other combinations could operate on both the word and sentence levels. For example, a document having relevant text could be missed because it has a sentence with one or more partially matching words that are also in the correct order.

An approach to supporting an analyst with identifying partially matching text that is relevant to one or more task statements is a goal of the operational concept discussed next.

OPERATIONAL CONCEPT

To aid in evaluating alignment, FAA analysts could benefit from an improved approach for identifying what text-based documents in the Tech Ops training curriculum are relevant to one or more JTA task statements. Such an approach should have data entry capabilities for the user to enter inputs of interest (Fig. 1a); data handling capabilities for utilizing the analyst's inputs (Fig. 1b); processing algorithms that can identify relevant text (Fig. 1c); and reporting capabilities that inform the analyst about what training document inputs are aligned with what task statements, what training document inputs are not aligned, and where these documents are in the corpus (i.e., what directories/sub-directories) (Fig. 1d).

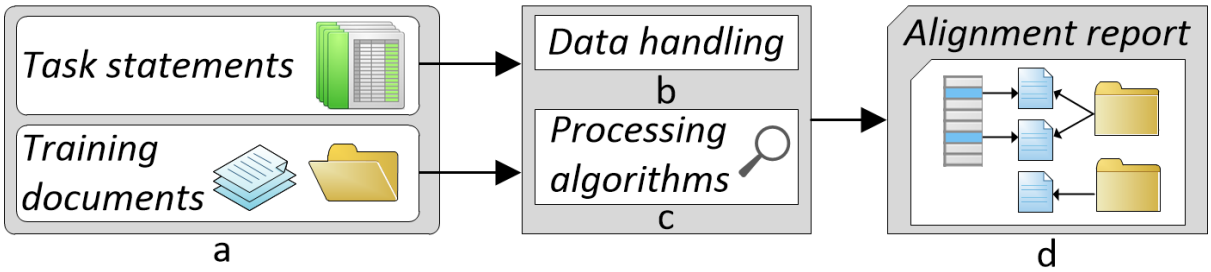

Figure 1: A text search approach to training document alignment with job task statements. (a) Inputs, including task statements from the JTA workbooks and text-based training documents from the Tech Ops curriculum corpus. (b) Data handling techniques that support utilization of data from the inputs. (c) Processing algorithms that can identify what training documents have text that is relevant to the task statements. (d) An alignment report that identifies what training documents are aligned with what task statements, what training documents are not aligned, and the locations of each training document within the corpus.

DATA SELECTION CAPABILITIES. The analyst should be able to compare all job task statements to all of the curriculum and to specify analyses by identifying the job task statement or statements as well as the Tech Ops curriculum of interest. Documents in the Tech Ops training curriculum are organized in a hierarchical structure of directories and sub-directories. Thus, the analyst should be able to select files of interest at the file, directory, and sub-directory levels.

One task statement could be needed to determine if a text-based training document is aligned; however, all task statements may be needed to be certain that a document is *not* aligned. Thus, the analyst should be able to select job task statements of interest for an alignment analysis. To support identifying what text-based training documents are not aligned, the analyst should be able to select the JTA workbook files containing job task statements of interest. To support identifying what text-based training documents are aligned, the analyst should be able to select one or more job task statements of interest from one or more JTA workbook files.

DATA HANDLING CAPABILITIES.

Data Extraction. For the inputs above, two kinds of data are applicable:

1. Text data from the text-based training documents and all job task statements, which are needed to analyze alignment
2. Metadata from the text-based training documents (i.e., filename, directory, subdirectories), which are needed to support alignment reporting

The FAA's text-based training documents have the formats shown in Table 1. When the analyst enters text-based training document inputs, text data and metadata need to be extracted. Thus, the approach should be capable of extracting these data from text-based documents having applicable formats (Table 1).

Table 1: Applicable text-based document formats in the Tech Ops curriculum.

Extension	Stands for (extension letters in bold)
.doc	Microsoft Office **DOC**ument
.docx	Microsoft Office **DOC**ument **X**ML
.pdf	Adobe **P**ortable **D**ocument **F**ormat
.ppt	Microsoft **P**ower**P**oin**T**
.pptx	Microsoft **P**ower**P**oin**T** **X**ML
.xls	Microsoft **E**xce**L** **S**preadsheet
xlsx	Microsoft **E**xce**L** **S**preadsheet **X**ML

Text data from the JTA workbooks are in columns and spreadsheets of Excel workbook files (.xlsx format) (see Abbate et al. (2017)). When the analyst enters JTA workbook inputs, text data need to be extracted. Thus, the approach should be capable of extracting text data from columns, spreadsheets, and workbooks that are in a .xlsx format.

Data Retrieval. Data retrieval refers to the process of getting a subset of the extracted data that is needed to support the alignment analysis and alignment reports. To support the alignment analysis, text data should be retrieved from each text-based training document and JTA task statement so that they can be compared. To support alignment reports, metadata needs to be retrieved from text-based documents so that their filenames, directories, and subdirectories can be presented to the analyst.

PROCESSING ALGORITHM CAPABILITIES. Processing algorithms are needed to determine what training documents are aligned/not aligned. This can be accomplished using text comparison algorithms that compare the text data from each task statement to the text of each training document. Text pre-processing algorithms could perform operations on the text data that support different kinds of comparisons.

Comparing Text Data. Text comparison algorithms are needed to determine what training documents contain text that is relevant to one or more task statements. In this research, matching text is proxy for relevance, and relevance is a proxy for alignment.

Partially Matching Text. As mentioned, one problem with defining alignment with respect to exactly matching text is that such a definition ignores partially matching text that is relevant. With respect to text-based training documents of interest, we hypothesized that relevant sentences were missed because they partially matched one or more full-sentence job task statements, while relevant words were missed because they partially matched words of one or more job task statements. Thus, an improved approach to alignment should define alignment formally with respect to exactly matching text *and* partially matching text.

In order to define alignment formally with respect to partially matching text, the approach should unambiguously specify what properties are measurable for sentences and what properties are measurable for words (Table 2). Sentence property measures compare a sentence in a text-based document to a task statement. They include overlap (i.e., how many exactly matching words are in a sentence) and ordering (i.e., how many exactly matching words are in the correct order) (Achananuparp et al., 2008). Word property measures compare words in a training document to words in a task statement. They include semantics (i.e., meaning of a non-matching word), orthography (i.e., spelling of a non-matching word), and morphology (i.e., part of speech of a matching word and root of a non-matching word) (Chomsky, 2014).

Table 2: Properties of partially matching text.

Sentence property		Word property			
Overlap	Ordering	Semantics	Orthography	Morphology	
				Part of speech	Root
Some word(s) in a sentence are exactly matching	Exactly matching words appear in the correct order	Non-matching word(s) have a similar meaning	Non-matching word(s) are the same, but spelled as an acronym, initialism, or abbreviation	Matching word(s) have the same part of speech	Non-matching word(s) have the same root

To characterize properties of partially matching sentences (overlap and ordering), consider an exemplar task statement from the JTA, "Troubleshoot other ARTCC interfaces." Regarding overlap, one partially matching sentence in a training document could include all four words, while another could include three of four: "Troubleshoot," "ARTCC," and interfaces." In this case, the analyst would need to consider whether the word "other" is relevant to alignment.

Regarding ordering, a sentence and a task statement could have overlapping words, but in different orders. For example, one training document could have a sentence with "interfaces" before "other ARTCC," while another document could have a sentence with "ARTCC interfaces" before "other." In this case, the analyst would need to consider if and how the ordering of "interfaces" affects alignment.

To characterize properties of partially matching words (semantics, morphology, and orthography), consider the words "Troubleshoot" and "ARTCC" from the exemplar task statement. Suppose a sentence in a training document is "Fix other ARTCC interfaces." Here, the word "Fix" could be a semantic match for "Troubleshoot" because it is a synonym.

For morphology and orthography, consider another hypothetical sentence in a training document: "Troubleshooting is important because the Air Route Traffic Control Centers

interface with other devices." Regarding part of speech morphology, "interfaces" is an exactly matching word, but it is a noun in the task statement and a verb in the sentence, which could be a poor match.

Regarding root morphology, "troubleshooting" contains the root word "Troubleshoot," which matches the task statement exactly (the stem, "ing," does not match). Here, the analyst would need to consider how the word's stem ("ing") affects alignment.

Regarding orthography, "Air Route Traffic Control Centers" is the spelled-out equivalent of the initialism "ARTCC," which could be interpreted as an exactly matching word. Similar orthography properties would apply if words are spelled with different letter cases (e.g. "Interfaces" instead of "interfaces") or with added punctuation (e.g. "inter-faces" instead of "interfaces").

In addition to individual word and sentence properties, the analyst might also need to measure combined properties (Table 3). To characterize combined properties of partially matching text, consider the hypothetical training document sentence, "Fix other ARTCC interfaces." As described above, "Fix" is a synonym of "Troubleshoot" (i.e., it is a semantic partial match). Additionally, "Fix" is a verb like "Troubleshoot" (i.e., it is a part of speech morphology partial match), and it is also the first word in the sentence like "Troubleshoot" (i.e., it is a sentence level ordering partial match).

Other word properties (i.e., orthography and root morphology) could combine with sentence level ordering, since any partially matching word could also be in the correct order. Additional combinations include root/part of speech morphologies, such as verbs that are in different tenses, and orthography/part of speech morphology, such as nouns that are spelled as an acronym, initialism, or abbreviation. These combinations could combine again with the sentence property of ordering, such as nouns that are spelled as an acronym, initialism, or abbreviation and are also in the correct order.

Table 3: Combined properties of partially matching text.

Partial-match property	Part of speech morphology	Ordering
Semantics	Word has a similar meaning and the same part of speech	Word has a similar meaning and appears in the correct order
Root morphology	Word has same root and same part of speech	Word has same root and appears in correct order
Orthography	Same word is spelled as an acronym, initialism, or abbreviation and has same part of speech	Same word is spelled as an acronym, initialism, or abbreviation and appears in correct order

Text Comparison Algorithms. Applicable properties of matching text are exact matches, sentence partial matches, word partial matches, and combined partial matches. Text comparison algorithms could aid in identifying these matches.

Another way of characterizing properties of matching text is by what units of text can exhibit them (referred to as *comparable units* hereinafter). Applicable comparable units include

- Words, which are contiguous strings of one or more characters having no whitespace between them
- Sentences, which are contiguous strings of whitespace-separated words

- Sentences with words, which are arrays of sequentially ordered words that can be joined to form individual sentences (with the ordering of words preserved)

These three comparable units are needed to define unambiguous inputs to text comparison algorithms that can identify each property. Below we specify what comparable units could reflect each property and what each property means with respect to the comparable units. (Note: all measures for which comparable units are words also apply if the comparable units are sentences with words.)

Comparable units that are sentences include one sentence of a training document and one full-sentence task statement of a JTA workbook. The only property that is applicable to these comparable units is *exact match*, which means that a training document has sentences that match a task statement character for character.

Comparable units that are words include one word of a training document and one word of a JTA workbook task statement. Eight properties of partially matching text are measurable with respect to these comparable units (Fig. 2). Comparable units that are

1. *Overlap*, which means that training document has words that match task statement words character for character
2. *Semantics*, which means that a training document has words that are semantically equivalent or related to task statement words
3. *Orthography*, which means that a training document has words that are the same as task statement words, but spelled as an acronym, initialism, or abbreviation
4. *Part of speech morphology*, which means that a training document has words that match task statement words character for character and also have the correct parts of speech
5. *Root morphology*, which means that a training document has words with the same roots as task statement words
6. *Semantics and part of speech morphology*, which means that a training document has words that are semantically equivalent or related to task statement words and also have the correct parts of speech
7. *Root and part of speech morphology*, which means that a training document has words with the same roots and parts of speech as task statement words
8. *Orthography and part of speech morphology*, which means that a training document has words that are the same as task statement words, but are spelled as acronyms, initialisms, or abbreviations, and also have the correct parts of speech

Figure 2: Properties of partially matching text that are measurable for *word* comparable units.

sentences with words include one sentence with words of a training document and one sentence with words of a JTA workbook task statement. Eight properties of partially matching text are measurable with respect to these comparable units (Fig. 3).

To determine if a training document is aligned, the algorithms should be capable of assigning relative values to all identified matches. This capability is needed for the alignment reports to identify what documents are aligned with what task statements and what documents are not aligned.

1. *Ordering*, which means that a training document has sentences with words that match task statement words character for character and are also in the correct order
2. *Semantics and ordering*, which means that a training document has sentences with words that are semantically matching task statement words and are also in the correct order
3. *Part of speech morphology and ordering*, which means that a training document has sentences with words that have the same parts of speech as task statement words and are in the correct order
4. *Root morphology and ordering*, which means that a training document has sentences with words that have the same roots as task statement words and are in the correct order
5. *Orthography and ordering*, which means that a training document has sentences with words that are the same, but are spelled as acronyms, initialisms, or abbreviations, and are also in the correct order
6. *Semantics, part of speech morphology, and ordering*, which means that a training document has sentences with words that are semantically equivalent or related to task statement words, have the correct parts of speech, and are also in the correct order
7. *Root morphology, part of speech morphology, and ordering*, which means that a training document has sentences with words that have the same roots as task statement words, have the correct parts of speech, and are also in the correct order
8. *Orthography, part of speech morphology, and ordering*, which means that a training document has sentences with words that are the same as task statement words, but are spelled as acronyms, initialisms, or abbreviations; have the correct parts of speech; and are also in the correct order

Figure 3: Properties of partially matching text that are measurable for *sentence with words* comparable units.

A document could be considered aligned if it has at least one exact match or at least one sufficiently relevant partial match. Documents having one or more exact matches should be considered the most aligned. Determining if a partial match is sufficiently relevant requires a way of determining the relative value of each partially matching property.

Since all partially matching properties involve words, the relative values of each property depend in part on the relative importances of words. This can be defined as a constant importance value that is based on part of speech, since some parts of speech will always be more important than others. Applicable parts of speech are the ones that are in the FAA-supplied task statements: nouns, verbs, adjectives, articles, prepositions, and conjunctions. We list these parts of speech below in descending order of importance, where all parts of speech on the same line should have the same importance value:

- Nouns and verbs
- Adjectives
- Articles, prepositions, and conjunctions

Verbs and nouns are the most important because each defines what the Tech Ops personnel need to do. Adjectives are moderately important because they carry some meaning. Articles, prepositions, and conjunctions are considered the least important because they do not carry meaning.

Considering the relative importance of nouns and verbs, one issue is that a document should minimally have one noun *and* one verb that is exactly or partially matching. Additionally, because all task statements are full sentences, it is critical that an exactly/partially matching noun and an exactly/partially matching verb are in the same sentence; otherwise, the whole sentence could be irrelevant to a task statement. Thus, for a document with partially matching text to be aligned, it must exhibit the property of part of speech morphology for two words that are in the same sentence: a verb and a noun. This means that all documents having no such text cannot be considered aligned. This also means that the relative values of potentially aligned documents must be measured for comparable units of sentences with words.

Acronyms, initialisms, or abbreviations are the best kinds of partially matching verbs and nouns. Verbs and nouns that have the same root are the next best, followed by verb and nouns that are semantically equivalent. While the correct ordering of verbs and nouns is not critical to alignment, having them in the correct order could reflect a higher degree of relevance. These conditions also apply to adjectives, articles, prepositions, and conjunctions, but with a relatively lower importance value.

The most-aligned documents with partially matching text, all of which are considered aligned, are listed in descending order of alignment below. For comparable units of sentences with words, these documents will have

1. A verb and a noun that are exactly matching the respective verb and noun of a task statement and are also in the correct order (i.e., *part of speech morphology and ordering*)
2. An exactly matching verb or noun, where the other word has the correct part of speech and is the spelled-out equivalent of an acronym, initialism, or abbreviation (or vice-versa), while both words are also in the correct order (i.e., *orthography, part of speech morphology, and ordering*)
3. A verb and a noun that are both the spelled-out equivalents of an acronym, initialism, or abbreviation (or vice-versa) and are also in the correct order (i.e., *orthography, part of speech morphology, and ordering*)
4. An exactly matching verb or noun, where the other word has the correct root and the correct part of speech, while both roots are in the correct order (i.e., *root morphology, part of speech morphology, and ordering*)
5. A verb and a noun that have the correct roots and are in the correct order (i.e., *root morphology, part of speech morphology, and ordering*)
6. An exactly matching verb or noun, where the other word is semantically equivalent and has the correct part of speech, while both words are in the correct order (i.e., *semantics, part of speech morphology, and ordering*)
7. A verb and a noun that are both semantically equivalent and are in the correct order (i.e., *semantics, part of speech morphology, and ordering*)
8. A verb and a noun that are exactly matching the respective verb and noun of a task statement (i.e., *part of speech morphology*)

9. An exactly matching verb or noun, where the other word has the correct part of speech and is the spelled-out equivalent of an acronym, initialism, or abbreviation (or vice-versa) (i.e., *orthography and part of speech morphology*)
10. A verb and a noun that are both the spelled-out equivalents of an acronym, initialism, or abbreviation (or vice-versa) (i.e., *orthography and part of speech morphology*)
11. An exactly matching verb or noun, where the other word has the correct root and the correct part of speech (i.e., *root morphology and part of speech morphology*)
12. A verb and a noun that have the correct roots (i.e., *root morphology and part of speech morphology*)
13. An exactly matching verb or noun, where the other word is semantically equivalent and has the correct part of speech (i.e., *semantics and part of speech morphology*)
14. A verb and a noun that are both semantically equivalent (i.e., *semantics and part of speech morphology*)

The most aligned among each respective set of documents will also having matching adjectives, articles, prepositions, and conjunctions, where the properties of these matching words are valued in the same way (i.e., exact matches, followed by orthography matches, followed by root matches, followed by semantics matches). The ordering of relatively less important words is not critical to alignment, but having them in the correct order could reflect a higher degree of relevance.

Pre-Processing Text Data. As mentioned, text pre-processing algorithms are needed to support text comparison. As shown above, text comparison algorithms identify matches between comparable units: sentences, words, and sentences with words. Thus, the pre-processing algorithms should be capable of:

- Breaking up training document text data into sentences
- Breaking up training document sentences into words
- Breaking up task statement text data into words
- Breaking up training document text data and task statement text data into sentences with words

Additional pre-processing algorithms could modify the comparable units in a way that supports text comparison algorithms. One potentially useful modification involves identifying relevant attributes of comparable text units. For example,

- Conceptually equivalent and conceptually related words can be identified in order to compare semantics
- Parts-of-speech can be identified in order to compare part of speech morphologies
- Root words can be identified in order to compare root morphologies
- Acronyms, initialisms, and abbreviations can be identified in order to compare orthographies

The pre-processing algorithms could leverage data that aid in identifying the attributes above. For example, orthography partial matches can be identified using a list of acronyms, initialisms, and abbreviations with mappings to their spelled-out forms. A dictionary can be utilized to identify root words and parts of speech. A thesaurus can be utilized to identify semantically equivalent and related words.

Another potentially useful modification involves removing or replacing text. For example, common words that are unimportant can be removed (e.g. "of," "and," "the"), punctuation characters can be removed (e.g. hyphens and slashes), and uppercase letters can be converted to lowercase. These kinds of modifications could be applied toward the identification of all applicable matches. Other modifications could be useful for identifying specific kinds of matches. For example, acronyms, initialisms, and abbreviations can be replaced with their spelled-out forms (of vice-versa); and words can be replaced with their root forms.

REPORTING CAPABILITIES. Alignment reports should identify what documents are aligned/not aligned (i.e., their filenames) and where the files are located in the corpus (i.e., directory and subdirectories). These outputs should be provided for all training document inputs.

The analyst could be interested in identifying what documents align with what task statements, or vice-versa. Thus, the alignment report should enable the analyst to compare a training document and a task statement. When the analyst focuses on a training document, the alignment report should show all task statements for which it is aligned. When the analyst focuses on a task statement, the alignment report should show what documents are aligned. For either focus, the alignment report should identify the results in descending order of relevance. If there are any exact matches, then the alignment report should differentiate them from partial matches (if any).

For the identified matches, the analyst could be interested in descriptive statistics about them. Statistics of interest could apply to all documents and task statements of interest, where one or more documents could have sufficiently relevant matching text and one or more task statements could be addressed. Thus, alignment reports should identify how many documents have sufficiently relevant matches and how many task statements are addressed.

Statistics of interest could also apply to each individual document and task statement of interest. For example, one document could have

- One sufficiently relevant match for one task statement
- One sufficiently relevant match for many task statements
- Many sufficiently relevant matches for one task statement
- Many sufficiently relevant matches for many task statements

Similarly, one task statement could have

- One sufficiently relevant match found in one document
- One sufficiently relevant match found in many documents
- Many sufficiently relevant matches found in one document
- Many sufficiently relevant matches found in many documents

Thus, alignment reports should incorporate the descriptive statistics listed above for each document and task statement of interest.

Additionally, the analyst might want to save an alignment report for later use. Thus, the approach should enable the analyst to export alignment reports.

The operational concept described above could be characterized as a kind of information retrieval (IR) application. IR is a discipline that addresses the need for automated data searching technologies (Frakes and Baeza-Yates, 1992). *Text search* is one kind of input/output IR technology that could be leveraged within this research (see Zobel and Moffat (2006) for a review).

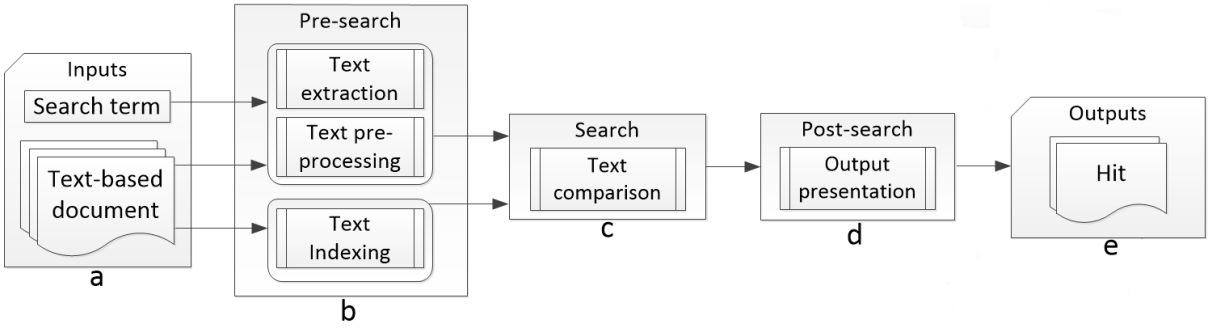

Figure 4: Inputs, processes, and outputs of a text search approach to information retrieval. Letters are added for reference in body text of this document.

ELEMENTS OF A TEXT SEARCH APPROACH. Fig. 4 shows the inputs, outputs, and processes that are common in a text search approach. Inputs (Fig. 4a) are

1. One set of documents through which the analyst wants to search (e.g. text-based documents that are stored on a computer)
2. One search term representing information for which the analyst wants to search (e.g. a keyword, a full sentence)

Outputs are information about the subset of documents having text that is relevant to the search term (called *hits*) (Fig. 4d). Generating outputs of interest involves a series of processes tht can be characterized as pre-search (Fig. 4b), search (Fig. 4c), and post-search (Fig. 4e).

Pre-Search Processes. Three kinds of pre-search processes facilitate searches: text extraction, which is the process of getting text data from the inputs into a form that can be read by other processes; text pre-processing, which is the process of modifying the text data to enable the search processes; and text indexing, which is the process of organizing the text data in ways that speed up a search process.

The first step in any text search approach is text extraction (Konchady, 2008). Text input is used to define the goal of the search. The source text (i.e., the target text to search) can be extracted once and then searched repeatedly. The source text only needs to be extracted again when it is updated. For example, to support Web-based search engines, "crawler" tools are constantly scanning the Web for newly created Hypertext Markup Language (HTML) Web pages (Berners-Lee, 1991). In other text search applications, the analyst manually updates the source of text-based documents by pointing to a file or directory containing them. Text extraction software can automatically render these data in a digital, machine-readable format.

Text pre-processing involves operating on the extracted text in ways that support different kinds of searches. This is typically accomplished using three kinds of highly

automated algorithms: tokenizers, filters, and annotators (Konchady, 2008). Tokenizers break up the extracted text into units (called *tokens*) that can be compared against each other, such as words in a search term and words in a document. Filters remove parts of the tokens that are not applicable to search, such as unimportant words (e.g. articles and prepositions). Annotators assign the tokens attributes that are relevant to a search, such as part of speech attributes to word tokens that are relevant if the analyst is interested in searching for documents containing a particular noun.

Text indexing involves organizing the tokens of documents into a specialized text-search data structure called an *index*, which can improve time efficiency of the search processes. In an index, all tokens from all text-based documents are organized such that each unique token is associated with the subset of documents containing it (where "unique" means that the index contains no duplicate instances of the same token) (Zobel and Moffat, 2006). This allows for search term tokens to be compared against all documents of interest without analyzing the same token twice, even if it appears in many documents.

Search Processes. Search processes utilize input/output text comparison algorithms that automate the process of identifying hits. Inputs are one token of an index and one token of a search term. Outputs are numerical scores that quantify the relevance of each document with respect to the search term (see for example Konchady, 2008, p. 153). The usefulness of a particular text comparison algorithm mainly depends on how the analyst defines relevance.

In many text search applications, such as Web-based search engines, relevance means that a document has one or more tokens that are exactly matching one or more search-term tokens (Frakes and Baeza-Yates, 1992). For such a definition, applicable text comparison algorithms are statistical, meaning they consider the number of tokens in a document, the number of tokens in a search term, and the number of tokens that are matching (Zobel and Moffat, 2006). A document having many instances of an exactly matching token relative to non-matching tokens could be considered relevant (Robertson and Jones, 1976).

Other text comparison algorithms are useful if the analyst defines relevance with respect to the properties of individual matching tokens. Extant algorithms can be characterized as *lexical*, meaning they compare syntax-related properties; *semantical*, meaning they compare meaning-related properties; and *hybrid*, meaning they compare both syntax- and meaning-related properties (Gomaa and Fahmy, 2013).

A lexical algorithm is useful if relevance means that tokens of a search term and tokens of a document have similar syntaxes (e.g. ordering of words in a sentence). A semantical algorithm is useful if relevance means that tokens of a search term and tokens of a document have similar meanings (e.g. words that are different, but capture the same concept). A hybrid algorithm is useful if relevance means that tokens of a search term and tokens of a document have similar syntaxes *and* similar meanings (e.g. different words that capture the same concept and are in the correct order).

Post-Search Processes. Post-search processes are for presenting outputs that render desired information about the hits, such as names, file types, locations, and text therein that is relevant to the search term. If there are multiple hits, then the post-search processes can also express relationships among them.

Rendering information about each hit commonly involves extracting applicable metadata and text data. Applicable metadata often come from a file system on which the

target document is stored (i.e., directory, subdirectory, filename, and file type). Applicable text data come from the target document (i.e., a subset of tokens that are relevant to the search term).

Rendering information about multiple hits commonly involves expressing one of two relationships: relevance with respect to the search term or relatedness of the hits with respect to each other (Shneiderman et al., 1997). Expressing the relevance of multiple hits with respect to the search term involves using numerical scores quantifying relevance for each document, which are outputs of one or more text comparison algorithms. These scores are used to render a list of hits that places them in descending order of relevance (Page et al., 1999). Such a presentation often separates a large number of hits by distributing them across multiple screens to which the analyst can navigate (e.g. sequentially ordered pages, each of which shows ten hits at a time) (Shneiderman et al., 1997).

Expressing the relatedness of multiple hits with respect to each other involves using descriptive statistics, such as how many of the same tokens appear in a particular subset of hits. These relationships are often computed using *latent semantic analysis* (LSA) algorithms that are similar to the statistical text comparison algorithms used to identify hits (Landauer, 2006). However, instead of comparing the tokens of a search term to the tokens of an index, they compare the tokens of an index with each other. This allows for the outputs to organize subsets of documents into categories (referred to as *clusters*) (Zamir and Etzioni, 1999).

SCOPE OF THE LITERATURE REVIEW. IR researchers have developed tools, techniques, and algorithms supporting a variety of text-search inputs, outputs, and processes. In this research, we are interested in identifying recommendations for a text search approach to alignment. Thus, the scope of our literature review is meant to support this objective.

With respect to the operational concept above, we focus on a subset of the relevant text search literature about

- Text extraction, which is needed to get text from training documents and task statements from JTA workbooks identified
- Text pre-processing, which is needed to break up and modify the extracted text
- Text indexing, which is needed to facilitate searches with respect to text-based documents of interest
- Text comparison, which is needed to determine what text-based documents have exactly/partially matching text that is sufficiently relevant to one or more task statements
- Output presentation, which is needed to inform the analyst about what text-based documents are hits with respect to one or more task statements, what text-based documents are *not* hits with respect to all task statements, and descriptive statistics about the hits

TEXT EXTRACTION. Search-term text is commonly extracted from a text-entry field in which the analyst enters strings of characters (e.g. a search bar in which the analyst can click and type). Document text is commonly extracted from formatted text files or Web pages.

In this research, search terms come from full-sentence job task statements, while document text comes from files having the formats shown in Table 1. Thus, we are interested in tools that can extract text from files having these formats.

There are currently open-source text extraction tools supporting the file formats of interest. Examples include Apache PDFBox (Apache Software Foundation, 2017b), which can extract text from files in PDF formats; Apache POI, which can extract text from files in Microsoft Office formats (DOC, PPT, XLS, DOCX, PPTX, and XLSX); and Apache Tika (Apache Software Foundation, 2017d), which can extract text from files in either kind of format (i.e., Adobe and Microsot Office).

When applying text extraction tools toward common file formats, one assumption is that the analyst's computer can interpret the document's contents as text data. This usually means that data are digitally encoded text characters, such as alphabetical letters, Arabic numbers, whitespace, and punctuation. If the document contains embedded images, then the tools listed above will ignore them and only extract digitally encoded text characters.

One potential issue is that some text-based files, such as PDFs, can store images of characters instead of their digital encodings; thus, such a file will not be usable in a text search approach. This issue sometimes arises when sheets of paper with characters printed on them are laser scanned and loaded onto a computer as a PDF file. For example, in the FAA Tech Ops course curriculum, one such document is *40289_TASA.pdf*, which is on the COE for TTHP SharePoint site in the directory *Technician Course Materials/40289/Course Documentation/Analysis Docs/TASA/*.

To render these kinds of files usable in a text search approach, researchers have developed optical character recognition (OCR) software that enables a computer to automatically convert images of characters to their digital encodings. Two open source examples are Tesseract (Google, 2017) and OCRopus (Breuel, 2017).

TEXT PRE-PROCESSING. As mentioned, text search approaches incorporate text pre-processing algorithms in order to break up and modify extracted text data. Algorithms that break up text data are called *tokenizers*, where the input is one contiguous string of characters and the output is a set of text-string units called *tokens*. Algorithms that modify tokens are called *filters* and *annotators*. Filters modify tokens by removing or replacing words and characters, and annotators modify tokens by assigning relevant attributes, such as parts of speech to words.

Tokenizers. Tokenizers utilized in text search applications range in granularity, from *keyword tokenizers* at the lowest granularity, which interpret one token as the continuous string of characters in an entire document, to *character tokenizers* at the highest granularity, which interpret one token as a single character (Konchady, 2008). In this research, we are interested in breaking up text into sentences, words, and sentences with words. Thus, the applicable tokens are words and sentences, which means that a *word tokenizer* and a *sentence tokenizer* are needed.

Extant algorithms usually accomplish word and sentence tokenization using regular expression (RegEx) syntax (Thompson, 1968) that defines the pattern of characters in an English word or sentence. The RegEx patterns used for word tokenizers often define words as contiguous character strings that begin and end with whitespace, such as the Apache Lucene *whitespace tokenizer* (Apache Software Foundation, 2016). Some techniques treat

all punctuation characters as individual tokens (see for example Konchady (2008, p. 2)), while others treat punctuation characters as parts of words, such as hyphens in compound words, periods in decimal numbers, and apostrophes in possessive nouns (see for example Konchady (2008, pp. 36–37)).

Sentence tokenizers use RegEx patterns defining start, middle, and end character sequences of English-language sentences. For example, the sentence tokenizer in Cunningham (2002) uses two kinds of *split* patterns defining how common English-language sentences end:

1. Internal splits, such as a period that is at the end of one sentence
2. External splits, such as a line break that comes between two sentences

RegEx-based sentence tokenizers can be effective, but they do not guarantee that the identified sentence tokens will be grammatically complete (where a grammatically complete sentence should contain at least one noun and one verb). For example, section titles of text-based document that begin with a capital letter and end with a line break could be sentence fragments, but RegEx patterns are incapable of distinguishing sentence fragments from grammatically complete sentences.

To address this issue, some sentence tokenizers use RegEx patterns in conjunction with machine learning algorithms to identify grammatically complete sentences. For example, the sentence tokenizer in Manning et al. (2014) uses a probabilistic *sentence model* (SM) that is derived from a set of known sentences. For a text-based document having an unknown number of sentences, rule-based RegEx patterns are used first; then the sentence model is applied to determine what tokens are likely to be grammatically complete. These kinds of tokenizers could be useful in this research because we are interested in comparing full-sentence job task statements with full sentences of text-based documents, where both contain at least one noun and at least one verb.

Filters. An mentioned, filters can automatically enhance the utility of tokens by removing or replacing words and characters. Filters that remove or replace characters typically address letter case and punctuation. These kinds of filters could be useful in this research for addressing situations in which relevant text of a document does not match text of a job task statement exactly due to punctuation and letter case.

Letter case and punctuation are commonly addressed in a variety of text search applications. For example, in Apache Lucene (Apache Software Foundation, 2016), the *lowercase filter* converts all uppercase letters of a token to lowercase letters, while the *standard filter* automatically removes all punctuation characters.

Commonly utilized filters remove unimportant words (called stop words), such as articles, prepositions, and conjunctions, and replace stemmed words with their roots. These kinds of filters could be useful in this research for addressing situations in which relevant text in a document does not match the text of a job task statement exactly due to a different morphology of the same root word or non-matching articles, prepositions, and conjunctions.

Similarly to letter case and punctuation, stop word and stem filtering are commonly applied in a variety of text search applications. Thus, widely utilized text search software, such as Apache Lucene (Apache Software Foundation, 2016) and Alias-i LingPipe (Alias-i, 2011), support these types of filtering processes.

This research needs to address matching text to appropriate acronyms, initials and abbreviations. Researchers have developed filtering algorithms that can accomplish this by automatically finding and replacing acronyms, abbreviations, and initialisms with their spelled-out forms (referred to as *expansions*) (Schwartz and Hearst, 2002; Taghva and Vyas, 2011; Park and Byrd, 2001). Extant algorithms range in applicability and accuracy.

Schwartz and Hearst (2002) develop a set of RegEx patterns for identifying acronyms appearing in parentheses immediately after their spelled-out forms. The algorithm builds a dictionary of acronyms-expansion pairs and then uses the dictionary to execute automated filtering with up to 96% accuracy. However, the accuracy of this algorithm was only evaluated for corpora of biomedical texts.

Taghva and Vyas (2011) use RegEx patterns to define a dictionary of acronyms, initialisms, and abbreviations having *candidate* expansions that come from text-based documents. An accompanying Hidden Markov Model (HMM) (Eddy, 1996) computes the probability of a candidate expansion having the corresponding acronym, initialism, or abbreviations. This approach achieves up to 92% accuracy in a general corpus of documents gathered from the Web.

Park and Byrd (2001) employ similar RegEx patterns, but combine them with machine learning algorithms that can automatically find acronyms, initialisms and abbreviations and replace them with expansions. This approach achieves up to 97% accuracy in a general corpus of documents gathered from the web.

Annotators. Annotators can automatically assign a token one or more attributes that are relevant to the search. Common algorithms start with a set of manually annotated word tokens that is used to parameterize the algorithm with word-annotation probability estimates (see for example Brill (1992)). Subsequent runs of the algorithm can identify different word-annotation pairs for new documents while also improving accuracy of the probability estimate.

These kinds of algorithms can support many different annotation tasks, but they are commonly used to assign parts of speech to word tokens. Such an algorithm would be useful in this research because we are interested in sentences of a document that contain both a verb and a noun.

Currently, there are two kinds of algorithms that are commonly utilized to annotate word tokens with part of speech attributes: Hidden Markov Model (HMM) and Maximum Entropy (ME) (Kupiec, 1992; Toutanova and Manning, 2000). HMM algorithms operate on sentence tokens having word tokens (i.e., sentences with words) by identifying part-of-speech sequences that are likely in an English sentence (e.g. a noun followed by verb). Maximum Entropy (ME) algorithms determine the probability of a word token having a particular part of speech, even if they are not in complete sentences (Toutanova and Manning, 2000). This is accomplished by determining the context of a word token based on the surrounding sequences of word tokens (i.e, those that come before and after).

HMM and ME algorithms proven up to 97% accurate for annotating parts of speech in a variety of English-language corpora (Tian and Lo, 2015). Apache Open NLP (Apache Software Foundation, 2017a) utilizes the HMM-based algorithm, while Stanford Core NLP (Manning et al., 2014) utilizes the ME-based algorithm.

Indexers. As mentioned, indexers organize document tokens within indexes that are constructed to support rapid retrieval of tokens appearing in text-based documents (Zobel

and Moffat, 2006). The tokens that come from all text-based documents are commonly organized as an alphabetically ordered list, while each token is associated with a subset of documents in which the token appears (Fig. 5).

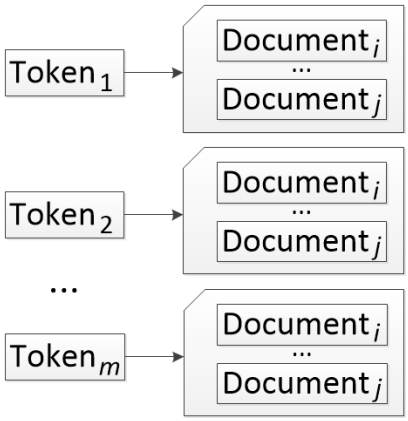

Figure 5: Graphical representation of an index. Subscripts 1–m identify unique tokens of n text-based documents. Subscripts i–j identify unique documents in which each unique token appears, where $1 \leq i < j \leq n$.

Text search approaches commonly utilize an index to associate relevant attributes with each token (Fig. 6b) and document (Fig. 6b). Relevant attributes could be text or numbers. For tokens, text attributes often identify annotations (e.g. part of speech for word tokens). For documents, text attributes often identify metadata that are needed in the output presentation (e.g. filename). These kinds of attributes could be useful in an index for this research, since we are interested in analyzing parts of speech and presenting applicable metadata for the documents of interest.

For tokens and documents alike, number attributes often identify parameters that enable search processes. For example, in a *term frequency-inverse document frequency* (TF-IDF) index (Ramos et al., 2003), each token is assigned a number that identifies how many times it appears in the full set of text-based documents being searched. In the list of documents associated with each token, each document is assigned a number that identifies how many times the token appears therein (Fig. 6c). Such an index is often used to support statistical text comparison algorithms that consider the relevant parameters. Such an index could be useful in this research for presenting descriptive statistics about documents that are aligned, such as how many documents contain text that exactly matches a job task statement and how many matches appear in each document.

Other common number attributes identify the positions of a token in each document with respect to all other tokens therein. (Zobel and Moffat, 2006, p. 11) refer to an index that incorporates these attributes as *word-level*. A word-level index could be useful in this research for comparing the ordering of words in a task statement with the ordering of words in a document.

TEXT COMPARISON. Text comparison algorithms can compare a search-term token against tokens in an index, both of which are a word, sentence, or sentence with words. As mentioned, these algorithms can be characterized as *statistical*, *lexical*, *semantical*, or *hybrid*. Statistical algorithms consider the number of tokens in a document that are exactly matching the tokens of a search term. To address the properties of individual

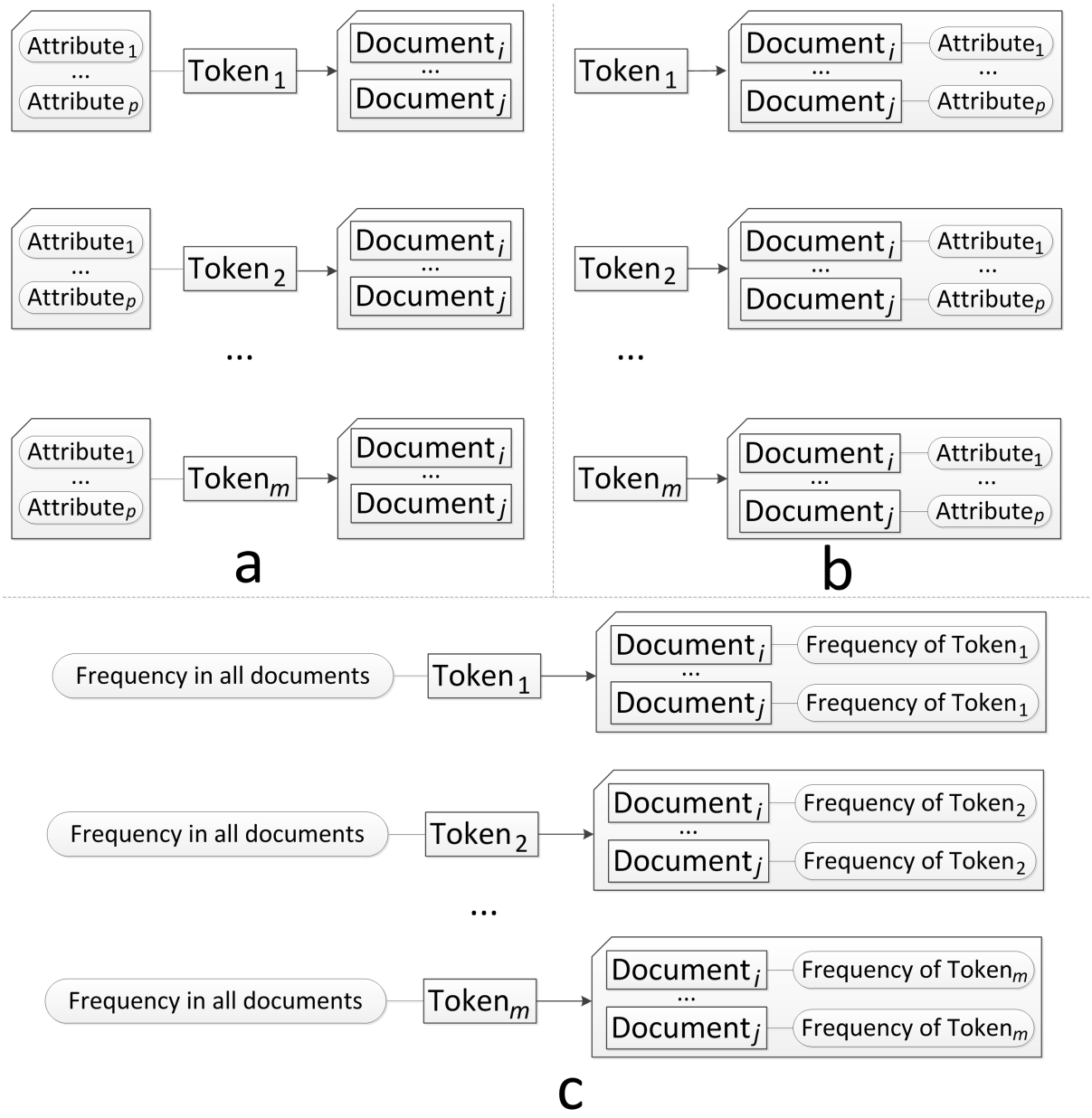

Figure 6: Graphical representation of an index, where there are n text-based documents of interest. Subscripts 1–m identify unique tokens of all text-based documents. Subscripts i–j identify unique documents in which each unique token appears, where $1 \leq i < j \leq n$.

tokens, lexical algorithms compare syntax, semantical algorithms compare meaning, and hybrid algorithms compare both syntax and meaning (Gomaa and Fahmy, 2013).

In this research, we are interested in identifying documents that are relevant to one or more job task statements based on properties of individual tokens (i.e., whether they are exactly matching or partially matching and sufficiently relevant). Thus, lexical, semantical, and hybrid algorithms could be useful.

Lexical Algorithms. Levenshtein distance (Levenshtein, 1966) is a common lexical algorithm that can operate on words, sentences, or sentences with words. When operating on words or sentences, the Levenshtein distance of a document token is the number of

characters that need to be inserted, deleted, or replaced for the token to match a search term token exactly. When operating on sentences with words, the Levenshtein distance of a document token is the number of words that need to be inserted, deleted, or replaced for it to match a search term exactly. In either case, document tokens having a lower Levenshtein distance generally reflect a higher relevance (i.e., tokens that are exactly matching a search term have a Levenshtein distance of zero). Open source software, such as Apache Lucene (Apache Software Foundation, 2016), come with built-in Levenshtein distance algorithms for comparing words, sentences, and sentences with words.

Word-overlap lexical algorithms, such as the Jaccard algorithm (as cited in Gomaa and Fahmy, 2013) and the Simpson algorithm (Simpson, 1960), support lexical comparisons between sentences with words, although the ordering of words is not considered. In the Jaccard algorithm, the *Jaccard similarity score* of a document token is equal to the number of words that match the search term exactly (excluding repeats) divided by the quantity of non-matching words plus the sum of words in both (excluding repeats). Tokens with a higher Jaccard similarity score generally reflect a higher degree of relevance. The Simpson algorithm (Simpson, 1960) is similar, although it involves dividing the number of words in a document token that are exactly matching words of a search term token by the number of words in whichever token has fewer words (i.e., instead of the quantity of non-matching words plus the sum of words in both token). For a given document token, the resulting measure is its *Overlap coefficient*. Document tokens with a higher Overlap coefficient generally reflect a higher degree of relevance. Both the Jaccard and Simpson algorithms are incorporated within open source software tools, such as Apache Open NLP (Apache Software Foundation, 2017a), Stanford Core NLP (Manning et al., 2014), and Natural Language Toolkit (NLTK) (Natural Language Toolkit Project, 2017).

Word-order lexical algorithms compare the ordering of words in a document token with the ordering of words in a search term token, where both tokens are sentences with words. Islam and Inkpen (2008) develop an algorithm in which words of a search-term token and words of a document token are assigned indexes. Two vectors list the indexes of exactly matching words in order of appearance. The *word-order similarity* score of a document token is computed as shown in Appendix A (Equation 1), where document tokens with a higher word-order similarity score generally reflect a higher degree of relevance.

Semantical Algorithms. Semantical algorithms can measure either the *similarity* or the *relatedness* of two word tokens (one from a document and one from a search term). Thus, the algorithms operate on words. Algorithms for computing either kind of measure use the WordNet (Miller, 1995) database of English words and their *is-a* taxonomies. In WordNet, a word's is-a taxonomy is the hierarchical structure of its synonyms (i.e., words of a similar meaning and specificity), hyponyms (i.e., words of a similar meaning but lower specificity), and hypernyms (i.e., words of a similar meaning and higher specificity). Each set of synonyms, hypernyms, and hyponyms is referred to as a *node*.

Common similarity algorithms consider paths between two words that are traversed over nodes. For example, in the Leacock-Chodorow algorithm (Leacock and Chodorow, 1998), the semantical similarity of two words is the shortest path traversed over common hypernyms/hyponyms nodes divided the is-a taxonomy depth of whichever word has a deeper is-a taxonomy (i.e., the search term word or the document word). The *Leacock-Chodorow similarity* score of a document token is the negative log of this value, which is between zero and one.

Relatedness algorithms compare two words as a function of their *relation sets*. In WordNet, a word's relation set depends on its part of speech. For nouns, it includes synonyms, hypernyms, hyponyms, and meronyms (i.e., part-whole words, such as "key" as a meronym of "keyboard"). For adjectives, a relation set includes synonyms, hypernyms, hyponyms, and antonyms (i.e., words with the opposite meaning, such as "slowly" as an antonym of "quickly"). For verbs, a relation set includes synonyms, hyponyms, and troponyms (i.e., the verb analog of a noun hypernym, such as "count" as a troponym of "measure"). The Adapted Lesk algorithm (Banerjee and Pedersen, 2002) uses these relation sets to compute the relatedness of two words. It considers

1. The relation sets of the words being compared
2. The definitions of words in the relation sets
3. The number of words in the definitions that are overlapping and in the same order (referred to as *overlaps*)

The *Adapted Lesk relatedness* score of a document token is equal to the sum of square root overlaps, where document tokens with a higher Adapted Lesk related score generally reflect a higher degree of relevance.

Some relatedness measures can be computed without using WordNet. For example, the Pointwise Mutual Information (PMI) algorithm (Turney, 2001) measures semantic relatedness of two words as a function of two probabilities: the joint probability of both words appearing within a "window" of n words divided by the independent probability of both words occurring anywhere in the set of documents under consideration. A document token with a higher *PMI relatedness* score generally reflects a higher degree of relevance.

Open source software tools currently support these measures. Examples include the Natural Language Toolkit (NLTK) project (Natural Language Toolkit Project, 2017), which has a built-in method for computing PMI, and WordNet Similarity for Java (WS4J) (Shima, 2017), which implements the Leacock-Chodorow and Adapted Lesk algorithms.

Hybrid Algorithms. Hybrid algorithms operate on sentences with words to determine the similarity of a document token with respect to a search-term token. They can be implemented using the lexical and semantical algorithms described above. This is because hybrid algorithms in the text search literature leverage a combination of measures that come from the aforementioned algorithms.

For example, Li et al. (2004) utilize a combination of three measures to determine overall similarity between two sentences: one of word overlap, one of semantical similarity, and one of word order. The *Li et al. sentence similarity* score of a document token with respect to a search-term token is defined in Equation 2 (Appendix A). Here, sentence similarity is a function of thee measures: one of word order, one of word overlap, and one of semantical similarity. The word-order measure is computed in a way that is similar to Equation 1, although their formulation requires the tokens being compared to have the same number of words. The word-overlap measure uses a formulation that is similar to an Overlap coefficient. The semantical similarity measure uses the Leacock-Chodorow algorithm (Leacock and Chodorow, 1998).

Islam and Inkpen (2008) develop a hybrid algorithm that can compare tokens having different numbers of words. The *Islam and Inkpen sentence similarity* score of a document token with respect to a search-term token is defined in Equation 3 (Appendix A). This algorithm applies four measures: one of word overlap, one of word order, one of semantical

relatedness, and one of lexical similarity. The word-overlap measure is the number of exactly matching words. The word-order measure is computed using Equation 1. The lexical similarity measure is similar to the Levenshtein distance algorithm (Levenshtein, 1966), while semantical measures are computed for each word using the PMI relatedness algorithm (Turney, 2001).

OUTPUT PRESENTATION. This literature review focuses on what information is commonly presented in the outputs of a text search approach, such as how many hits were found, and for each hit, name, file type (e.g PDF), location (e.g. Web address), and contents (e.g. first few sentences). If there are multiple hits, then they are commonly presented as lists of hits that are distributed across pages, where one page shows a constrained set of hits (e.g. ten hits to a page). The AN003 report focuses on usability-related considerations that are identified in the relevant HCI literature.

Information about Each Hit. In Web-based search engines of the 1990s, an output presentation would commonly show the first few sentences of a hit directly under its name; however, researchers have found that such a presentation has limited usefulness (Tombros and Sanderson, 1998). This is because the few sentences of a hit are often irrelevant to an end user's search term. Thus, more recent text search applications employ improved presentations.

A commonly utilized presentation is a *keyword in context* (KWIC) extraction (Tombros and Sanderson, 1998). A KWIC extraction usually shows one or more snippets of text in a hit, including the relevant tokens and the surrounding text (relevant tokens are emphasized and surrounding text is not) (Shneiderman et al., 1997). Such a presentation usually varies in length of the snippet and how much surrounding text is included for each relevant token. Researchers have found that users generally prefer a presentation of relevant tokens and surrounding text that forms a complete sentence, while the desired length of a snippet generally increases with length of the search term (Kaisser et al., 2008).

Information about Multiple Hits. As mentioned, a listing of multiple hits commonly presents them in descending order of relevance. The relevance of multiple hits with respect to each other could be determined using one or more text comparison algorithms described earlier.

An alternative representation of multiple hits presents them in subsets of hits that are similar to each other (see for example Zamir and Etzioni (1999)). Such a representation commonly leverages statistical text comparison algorithms and a TF-IDF index to identify what hits have similar word distributions (see for example Hearst (1995)). Each subset of hits that have similar word distributions is called a cluster. Clusters could be useful in the outputs for this research as a way of representing what hits address the same task statements (i.e., to supplement the descriptive statistics about each hit).

RECOMMENDATIONS FOR A TEXT SEARCH APPROACH TO ALIGNMENT

In this section we identify recommendations for a text search approach to alignment, including an architecture for pre-search, search, and post-search functions that are executed by the analyst or by the system. The first subsection below describes the recommended architecture. Lower-level sections describe functions, inputs, outputs. The second subsection maps each identified function to an applicable tool or algorithm described in the text

search literature review. All aspects of the recommended architecture correspond to the *basic cross-reference search* described in the AN003 report.

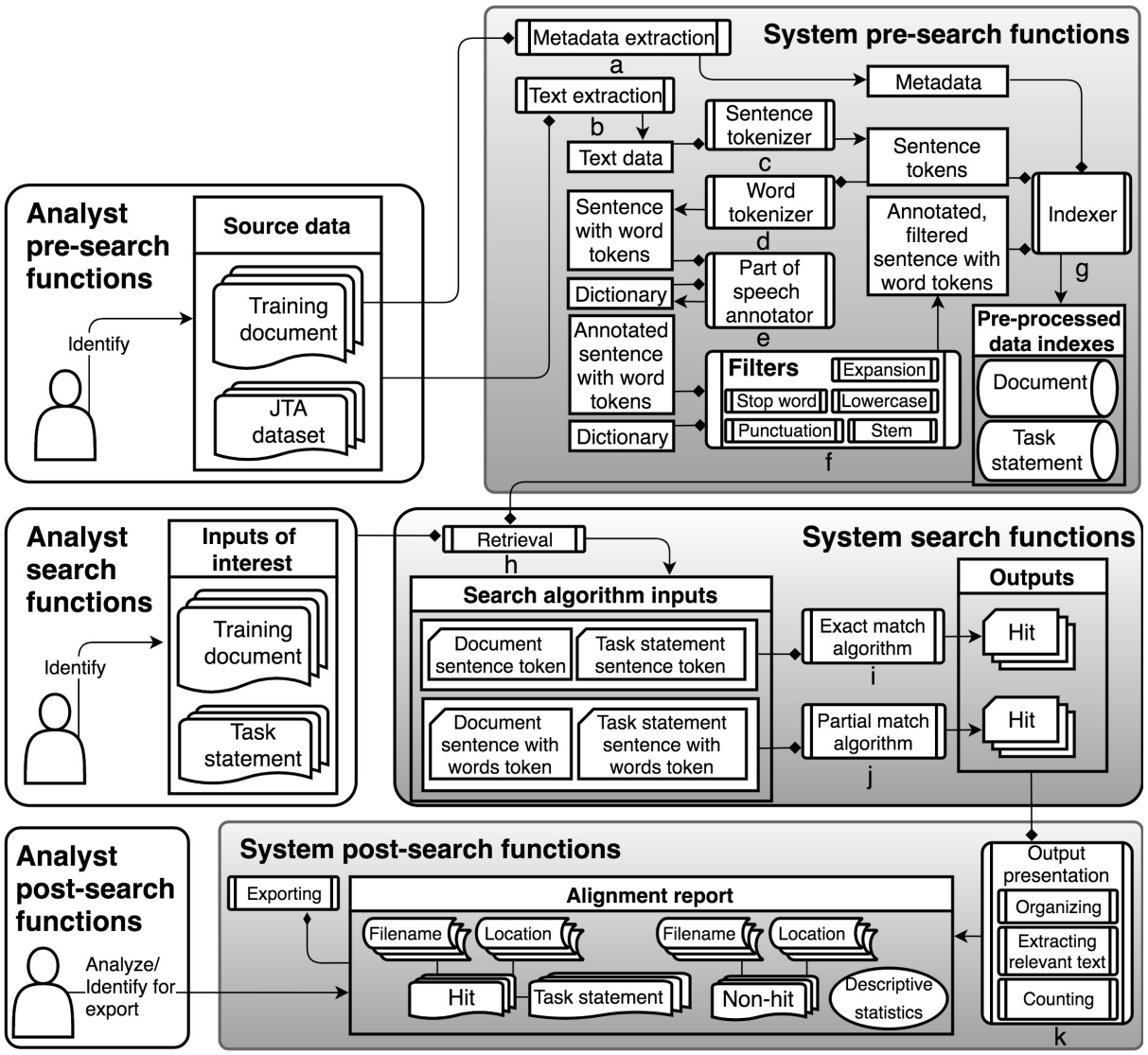

Figure 7: Recommended architecture for a text search approach to alignment. Rounded-edge rectangles with boldface headings identify functions performed by the analyst and by the system. Rounded-edge rectangles with interior vertical lines pre-search, search, and post-search functions. Lines with outgoing diamonds identify function inputs. Lines with outgoing arrows identify function outputs. Square-edge rectangles without vertical lines are inputs/outputs of the system's pre-search functions. Square-edge rectangles with a flat upper left-hand corner are inputs/outputs of search functions. Rectangles with a curved bottom edge are pre-search inputs (i.e., training documents, JTA datasets), search inputs (i.e., training documents and task statements of interest), and post-search outputs (i.e., hits and aligned task statements). Cylinders are pre-processed data. Rectangles with a concave right edge are hit metadata presented in the alignment report. Letters are added for reference in text of this document.

RECOMMENDED ARCHITECTURE. Fig. 7 shows a recommended architecture for a text search approach to alignment. Such an architecture incorporates pre-search, search,

and post-search functions that are executed by the analyst or by the system. Below we describe the functions and identify their mappings to elements of the operational concept.

Analyst Pre-Search Functions. The analyst's pre-search functions correspond to data selection capabilities of the operational concept. They include identifying source data (i.e., training documents and JTA datasets). The purpose of these functions is to identify data that are inputs to the system's pre-search functions.

System Pre-Search Functions. The system's pre-search functions correspond to data extraction and text-preprocessing capabilities of the operational concept. Their purpose is to operate on the identified source data in order to facilitate time-efficient search processes. They include

- Extracting metadata from the identified training documents (Fig. 7a)
- Extracting text from the identified training documents and JTA datasets (Fig. 7b)
- Breaking up the extracted text into sentence tokens (Fig. 7c)
- Breaking up the sentence tokens into sentence with word tokens (Fig. 7d)
- Annotating words of the sentence with word tokens by assigning part-of-speech attributes (Fig. 7e), where a dictionary input supports the tool or algorithm that executes the annotation function
- Filtering the annotated sentence with word tokens (Fig. 7f), where a dictionary input supports one or more tools/algorithms that execute the following filtering functions: stop word (i.e., removing unimportant words), expansion (i.e., converting acronyms, abbreviations, and initialisms to their spelled out forms, or vice-versa), lowercase (ie., converting all uppercase letters to lowercase), punctuation (i.e, removing all punctuation characters), and stem (i.e., converting all stemmed words to their root forms)
- Indexing the training document tokens (Fig. 7g) and storing the task statement tokens

This filtering functions can execute individually or in conjunction with all other filtering functions. To address all applicable properties of partially matching text (Tables 2 and 3), all combinations of filters will be applied. The indexer will ensure that the document and task statement indexes store each set of tokens that is filtered in a different way.

Analyst Search Functions. Similarly to the analyst's pre-search functions, the analyst's search functions correspond to data selection capabilities of the operational concept. They include identifying the training documents and JTA task statements of interest. The purpose of these functions is to identify inputs for the system's search functions.

System Search Functions. The system's search functions correspond to data retrieval and text comparison capabilities of the operational concept. Their purpose is to identify what training documents of interest are aligned/not aligned with task statements of interest. These functions include

- Retrieving pre-processed data for the analyst's inputs of interest (Fig. 7h)

- Executing exact match text comparison algorithms (Fig. 7i) on one sentence token of a target training document and one sentence token of a target JTA task statement.
- Executing partial match text comparison algorithms (Fig. 7j) on one sentence with words token of a target training document and one sentence with words token of a target JTA task statement

The relative value of each document of interest will come from the score computed by an applicable text comparison algorithm. All documents identified as hits will satisfy a formal definition of alignment (e.g. the hit contains at least one sentence token having a Levenshtein distance of zero).

Analyst Post-Search Functions. The analyst's post-search functions correspond to reporting capabilities of the operational concept. The purpose of these functions is gleaning information about alignment with respect to the inputs of interest. They include analyzing alignment reports generated by the system and optionally identifying an alignment report to be exported.

System Post-Search Functions. The system's post-search functions correspond to reporting capabilities of the operational concept. The purpose of these functions is to render the outputs for presentation in an alignment report. They include

- Organizing (i.e., associating each hit/non-hit with applicable metadata, associating sets of hits with the task statements they address, associating sets of task statements with their aligned hits, and grouping hits/non-hits into subsets).
- Extracting relevant text (i.e., getting text of each hit that will be presented in the alignment report)
- Counting (i.e., identifying how many hits are aligned with how many task statements, and vice-versa))
- Exporting (i.e., exporting the alignment report identified for export by the analyst)

The organizing functions will group hits into subsets based on what formal definition of alignment they satisfy. The operational concept identifies 15 such definitions (i.e., one for exactly matching text and 14 for partially matching text). These definitions of alignment are applicable to one task statement token and one document token.

Leveraging groups of hits and descriptive statistics, the organizing functions will support formal definitions of alignment that are applicable to sets of tokens that come from multiple task statements and documents:

- *Redundant* will refer to multiple hits addressing the same task statement.
- *Outdated* will refer to a non-hit addressing equipment that is no longer in use
- *Complete* will refer to hits that address particular task statements and are not in the same group as hits that address different task statements

A formal definition of *correct* will aid in ensuring that the identified hits contain sufficiently relevant text with respect to the task statements while the identified non-hits do not.

RECOMMENDED TOOLS AND ALGORITHMS. In this section we identify recommended tools and algorithms for executing the system's pre-search, search, and post-search functions (Tables 4–6). Each identified tool or algorithm is described in the literature review of text search approaches.

No single tool or algorithm recommended in Tables 4–6 addresses every function of interest; while in some cases, there are multiple tools or algorithms addressing the same function. Thus, Table 7 identifies *primary* recommendations that are hypothesized best choices for a future implementation of the approach.

Pre-Search Tools and Algorithms. Table 4 identifies recommended tools and algorithms for supporting the system's pre-search functions.

Table 4: Tools and algorithms supporting system pre-search functions.

Function	Tool/algorithm citation	Rationale
Metadata extraction	N/A	Supported by widely utilized programming languages
Text extraction	Apache Software Foundation (2017d,c,b); Google (2017); Breuel (2017)	Can extract digitally encoded characters or execute OCR for all applicable file formats (Table 1)
Sentence tokenizer	Manning et al. (2014)	Can break up text into grammatically complete sentences
Word tokenizer	Apache Software Foundation (2016); Alias-i (2011)	Can break up sentence tokens into sentence with word tokens
Part of speech annotator	Apache Software Foundation (2017a); Manning et al. (2014)	Can assign part of speech attributes to word tokens
Expansion filter	Schwartz and Hearst (2002); Taghva and Vyas (2011); Park and Byrd (2001)	Can replace acronyms, initialisms, and abbreviations with their expansions (or vice-versa)
Stop word filter	Apache Software Foundation (2016); Alias-i (2011)	Can remove unimportant words
Lowercase filter	Apache Software Foundation (2016); Alias-i (2011)	Can replace uppercase letters with lowercase
Punctuation filter	Apache Software Foundation (2016); Alias-i (2011)	Can remove punctuation characters
Stem filter	Apache Software Foundation (2016); Alias-i (2011)	Can replace stemmed words with roots
Indexer	Apache Software Foundation (2016)	Can organize tokens into an applicable index (Fig. 6)

Search Tools and Algorithms. Table 5 identifies recommended tools and algorithms for supporting the system's search functions.

Table 5: Tools and algorithms supporting system search functions.

Function	Tool/algorithm citation	Rationale
Text retrieval	Apache Software Foundation (2016)	Can retrieve tokens from an index
Exact match algorithm	Levenshtein (1966)	Can determine if two sentence tokens match exactly
Partial match algorithm	Li et al. (2004); Islam and Inkpen (2008)	Can address all applicable properties of partially matching text for two sentence with words tokens

Post-Search Tools and Algorithms. Table 6 identifies recommended tools and algorithms for supporting the system's post-search functions.

Table 6: Tools and algorithms supporting system post-search functions.

Function	Tool/algorithm citation	Rationale
Organizing	Hearst (1995); Zamir and Etzioni (1999)	Can address formal definitions of alignment that address sets of task statement tokens and sets of document tokens
Extracting relevant text	Tombros and Sanderson (1998)	Can extract tokens of a document that are most relevant to a job task statement
Computing statistics	N/A	Descriptive statistics can be stored in an index
Exporting	N/A	Supported by widely utilized programming languages

Primary Recommendations. Table 7 identifies one recommended tool or algorithm for supporting all pre-search, search, and post-search functions. The identified tools are implemented in Java, and the identified algorithms can be instantiated in Java. Thus, the tools and algorithms identified in Table 7 are inter-operable.

DISCUSSION

This work identified recommendations for an approach to analyzing the alignment of existing Tech Ops courses and JTA data. We accomplished this by characterizing the operational concept of such an approach; reviewing potentially useful tools and algorithms from the text search literature; and specifying recommendations for a text search approach to alignment.

The operational concept identifies what data handling, text comparison, and reporting capabilities are needed to facilitate alignment analyses. Data selection capabilities enable the analyst to identify inputs of interest (Tech Ops curriculum courses of JTA data). Data handing capabilities support utilization of applicable data from the selected inputs: text data from text-based documents of the selected Tech Ops curriculum courses (Table 1), metadata from the text-based documents (i.e., filenames, directories, and subdirectories), and full-sentence job task statements from the selected JTA data. Text comparison algorithms identify what documents have text that is sufficiently relevant to one or more job task statements, where properties of matching text are a proxy for relevance and relevance is a proxy for alignment. To inform the text comparison algorithms, we specified what properties of matching text are applicable for a training document's sentences, words, and sentences with words. A training document exhibits an *exact match* property if any of its sentences match a task statement character for character. A training document exhibits a *partial match* property if

- Any of its words match a task statement as specified in Fig. 2
- Any of its sentences with words match a task statement as specified in Fig. 3

Applicable text comparison algorithms facilitate the generation of alignment reports that aid in identifying

- The text-based document(s) that are aligned with each task statement
- The exact match and partial match properties that each aligned document exhibits

Table 7: Recommended tools and algorithms supporting system pre-search, search, and post-search functions.

Function	Tool/algorithm citation	Rationale
Text extraction	Apache Software Foundation (2017d)	Supports all applicable file formats and comes with the Tesseract OCR engine built in
Sentence tokenizer	Manning et al. (2014)	Can break up text into grammatically complete sentences
Word tokenizer	Apache Software Foundation (2016)	Includes an indexer and filters (stop word, lowercase, punctuation, and stem)
Part of speech annotator	Manning et al. (2014)	Includes the only recommended sentence tokenizer
Expansion filter	Park and Byrd (2001)	Achieves the highest accuracy relative to other reviewed algorithms (97%)
Stop word filter	Apache Software Foundation (2016)	Includes an indexer and other identified filters (lowercase, punctuation, and stem)
Lowercase filter	Apache Software Foundation (2016)	Includes an indexer and other identified filters (stop word, punctuation, and stem)
Punctuation filter	Apache Software Foundation (2016)	Includes an indexer and other identified filters (stop word, lowercase, and stem)
Stem filter	Apache Software Foundation (2016)	Includes an indexer and other identified filters (stop word, lowercase, and punctuation)
Indexer	Apache Software Foundation (2016)	Comes with filters built in (stop word, lowercase, punctuation, and stem)
Text retrieval	Apache Software Foundation (2016)	Includes an indexer and filters (stop word, lowercase, punctuation, and stem)
Exact match algorithm	Levenshtein (1966)	Can determine if two sentence tokens match exactly
Partial match algorithm	Islam and Inkpen (2008)	Can compare two sentence with word tokens having different lengths
Organizing	Zamir and Etzioni (1999)	Can group sets of hits based on what task statements they address
Extracting relevant text	Tombros and Sanderson (1998)	Can extract tokens of a document that are most relevant to a job task statement

- The task statement(s) that each text-based document addresses
- The text-based documents are not aligned with any task statements
- The task statement(s) that the text-based documents do not address
- Descriptive statistics about the outputs above

To inform recommendations for such an approach, we reviewed the text search literature about potentially useful tools and algorithms. Our review focused on extracting text data from files having the applicable formats (Table 1), modifying extracted text in ways that support alignment analyses, comparing text of the text-based documents to text of the job task statements in ways that identify what documents are aligned, and rendering outputs of interest within alignment reports that the user can analyze.

Leveraging the literature review, we elicited recommendations for a text approach to alignment. The recommended approach includes an architecture of pre-search, search, and post-search functions executed by the analyst and by the system; extant tools and algorithms that support each function; and one of each tool or algorithm that is a primary

recommendation for a future implementation of the approach.

FUTURE WORK. While Tables 4–6 present a body of research that has yielded an array of algorithms (and in some cases, discrete tools), our effort could be significantly enhanced by leveraging existing technologies that incorporate advanced text analysis capabilities. One research initiative that has demonstrated promising technologies for automating the processing of text and metadata is Tools for the Rapid Development of Expert Models (TRADEM). TRADEM (Robson and Robson, 2014) demonstrates the use of multiple AI algorithms to transform and tag content and store the results in a neutral format (e.g. XML, HTML5) not tied to a specific authoring or delivery technology. TRADEM is currently being extended by the U.S. Army Research Lab to make automated tagging more sophisticated, improve alignment of discovered materials with learning objectives and to process non-text materials by analyzing meta-data.

We also recommend adoption of a competency-based approach to processing and aligning instructional materials with job performance requirements, in accordance with FAA initiatives emphasizing competencies. To incorporate this construct into our project, we proposed to adopt a US Government-funded open source framework called CaSS (Competency and Skills System). CaSS will store competencies (including job requirements and learning objectives), securely maintain persistent competency-based learner records, and track learning goals. CaSS organizes competencies into structured frameworks that can be accessed through standards-based application programming interfaces (APIs) and persistent URLs, and formulates learner profiles from evidence-linked assertions about whether an individual (or team) has or does not have given competency. Assertions are collected from training systems, performance records, and other sources and can be time stamped, assigned a confidence measure, given an expiration date, and indicate how competence degrades over time. CaSS decides if a given competency is held by examining related assertions and applying deterministic or ML algorithms to draw a conclusion. The 2018–2020 roadmap for CaSS, funded by the US Advanced Distributed Learning (ADL) initiative, includes capabilities for storing learning goals and using blockchain technology to enable multiple systems to securely write and read learner's record while enforcing permissions (Robson and Poltrack, 2017).

The recommended approach supports the set of capabilities specified in the operational concept. However, some issues are not addressed and should be explored in future work, including

- The treatment of files that are not text-based documents
- The treatment of text-based documents that are not aligned
- Characterizing alignment with respect to the JTA hierarchy

Treatment of Files that are Not Text-Based Documents. To identify what file formats are applicable in this work we conducted an analysis of eight courses in the existing Tech Ops curriculum, listed below as "**Course number**, *Course name* (size of directory containing materials)."

1. **40289**, *VSCS System Overview* (321.9 MB)
2. **40444**, *EFSTS Hardware Maintenance Training* (32.7 MB)
3. **40611**, *Weather System Processor (WSP)* (129.6 MB)

4. **40664**, *Airport Surveillance Radar (ASR) Model 9* (187.4 MB)
5. **44031**, *Communications Equipment (CE)* (34.4 MB)
6. **45110**, *ARTCC Critical and Essential Power System (ACEPS)* (597.3 MB)
7. **47012**, *Introduction to Telecommunication* (70.7 kB)
8. **47112**, *ARTCC Critical and Essential Power System (ACEPS)* (4.2 GB)

Our analysis identified 277 files having applicable text-based formats (Table 1) and 16,805 files having other formats (Table 8). Future work should explore ways of supporting the

Table 8: Formats and quantities of files in the FAA-supplied Tech Ops curriculum that should be addressed in future work. File extensions marked with "‡" can be addressed using a text extraction tool discussed in this work. File extensions marked with "*" could be addressed using OCR software. Unmarked file extensions could require additional software that was not reviewed in this work.

Extension	Description	Stands for (key letters emphasized)	Number of files
.bmp*	Formatted image file	**B**it**M**ap **P**icture	4
.css	Data file used to format web pages	**C**ascading **S**tyle **S**heet	12
.db	Database file	**D**ata**B**ase	42
.dll	Data file used to store Windows program code	**D**ynamic **L**ink **L**ibrary	15
.dsn	Database file	**D**atabase **S**ource **N**ame	2
.exe	Executable Windows program file	**EXE**cutable	30
.gif*	Formatted image file	**G**raphics **I**nterchange **F**ormat	22
.hlp	Data file used to store Windows program support	**HeLP**	28
.htm‡	Document file used to store a webpage	**H**yper**T**ext **M**arkup	2
.html‡	Document file used to store a webpage	**H**yper**T**ext **M**arkup **L**anguage	206
.ini	Data file used configure a computer application	**INI**tialization	1
.jpg*	Formatted image file	**J**oint **P**hotographic **G**roup	1,291
.js	Data file used to store code	**J**ava**S**cript	949
.mp3	Formatted audio file	**MP**eg **3**	4,799
.mp4	Formatted video file	**MP**eg **4**	34
.pal	Data file used to store colors	**PAL**ette	3
.png*	Formatted image file	**P**ortable **N**etwork **G**raphics	4,393
.psd*	Formatted image file	**P**hoto**S**hop **D**ocument	1
.story	Data file used to store a storyboard or script	**STORY**ist	12
.swf	Adobe Flash file	**S**mall **W**eb **F**ormat	4,784
.txt‡	Document file used to store unformatted text	**TeXT**	8
.u32	Data file used with Adobe Authorware applications	cover data **32**-bit	15
.ucd	Data file used to store a UML model	**U**se **C**ase **D**iagram	2
.wav	Formatted audio file	**WAV**e	16
.x32	Data file used to store Macromedia program plugins	e**X**tension **32**-bit	48
.xml‡	Data file used to store structured data	e**X**tensible **M**arkup **L**anguage	36
.xsd‡	File used to store an XML grammar	**X**ML **S**chema **D**escription	48

file formats in Table 8, such as by converting their data to digitally encoded characters. For some file formats, this could accomplished using tools described in this work, such as common programming languages that can address .txt files, OCR software that can address formatted images (Google, 2017), and Apache Tika that can address formatted Web pages (Apache Software Foundation, 2017d). Other file formats, such as formatted audio files and Adobe Flash files, could require additional tools that were not reviewed in this work. A future analysis should identify what file formats can be addressed automatically using existing software and what file formats could require the development of custom tools.

Treatment of Documents that are Not Aligned. The recommended approach aids in identifying what text-based documents that are not aligned with any task statements.

However, the analyst could also be interested in understanding *why* a text-based document is not aligned. Such information could aid in identifying how a document could be improved with respect to alignment. For example, one issue could be that a document's text is relevant to a piece of equipment that is no longer in service. Such text could be considered *outdated*. At least two areas of future work are needed to address these kinds of issues:

1. Developing formal definitions of non-alignment (e.g. outdated)
2. Rendering decision support within alignment reports that can aid in identifying potential improvements (e.g. updating a document to address a new piece of equipment)

A related issue is identifying what documents are important with respect to alignment, such as textbooks and presentations, and what documents are not, such as attendance sheets and exams. For example, suppose two text-based documents contain a sentence that matches the same job task statement exactly. The first document is a PowerPoint presentation explaining how to execute the job task. The second document is an exam in which the exactly matching sentence is a multiple choice answer. In the recommended approach, the system would identify both documents as aligned; however, the PowerPoint presentation might be relevant with respect to alignment, while the exam might be irrelevant. Thus, the exam could be incorrectly identified as aligned. Future work should explore ways of addressing these situations, such as by automatically grouping documents by purpose (e.g. training, assessment, taking attendance).

Alignment with Respect to the JTA Hierarchy. The recommended approach defines alignment formally with respect to each job task statement of interest, where all job task statements are considered independently of other job task statements. However, the JTA datasets characterize job task statements as hierarchically structured sets of activities, sub-activities, tasks, sub-tasks, steps, and elements (See Appendix B, Fig. 6). Thus, in addition to identifying what files in the Tech Ops curriculum are aligned with what task statements, the analyst could also be interested in identifying what files are aligned with sets of related task statements. For example, suppose there is an activity in the JTA consisting of two sub-activities: one for removing an equipment part and one for replacing it. A document in the Tech Ops training curriculum could have text that is sufficiently relevant to the first sub-activity (i.e., removing the part), but not the second (i.e., replacing the part). Such a document could be considered *incomplete* with respect to the activity because it only addresses one of two sub-activities. Thus, future work should explore new formal definitions of alignment that consider sets of related task statements defined in the JTA hierarchy (e.g. incomplete with respect to an activity).

ACKNOWLEDGMENT

The authors thank Benjamin Bell of Eduworks for supporting this research.

REFERENCES

Abbate, A. J., Bass, E. J., and Stilling, J. (2017). README for Tech Ops Job Task Analysis Workbooks. Technical report.

Achananuparp, P., Hu, X., and Shen, X. (2008). The evaluation of sentence similarity measures. *Data Warehousing and Knowledge Discovery*, pages 305–316.

Alias-i (2011). LingPipe 4.1.2. https://alias-i.com/lingpipe/. Accessed November 14, 2017.

Apache Software Foundation (2016). Apache Lucene™. https://lucene.apache.org/. Accessed November 14, 2017.

Apache Software Foundation (2017a). Apache Open NLP 1.8.3. https://opennlp.apache.org/. Accessed November 14, 2017.

Apache Software Foundation (2017b). Apache PDFBox®. https://pdfbox.apache.org/. Accessed November 14, 2017.

Apache Software Foundation (2017c). Apache POI. https://poi.apache.org. Accessed November 14, 2017.

Apache Software Foundation (2017d). Tika 1.16. https://tika.apache.org/. Accessed November 14, 2017.

Banerjee, S. and Pedersen, T. (2002). An adapted lesk algorithm for word sense disambiguation using wordnet. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145.

Berners-Lee, T. (1991). Hypertext Markup Language 1.0. https://www.w3.org/html/.

Breuel, T. M. (2017). OCRopus 2.0. https://github.com/tmbdev/ocropy. Accessed December 9, 2017.

Brill, E. (1992). A simple rule-based part of speech tagger. In *Proceedings of the Workshop on Speech and Natural Language*, pages 112–116.

Chomsky, N. (2014). *Aspects of the Theory of Syntax*, volume 11. MIT Press, Cambridge, MA.

Cunningham, H. (2002). Gate, a general architecture for text engineering. *Computers and the Humanities*, 36(2):223–254.

Eddy, S. R. (1996). Hidden markov models. *Current Opinion in Structural Biology*, 6(3):361–365.

Frakes, W. B. and Baeza-Yates, R., editors (1992). *Information Retrieval: Data Structures and Algorithms*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Gomaa, W. H. and Fahmy, A. A. (2013). A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13):13–18.

Google (2017). Tesseract Open Source OCR Engine 3.05.01. https://github.com/tesseract-ocr/tesseract/. Accessed December 7, 2017.

Hearst, M. A. (1995). Tilebars: Visualization of term distribution information in full text information access. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 59–66, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

Islam, A. and Inkpen, D. (2008). Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):10.

Kaisser, M., Hearst, M. A., and Lowe, J. B. (2008). Improving search results quality by customizing summary lengths. In *46th Annual Meeting of the Association for Computational Linguistics (ACL) System Demonstrations*, pages 701–709, Stroudsburg, PA.

Konchady, M. (2008). *Building Search Applications: Lucene, LingPipe, and Gate*. Mustru Pub., Oakton, VA, USA.

Kupiec, J. (1992). Robust part-of-speech tagging using a hidden markov model. *Computer Speech & Language*, 6(3):225–242.

Landauer, T. K. (2006). *Latent Semantic Analysis*. Wiley Online Library.

Leacock, C. and Chodorow, M. (1998). Combining local context and WordNet similarity for word sense identification. *WordNet: An Electronic Lexical Database*, 49(2):265–283.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet Physics Doklady*, volume 10, pages 707–710.

Li, Y., Bandar, Z., McLean, D., O'Shea, J., et al. (2004). A method for measuring sentence similarity and its application to conversational agents. In *FLAIRS Conference*, pages 820–825.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., and McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. In *52nd Annual Meeting of the Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Natural Language Toolkit Project (2017). NLTK 3.2.5. https://www.nltk.org. Accessed November 20, 2017.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Park, Y. and Byrd, R. J. (2001). Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 126–133.

Ramos, J. et al. (2003). Using TF-IDF to determine word relevance in document queries. In *Proceedings of the First Instructional Conference on Machine Learning*, volume 242, pages 133–142.

Robertson, S. E. and Jones, K. S. (1976). Relevance weighting of search terms. *Journal of the Association for Information Science and Technology*, 27(3):129–146.

Robson, E. and Poltrack, J. (2017). Using competencies to map performance across multiple activities. In *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2017*, Orlando, FL, USA. National Training and Simulation Association.

Robson, E. and Robson, R. (2014). Automated content alignment for adaptive personalized learning. In *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2014*, Orlando, FL, USA. National Training and Simulation Association.

Schwartz, A. S. and Hearst, M. A. (2002). A simple algorithm for identifying abbreviation definitions in biomedical text. In *Biocomputing 2003*, pages 451–462. World Scientific.

Shima, H. (2017). WordNet Similarity for Java 0.13.9. https://github.com/Sciss/ws4j. Accessed December 26, 2017.

Shneiderman, B., Byrd, D., and Croft, W. B. (1997). Clarifying search: A user-interface framework for text searches. *D-Lib Magazine*, 3(1):18–35.

Simpson, G. G. (1960). Notes on the measurement of faunal resemblance. *American Journal of Science*, 258(2):300–311.

Taghva, K. and Vyas, L. (2011). Acronym expansion via hidden markov models. In *21st International Conference on Systems Engineering (ICSEng)*, pages 120–125.

Thompson, K. (1968). Programming techniques: Regular expression search algorithm. *Communications of the ACM*, 11(6):419–422.

Tian, Y. and Lo, D. (2015). A comparative study on the effectiveness of part-of-speech tagging techniques on bug reports. In *2015 IEEE 22nd International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 570–574.

Tombros, A. and Sanderson, M. (1998). Advantages of query biased summaries in information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 2–10, New York, NY, USA. ACM.

Toutanova, K. and Manning, C. D. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 63–70, Stroudsburg, PA, USA.

Turney, P. (2001). Mining the web for synonyms: PMI-IR versus LSA on TOEFL. *Machine Learning: ECML 2001*, pages 491–502.

Zamir, O. and Etzioni, O. (1999). Grouper: a dynamic clustering interface to web search results. *Computer Networks*, 31(11):1361–1374.

Zobel, J. and Moffat, A. (2006). Inverted files for text search engines. *ACM Computing Surveys (CSUR)*, 38(2):1–56.

APPENDIX A–EQUATIONS

The *word-order similarity* score of a document token with respect to a search-term token is defined in Equation 1:

$$1 - \frac{(X_1 - Y_1) + (X_2 - Y_2) + (X_k - Y_k)}{(X_1 - X_k) + (X_2 - X_{k-1}) + (X_k - X_1)} \tag{1}$$

Here, words of a search-term token and words of a document token are assigned indexes. Two vectors list the indexes of exactly matching words in order of appearance (1, ..., k, where k is the number of exactly matching words). A vector X contains indexes of search-term words that are in the document token, and a vector Y contains indexes of document-token words that are in the search-term token.

The *Li et al. sentence similarity* score of a document token with respect to a search-term token is defined in Equation 2:

$$S(T_1, T_2) = \delta S_s + (1 - \delta) S_r \tag{2}$$

where T_1 and T_2 are the tokens being compared, S_s is the word-overlap measure, S_r is the word-order measure, and δ is the semantical-similarity measure.

The *Islam and Inkpen sentence similarity* score of a document token with respect to a search-term token is defined in Equation 3:

$$S(P, R) = \frac{(\delta(1 - w_f + w_f S_0) + \sum_{i=1}^{|\rho|} \rho_i)(m + n)}{2mn} \tag{3}$$

Here, P and R are the tokens being compared with respect to three variables: δ is the word-overlap measure, S_0 is a word-order measure, and ρ is a vector of combined lexical/semantical measures for each word in the respective tokens. Constant parameters are m, which is the number of words in P; n, which is the number of words in R; and w_f, which is a constant between 0 and 0.5 (chosen by the analyst).

The value of w_f reflects the analyst's beliefs about how important the ordering of words is with respect to sentence similarity. For example, if the analyst believes that the ordering of words is a very important, then w_f could be set to 0.5. If the analyst believes that the ordering of words has no bearing on sentence similarity, then Equation 3 becomes

$$S(P, R) = \frac{(\delta + \sum_{i=1}^{|\rho|} \rho_i)(m + n)}{2mn} \tag{4}$$

APPENDIX B–TECH OPS WORKBOOKS README DOCUMENT

README for Tech Ops Job Task Analysis Workbooks

Andrew J. Abbate *[1], Ellen J. Bass †[1], and John Stillings ‡[2]

[1]Drexel University
[2]U.S. FAA

12/27/2017

This document applies to the following four files:

1. 2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx (4.4 MB)
2. 2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx (11.1 MB)
3. 2013-09-24 Tech Ops JTA Specific Part1.xlsx (8.3 MB)
4. 2013-09-24 Tech Ops JTA Specific Part2.xlsx (11.4 MB)

All four files contain multiple spreadsheets that are organized within named tabs, and each tab name aids in identifying what data are in the spreadsheet (see Fig. 1). These data encompass:

1. Job task analysis (JTA) outputs identifying the complete set of tasks for FAA technicians
2. A notional (i.e., hypothetical) training curriculum that covers the complete set of job tasks

Filenames containing "COMMON" are applicable to all FAA technicians or some FAA technicians within a given specialty. Filenames containing "Specific" and "SPECIFIC" are applicable to FAA technicians who work in a particular facility and use a concomitant set of equipment.

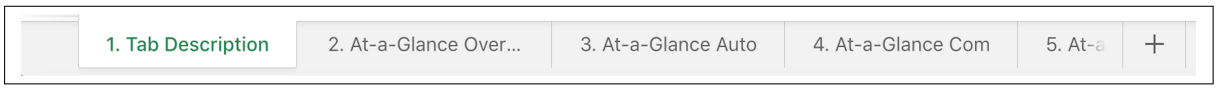

Figure 1: Section of the Microsoft Excel interface showing tabs in the workbook file *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx* (viewed in Excel 2010 for Mac)

1 Conventions of this Document

Starting with Section 2, each section of this document supports understanding of one file. Text under each main section heading identifies tabs within the workbook and a dialog box that may appear upon opening the file. Each subsection heading corresponds to one or more tabs within the file, and sub-subsections are added where supplementary explanations are needed.

Throughout the document, workbook filenames are listed in italic text, and tab names for spreadsheets therein are referenced within quotation marks, e.g. "1. Tab Description." Rows and columns of cells within a spreadsheet are referenced using italic text, e.g. *25* for row-25, *A* for column-A. All cells within a range of rows or columns are referenced using an m-dash between italic text row/column references, e.g. *A*–*Z* for all cells in columns *A*, *B*, . . . , *Z*. Some of the spreadsheets have hidden columns that are identified by thickened vertical column separators (Fig. 2a); thus, where a range of columns is referenced in this document, cells within hidden ones (if any) are excluded.

Finally, constrained sets of cells are referenced using the notation (*column(s), row(s)*), e.g.:

*andrew.j.abbate@drexel.edu
†ellen.j.bass@drexel.edu
‡john.stillings@faa.gov

- (*A, 6*): an individual cell (Fig. 2b)
- (*A–R*, 8): a one-dimensional group of cells within columns-Y through AB and along row-2, excluding hidden columns (Fig. 2c)
- (*U–X, 7–9*): a two-dimensional group of cells within columns-BS through CD and within rows-6 through 370 (Fig. 2c)

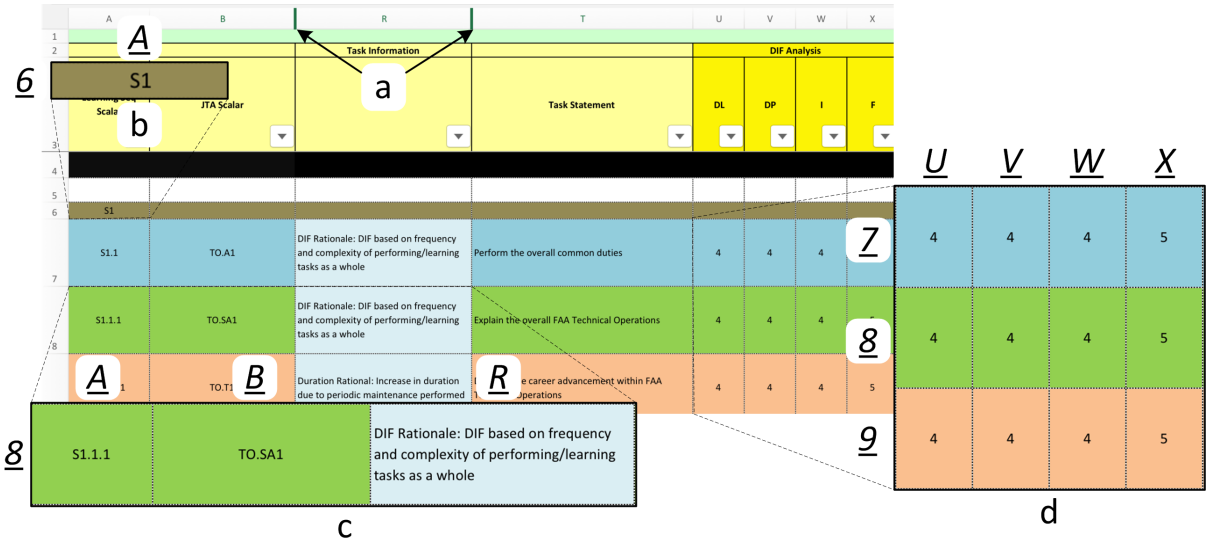

Figure 2: Section of a spreadsheet from *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*. In b–d, underlined italic numbers and letters are added near zoomed-in images to identify row and column indexes respectivey. (a) Arrows point to thickened vertical column separators. These separators indicate that are columns-*C–Q* and *S* are hidden in the spreadsheet. Cells within hidden columns are excluded from all references in this document. (b) Cell at row-6, column-A, which would be referenced as (*A, 6*). (c) One-dimensional group of cells within columns-A–R (columns C–Q hidden) and along row-8, which would be referenced as (*A–R, 8*). (d) Two-dimensional group of cells within columns A–R (columns C–Q hidden) and within rows 7–9, which would be referenced in this document as (*U–X, 7–9*)

1.1 Mathematical Operations

Many of the spreadsheets leverage Excel's built-in functions and arithmetic operators to generate processed data. For broader applicability, subscripts are used to specify function parameters (i.e., cell contents) in a general way. To reduce the need for Excel syntax, parameters having subscripts appear within mathematical equations, which aid in specifying the corresponding Excel commands. While expressing the functions in this way reduces the need for Excel syntax, some familiarity with discrete math is needed. Thus, some knowledge of either Excel syntax or discrete math should be sufficient for understanding.

2 2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx

This workbook contains 11 spreadsheets, each of which contains cells having raw data, data that have been processed via arithmetic operations (i.e. Excel math commands), or exposition (e.g. descriptions of the data, word definitions). Each spreadsheet serves a general purpose that can be broadly characterized in terms of its contents (Table 1).

Upon opening the file in Microsoft Excel, a dialog box similar to the one shown in Fig. 3a may appear, indicating that the workbook contains links to data from other files. These linked files are:

1. (*CUT AND PASTE_FAA_OCEANIC ANALYSIS_CURRICULUM MAPPING_D01_01MAR12.xls*
2. (*FAA%20TECHOPS%20D67V1_APRIL2013_COMMUNICATIONS.xlsm*

Table 1: Spreadsheets of *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx* by tab name and general purpose in terms of its contents. A "★" indicates that the purpose of the spreadsheet is to provide referent content: raw data, processed data, or exposition

Tab name	Spreadsheet contents		
	Raw data	Processed data	Exposition
1. Tab Description			★
2. At-a-Glance Overall Common		★	
3. At-a-Glance Auto		★	
4. At-a-Glance Com		★	
5. At-a-Glance Nav		★	
6. At-a-Glance Env		★	
7. At-a-Glance Sur		★	
8. Analysis	★		
23. Verbs			★
25. Initial Rating		★	
26. Refresher Rating		★	

If you have one or both files, the links can be incorporated sequentially by clicking"Update," and then navigating to a file's location using a subsequent dialog. If you do not have either file, click "Ignore Links." Links and other macros will not be discussed further as they are platform-dependent; for example, links and macros are not available on the iOS version of Excel (Fig. 3b).

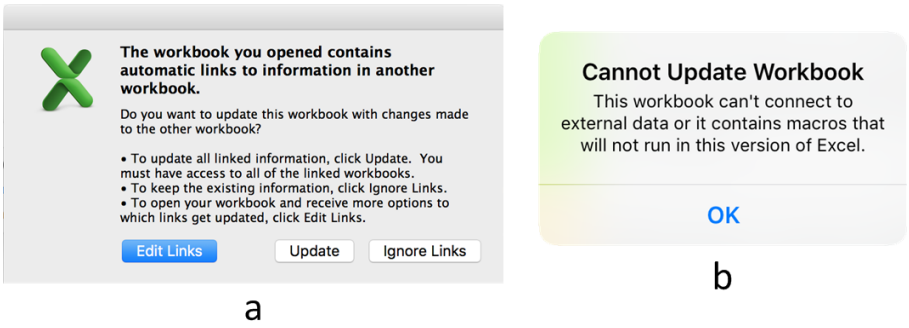

Figure 3: Dialog boxes that may appear upon opening *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*. (a) Microsoft Excel 2010 for Mac. (b) The iOS version of Excel

2.1 Tab Description

This tab has content within (*C–E, 3–39*) that maps to all tabs in this file as well as tabs within *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx*. *C* lists tabs by number (e.g. "2."); *D* lists tabs by name, which is the text immediately following a number in the tab name (e.g. "At-a-Glance Overall Common"); and *E* contains a brief description of the content within each tab (Fig. 4).

Tab Number	Tab Name	Tab Description
1	Tab Description	Description of each tab
2	At-a-Glance Overall Common	Summary of topics and modules by media selection, duration (time to teach, assessment time, remediation time, and time to proficiency) Estimated time to develop the training based on the duration and selected media (ROM)

Figure 4: Segment of "Tab Description" in *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*

A subset of the descriptions in *E* contain boldface text, indicating that the text maps to a tab within this file. Plain text within (*E, 9–22*), (*E, 24*), and (*E, 27*) describes tabs within *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx* (discussed in Section 3).

2.2 Analysis

This tab has content within (*A–GN, 1–4288*) that are organized in a hierarchical-heterarchical way. *1–3* contain text identifying a hierarchical set of categories in which JTA/notional training curriculum data are expressed. The remainder of this section explains these categories and their hierarchical structure.

Color coding aids in identifying the categories, subcategories, and sub-subcategories in *1–3*. These rows remain fixed to the top of the Excel interface while scrolling vertically, which aids in mapping the content in each cell to a position in the categorical hierarchy. Text labels in *1* are applicable to many columns, and screen size constraints may push the text off-screen. To reduce the need for horizontal scrolling, Table 2 identifies the mappings between colors and text[1].

Table 2: Top-level categories in *1* of "8. Analysis"

Columns	Category name	Color
A–AW	**WORK ANALYSIS**	
AZ–CD	**CURRICULUM DESIGN**	
CJ–GN	**KSAOs** (**K**nowledge, **S**kills, **A**bilities, **O**ther characteristics)	

2 identifies subcategories of *1*, and *3* identifies subcategories of *2* (Fig. 5).

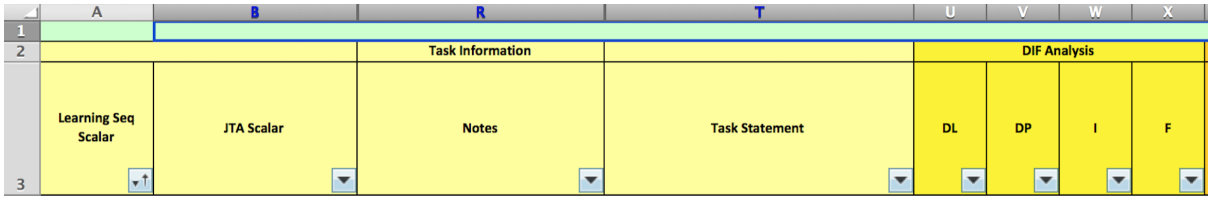

Figure 5: Section of "8. Analysis" in *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*. For this segment of the spreadsheet, colors indicate that *A–X* are in the same category (denoted by the color of *1*), while *A–T* and *U–X* are in respective subcategories (denoted by the color of *2* and *3*. Vertical lines separate each individual column containing data within one subcategory at the lowest level)

Two outlines are provided next to aid in understanding the hierarchical categories of data within (*A–CD, 1–3*). Cell background colors and groups of cells having the same background color are listed, followed by the category name (in boldface text). Brief descriptions of category names are provided in plain text where applicable; and for descriptions of acronyms and initialisms, the first letter of each corresponding word is printed in boldface text for emphasis. Some categories in the outline are explained within later sections of this document or in spreadsheets of a different workbook file, and cross-references to those explanations are identified where applicable.

- (*A–AW, 1*), **WORK ANALYSIS**: JTA data
  - (*A–T, 2*), **Task Information**: content that aids in identifying and understanding what the job tasks are
    - (*A, 3*), **Learning Seq Scalar**: unique identifier for a training curriculum item (explained in Section 2.2.1)
    - (*B, 3*), **JTA Scalar**: unique identifier for a job task item (explained in Section 2.2.1)
    - (*R, 3*), **Notes**: optional, miscellaneous descriptive text
    - (*T, 3*), **Task Statement**: one-sentence task description

[1]Colors are shown as they appeared on the display used to create this document. These colors may appear differently on your display; and for reference, numeric RGB values are provided in Appendix A.

(*U–X, 2*), **DIF Analysis**: a task's **D**ifficulty to learn/perform, **I**mportance, and **F**requency, measured on an ordinal scale of 1–5 from least to most difficult (explained in "22. DIF Criteria and Examples" of *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx*)

- (*U, 3*), **DL**: **D**ifficulty to **L**earn
- (*V, 3*), **DP**: **D**ifficulty to **P**erform
- (*W, 3*), **I**: **I**mportance
- (*X, 3*), **F**: **F**requency

(*Y–AB, 2*), **Additional Task Data**: content that aids in characterizing the job task

- (*Y, 3*), **Knowledge or Performance**: whether the job task item involves knowing or performing (cannot be both)
- (*Z, 3*), **Activity, Sub-Activity, Task, Sub-Task**: granularity identifier for the job task item (explained in Section 2.2.1)
- (*AA, 3*), **Knowledge Proficiency**: level of knowledge proficiency, measured on an ordinal scale of a–d from least to most proficient (explained further in "21. Proficiency Level" of *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx*)
- (*AB, 3*), **Proficiency (expected output proficiency** [sic][2]: Level of performance proficiency, measured on an ordinal scale of 1–4 from least to most proficient (explained further in "21. Proficiency Level" of *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx*)

(*AC–AD, 2*), **Media Recommendation**: how the job task item should be taught and why, informed by input from SMEs

- (*AC, 3*), **Suggested Instructional Method (DIF Calculated)**: one of three instructional methods: **FAC**ilitated (FAC), **S**elf-**P**aced (SP), or blended (mix of FAC and SP) (explained further in "21. Delivery Options" of *2013-09-16 Tech Ops_JTA_COMMON_Part2_-V01.1_.xlsx*)
- (*AD, 3*), **Suggested Instructional Method Rationale**: bulleted list of reasons for selecting the instructional method identified in *AC*

(*AH–AI, 2*), **Task Priority**: relative importance of training for a task, measured on an ordinal scale of low, medium, or high

- (*AH, 3*), **Initial Training Priority Rating**: relative importance of initial training (discussed in Section 2.5)
- (*AH, 3*), **Refresher Training Priority Rating**: relative importance of refresher training (discussed in Section 2.5)

(*AJ–AW, 2*), **Audience Identification**: FAA technicians who must perform job task items and receive the corresponding training

- (*AJ–AN, 3*), **Tech Ops Specialist-**: Personnel belonging to one of five specialty areas: Automation (*AJ*), Environmental (*AK*), Navigiation (*AL*), Surveillance (*AM*), or Communication (*AN*)
- (*AO, 3*), **Common, Cross Functional, Common Specific**: one of three audience identification classifications (identified in title). This column is processed data derived from columns *AJ–AN*. More than four "X"s in a row means data therein are common to all specialties (*Common*), while one "X" means they apply to a specific specialty (*Common Specific*). Otherwise, they apply across a few specialties (*Cross Functional*)
- (*AQ–AW, 3*), Assorted titles, e.g. **Tech Ops Manager**: Managerial or supervisory personnel

[2] There should be a closing parenthesis here, but it is omitted

(*AZ–CD, 1*), **CURRICULUM DESIGN**

(*AZ–BI, 2*), **Learning Objectives**

(*AZ, 3*), **Objective Type (Terminal or Enabling)**: categorical measure of the learning objective as either terminal or enabling. The meanings of "terminal" and "enabling" are not defined

(*BA, 3*), **Objective Verb**: one verb identifying the action being instructed (e.g. "describe," "repair")

(*BB, 3*), **Objective Object**: the piece of equipment for which task knowledge or performance is being instructed (e.g. "the real-time clock")

(*BC, 3*), **Objective Detail**: a noun or noun-phrase that aids in categorizing the objective object (e.g. "software," "contaminated materials")

(*BD, 3*), **Objective Condition**: the equipment to which the objective object belongs (e.g. "the Microprocessor En Route Automated Tracking System")

(*BE, 3*), **Objective Standard**: guidance, order, notice, or standards document that informs training requirements. Some cells reference an exact document (e.g. "JO 6190.16"), while others are ambiguous (e.g. "available documentation")

(*BI, 3*), **Objective Statement**: a one-sentence statement of the learning objective

(*BL–BR, 2*), **Learning Method Analysis**

(*BL, 3*), **Baseline Training Time (Hours) Based on ILT**: estimated training time in hours, expressed as a decimal and based on the instructor-led training (ILT) delivery method

(*BN, 3*), **Assessment Time Based on ILT**: same as above for test-taking time

(*BO, 3*), **Remediation Time Based on ILT**: same as above for remediation

(*BP, 3*), **Total Training Time Based on ILT**: sum of the values in *BN–BP*

(*BQ, 3*), **Recommended Delivery Method (SME)**: identifies the course delivery method recommended by a **S**ubject **M**atter **E**xpert (SME). Can be one of eight methods defined in "21. Delivery Options" of *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_-.xlsx*

(*BR, 3*), **New Instructional Training Time based on Recommended Delivery Method (SME)**: new total training time that is calculated using the fraction of one ILT hour that the new instructional delivery method requires, expressed as a decimal. For example, (*BL, 14*) contains *0.5* (i.e., one half-hour of ILT) and (*BQ, 14*) contains *eLearning* (i.e. the recommended delivery method). Because eLearning requires 0.7 ILT hours, (*BR, 14*) contains *0.35* (i.e., 0.5×0.7)

(*BS–CD, 2*), **Curriculum Build**: describes the training curriculum item

(*BS, 3*), **Learning Sequence Scalar**: unique identifier for a training curriculum item (explained in Section 2.2.1)

(*BT, 3*), **Curriculum Area (Stream)**: the curriculum area, which is almost always "Tech Ops" in this file. Exceptions are blank cells in *1264, 1813, 1916, 2778, 3921,* and *4286–4289*. The meaning of "Stream" in this context is not defined in any of the six files

(*BU, 3*), **Curriculum Area (Sub Stream) Sequence**: identifies the temporal ordering of training curricula by sub-area, ranging from S1–S6 (first–last). Each identifier corresponds to one sub-area, listed in *BV*

(*BV, 3*), **Curriculum Sub Area (Sub Stream)**: identifies the curriculum sub-area corresponding to an identifier S1–S6. Can be one of the following (corresponding identifiers in parentheses): Overall Common (S1), Automation Common (S2), Communication Common (S3), Navigation Common (S4), Environmental Common (S5), Surveillance Common (S6)

(*BW, 3*), **Course Sequence**: identifies the temporal ordering of courses taught in the curriculum sub-area from 1–9 (first–last). Note that some curricula have fewer than nine courses

(*BX, 3*), **Course Title**: name of the course

(textitBY, 3), **Module Sequence**: identifies the temporal ordering of modules taught in the course from 1–17 (first–last). Note that some courses have fewer than 17 modules

(*BZ, 3*), **Module Title**: name of the module

(*CA, 3*), **Lesson Sequence**: identifies the temporal ordering of lessons taught in the module from 1–21 (first–last). Note that some modules have fewer than 21 lessons

(*CB, 3*), **Lesson Title**: name of the lesson

(*CC, 3*), **Topic Sequence**: identifies the temporal ordering of topics taught in the lesson from 1–16 (first–last). Note that some lessons have fewer than 16 topics, and some topics are decomposed (e.g. 1.1, 1.2, 1.3). The term "sub-topic" is not utilized to describe such a decomposition

(*CD, 3*), **Topic Title**: name of the topic

(*CJ–GN, 2*) identify KSAOs and subcategories that are sets of similar KSAOs, while each individual column along *3* identifies one KSAO within a subcategory. Below *3*, cells have no text and their backgrounds are colored gray, blue, or white. A legend within (*CJ–CK, 2–3*) explains the meanings of cell background colors:

The KSAO is used to perform a task

The KSAO is used to perform a task and emphasized in training

A cell with no background color (i.e. a white background) indicates that the KSAO is neither used to perform a task nor emphasized in training.

Along *2*, four sets of columns are demarcated by background color, one for each word represented in the initialism "KSAO." Within each set of columns having the same background color, KSAO subcategories are identified using the notation "Category: subcategory" (e.g. "Knowledge: FAA procedures" in (*CL–DE, 2*). Screen size constraints may push this text labels off-screen; thus, as a visual aid, the "Category: subcategory" labels and background colors are listed below. Plain text is added to identify suspected typos. The list is only meant to address screen size constraints; thus, individual subcategories in each column along *3* are not included, and the semantics of "Category: subcategory" labels are not explained.

(*CJ–GN, 1*), **KSAOs**

(*CL–DE, 2*), **Knowledge: FAA Procedures**

(*DF–EE, 2*), **Knowledge: technical common**

(*EF–EI, 2*), **Knowledge: technical environmental**

(*EJ, 2*), **Knowledge: technical telecommunications**

(*EK–EP, 2*), **Knowledge: other procedures**

(*EQ–EX, 2*), **Knowledge: basic and advanced technical computer**

(*EY–FD, 2*), **Skills: communication (oral and written)**

(*FE, 2*), **Skills: Skills: interpersonal**

(*FF–FH, 2*), **Skills: physical**

(*FI–FK, 2*), **Skills: supervisory**

(*FL–FO, 2*), **Skills: technical**

(*FP–FW, 2*), **Abilities: Knowledge**

(*FX–GB, 2*), **Abilities: decision making**

(*GC–GG, 2*), **Abilities: physical**

(*GH–GJ, 2*), **Abilities: professional**

(*GK–GL, 2*), **Abilities: technical**

(*GM, 2*), **Ability: availability**

(*GN, 2*), **Other**

As mentioned, *4* and *5* contain empty cells with black and white backgrounds respectively. Each subsequent row, starting with *6*, contains all data for one course curriculum item and one job task item. The structure of these data is explained in Section 2.2.1.

2.2.1 Job Task–Course Curriculum Pairings in "8. Analysis"

Paired job task–course curriculum items are organized in a hierarchical-heterarchical way using three redundant techniques:

1. Outline-like identifiers (called scalars)
2. Verbal identifiers of:
   (a) Course curriculum items (curriculum sub-area, course, module, lesson, topic)
   (b) Job task items (activity, sub-activity, task, sub-task, step)
3. Color coding

Learning Sequence Scalars (first appearing in *A*, see Fig. 5) identify training curriculum items, while JTA scalars (first appearing in *B*, see Fig. 5) identify job task items. Each scalar is unique such that one JTA Scalar identifies one job task item and one Learning Sequence Scalar identifies one course curriculum item. To aid in keeping track of the job task–course curriculum hierarchy, scalars are encoded using an outline-like notation: a top-level scalar is a token (e.g. "S1") and decomposed scalars are appended with a period and a subsequent token (e.g. "S1.1" for the next-level down, "S1.1.1" for the subsequent-level down).

In regard to verbal identifiers, the job task hierarchy and training curricula are structured such that activities map to courses, subactivities map to modules, tasks map to lessons, and sub-tasks map to topics. If a sub-task is decomposed into steps, subsequent topics also map to steps (i.e., there are no "sub-topics").

Finally, color coding provides a third way of characterizing the hierarchical structure. In *A–CD*, all cells in the same row have the same background color, and all rows with the same background color are heterarchical. As a visual aid, Fig. 6 combines scalars, verbal identifiers, and background colors in outline form.

2.3 At-a-Glance Overall Common—At-a-Glance Overall Sur

Leveraging a subset of the data in "8. Analysis," the next-six tabs contain processed data about the notional course curriculum for a respective sub-area:

"1. At-a-Glance Overall Common": notional courses for all FAA technicians

"2. At-a-Glance Auto": notional courses for FAA technicians having job tasks in the automation sub-area (e.g. monitoring traffic flow management system)

"3. At-a-Glance Com": notional courses for FAA technicians having job tasks in the communication sub-area (e.g. repairing digital voice recorder)

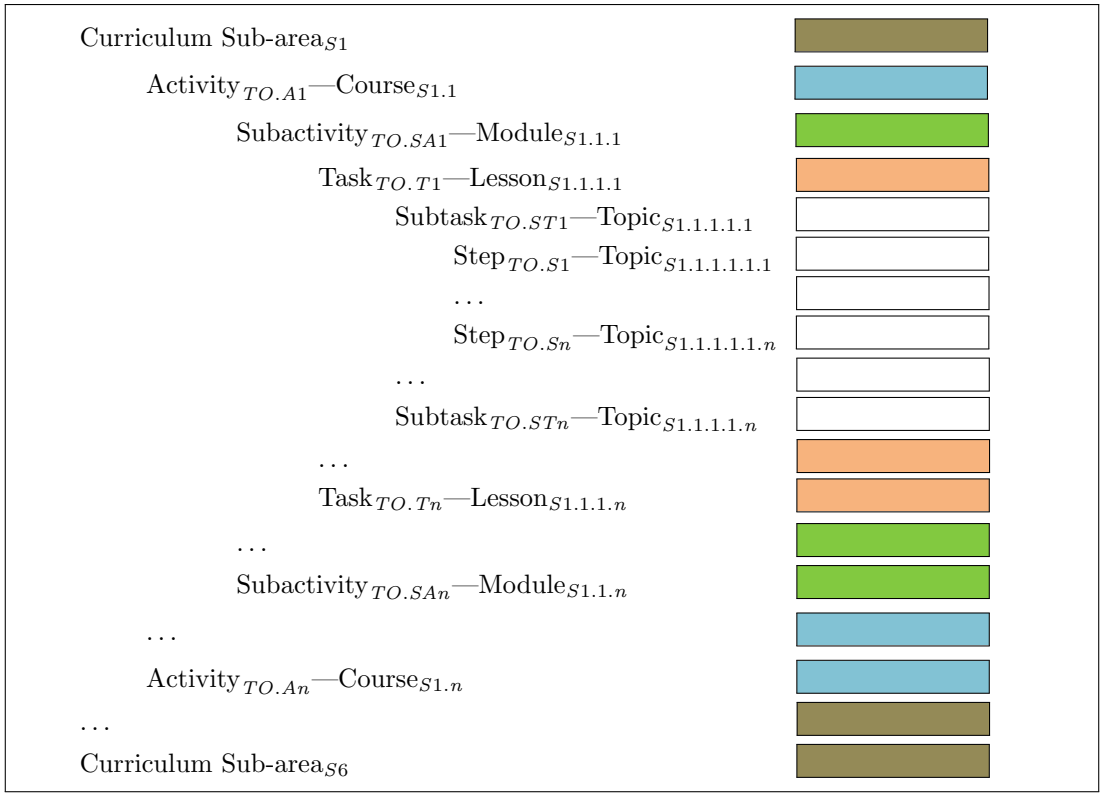

Figure 6: General, outline-form representation of the job task and course curriculum hierarchies. Colors aid in identifying the corresponding cell backgrounds for heterarchical rows. Scalars are shown generally using subscripts. The terminal subscript element n is any positive integer, and the value of n is local to the level in which it appears. "..." appears in place of $n-1$ heterarchical rows

"4. At-a-Glance Nav": notional courses for FAA technicians having job tasks in the navigation sub-area (e.g. configuring the local area automation system)

"5. At-a-Glance Env": notional courses for FAA technicians having job tasks in the environmental sub-area (e.g. maintaining control room HVAC systems)

"6. At-a-Glance Sur": notional courses for FAA technicians working in the surveillance sub-area (e.g. replacing waveguide assembly components)

Each spreadsheet leverages a subset of data within "8. Analysis" to compute two sets of measures: one with respect to instructional delivery method (IDM), and one with respect to the lessons in the curriculum. The first set of measures appears within (*D–L, 6–13*) of each respective spreadsheet. In the list of measures and explanations below, each row is expressed generally as row_i, where $6 \leq i \leq 13$. This notation aids in expressing arithmetic operations generally.

1. **Instructional Delivery Methods (IDM)** (*D*, row_i): listed and defined in "24. Delivery Options" of *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx*
2. **NUMBER OF LESSONS** (*E*, row_i): lessons employing the IDM in (*D*, row_i) (i.e., courses, modules, and topics are not counted), expressed as an integer
3. **PERCENT** (*F*, row_i): (contents of (*E*, row_i) / contents of (*E, 13*))$\times 100$, rounded to the nearest tenth
4. **TIME TO TEACH** (*G*, row_i): sum of developmental instruction hours for all lessons using the IDM in (*D*, row_i), rounded to the nearest hundredth
5. **ASSESSMENT TIME** (*H*, row_i): sum of hours spent assessing knowledge or performance for all lessons using the IDM in (*D*, row_i), rounded to the nearest hundredth

6. **REMEDIATION TIME** (*I*, row_i): sum of remedial instruction hours for all lessons using the IDM in (*D*, row_i), rounded to the nearest hundredth
7. **TIME TO PROFICIENCY** (*J*, row_i): contents of (*G*, row_i) + contents of (*H*, row_i) + contents of (*I*, row_i)
8. **COMPRESSION FACTOR FOR DEVELOPMENT BASED ON MEDIA** (*K*, row_i): percentage of one instructor-led-training hour required for a lesson applying the IDM in (*D*, row_i), expressed as a decimal. This measure is not computed using JTA data, and it is the same in every spreadsheet
9. **Development Ratios (IDM:1)** (*L*, row_i): number of hours spent developing course materials for one lesson employing the IDM in (*D*, row_i). This measure is not computed using JTA data, and it is the same in every spreadsheet

In all six spreadsheets, the second set of measures is organized within 11 columns, *B*–*L*. The first measure always appears in (*B, 18*) and the last measure appears in (*L*, *n*), where *n* is the sum of courses, modules, and lessons in the referent sub-area. Measures are computed for each lesson appearing in a cell within *D*. In the list of measures and explanations below, each row is expressed generally as row_j, where $18 \leq j \leq n$. As mentioned, all of these measures are derived from data in "8. Analysis." Some of them involve values from cells in the above list of measures; thus, row_i references such a row.

1. **MEDIA**, (*E*, row_j): The IDM from (*D*, row_i) applied to teach the lesson (IDMs explained in "24. Delivery Options" of *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx*)
2. **TOPICS**, (*F*, row_j): *number of topics in the lesson* + 1 (i.e., the lesson itself is counted as a topic), expressed an integer, derived from *CB* of "8. Analysis"
3. **MODULE %**, (*G*, row_j): ((contents of (*F*, row_j)/ (*number of topics in the module* + *number of lessons in the module*)) $\times 100$, rounded to the nearest whole integer. The numbers of topics and lessons in the module are derived from *F*
4. **TIME TO TEACH**, (*H*, row_j): developmental instruction hours for the lesson, rounded to the nearest hundredth, derived from *BL* of "8. Analysis"
5. **ASSESSMENT TIME**, (*I*, row_j): hours spent assessing knowledge or performance for the lesson, rounded to the nearest hundredth, derived from *BN* of "8. Analysis"
6. **REMEDIATION TIME**, (*J*, row_j): remedial instruction hours for the lesson, rounded to the nearest hundredth, derived from *BO* of "8. Analysis"
7. **TIME TO PROFICIENCY**, (*K*, row_j): contents of (*H*, row_j) + (contents of J, row_j)
8. **ROM DEVELOPMENT TIME (Based on Time To Teach)**, (*L*, row_j): contents of (*H*, row_j) $\times$ contents of (*K*, row_i), rounded to the nearest whole integer. The meaning of "ROM" is not defined

2.4 Verbs

This tab has content within (*B–D, 2–81*) defining all objective verbs in (*BA, 6–4288*) of "8. Analysis." *B* lists six domains in which the verbs and definitions are categorized; *C* lists each verb; and *D* lists one or more verb definitions, separated by semicolons.

2.5 Initial Rating and Refresher Rating

These tabs have processed data and explanatory text within (*A–T, 1–63*). In either spreadsheet, the analyst can interactively create three sets of job task items having an increasingly important training priority: *Low*, *Medium*, and *High*. "25. Initial Rating" supports these priority calculations for initial training, while "26. Refresher Rating" supports similar calculations for refresher training. Here, initial rating provides a guide to the priority for training job tasks. Refresher rating identifies priorities for job tasks that need refresher training to avoid skill decay.

In either spreadsheet, what job task items are in each set depends on a parameter called *priority_number*: an integer-valued output based on a subset of data from "8. Analysis." These formulas are

different for initial and refresher training; and in both spreadsheets, the respective formulas are shown for one job task item in (*N–T, 9*). Utilizing the notation from Section 2.3 to reference cells from "8. Analysis," the formula for initial training priority number is shown more generally for n job task items in (1).

$$\begin{aligned} priority\_number_i &= F(2I + DL + DP) \\ &\text{where:} \\ F &= \text{contents of } (X, row_i) \text{ in "8. Analysis"} \\ I &= \text{contents of } (W, row_i) \text{ in "8. Analysis"} \\ DL &= \text{contents of } (U, row_i) \text{ in "8. Analysis"} \\ DP &= \text{contents of } (V, row_i) \text{ in "8. Analysis"} \\ 1 &\leq i \leq n \end{aligned} \tag{1}$$

Using the same notation, the refresher training priority number formula is shown generally for n job task items in (2).

$$\begin{aligned} priority\_number_i &= \frac{2I + DP}{F} \\ &\text{where:} \\ F &= \text{contents of } (X, row_i) \text{ in "8. Analysis"} \\ I &= \text{contents of } (W, row_i) \text{ in "8. Analysis"} \\ DP &= \text{contents of } (V, row_i) \text{ in "8. Analysis"} \\ 1 &\leq i \leq n \end{aligned} \tag{2}$$

The elements in *Low*, *Medium*, and *High* are computed in three steps, which are the same for "25. Inital Rating" and "26. Refresher Rating." First, the analyst sets the lower- and upper-bounds of priority number for a job task item in *Medium*: a lower bound is entered within (*L, 3*) and an upper bound is entered within (*L, 4*). These values must be integers, and if the analyst enters a string containing any characters other than integers, the string is interpreted as a very large number (i.e., infinity). Pressing the return key or clicking anywhere outside the current cell initiates the second and third steps in parallel:

2. Excel computes the priority numbers for all job task items in "8. Analysis," and
3. Excel computes (and displays) the the number of job task items in *Low*, *Medium*, and *High*

Low, *Medium*, and *High* are computed using the bounds on *Medium* defined in the first step and the priority numbers calculated in the second step. In both spreadsheets, the number of job task items between the bounds[3] on *Medium* is displayed in (*O, 5*). This value is specified generally for n job task items in (3), which states that:

- *Medium* contains job task items in "8. Analysis" having a priority number that is greater than the value in *(L, 3)* and less than the value in *(L, 3)*
- The number of job task items in *Medium* is displayed in (*O, 5*)

$$\begin{aligned} Medium &= \{priority\_number : P \mid l < priority\_number < u\} \\ \#Medium &= \text{contents of } (O, 5) \\ &\text{where:} \\ P &= \{priority\_number_1, priority\_number_2, \ldots, priority\_number_n\} \\ l &= \text{contents of } (L, 3) \text{ in "25. Inital Rating" or "26. Refresher Rating"} \\ u &= \text{contents of } (L, 4) \text{ in "25. Inital Rating" or "26. Refresher Rating"} \end{aligned} \tag{3}$$

2.5.1 Low- and High- Priority Training

The numbers of job task items in *Low* and *High* are calculated in a similar way; however, job task items having priority numbers equal to the bounded integers are counted. For "25. Initial Rating," the set of

[3]Here, *between the bounds* means that any job task items having priority numbers equal to the bounding integers will not be counted.

job task items in *Low* have priority numbers greater than or equal to 4 and less than or equal to the integer in *(3, L)*. Here, the lower bound is 4 because there are no job task items having an initial training priority number less than 4. If the analyst enters such an integer, the number of job task items in *Low* will be 0. This calculation is specified generally in (4) for n job task items.

$$\begin{aligned} Low &= \{priority\_number : P \mid 4 \leq priority\_number \leq l\} \\ \#Low &= \text{contents of } (O, 4) \text{ in “25. Inital Rating”} \\ &\text{where:} \\ P &= \{priority\_number_1, priority\_number_2, \ldots, priority\_number_n\} \\ u &= \text{contents of } (L, 3) \text{ in “25. Inital Rating”} \end{aligned} \tag{4}$$

The set of job task items in *High* have priority numbers greater than or equal to the upper bound in *(L, 4)* and less than or equal to 100. Similarly, the upper bound is 100 because there are no job task items having an initial training priority number that is greater than 100; and if the analyst enters any non-integer string of characters or an integer that is greater than 100, the number of job task items in *High* will be 0. This calculation is specified generally in (5) for n job task items.

$$\begin{aligned} High &= \{priority\_number : P \mid u \leq priority\_number \leq 100\} \\ \#High &= \text{contents of } (O, 6) \text{ in “25. Inital Rating”} \\ &\text{where:} \\ P &= \{priority\_number_1, priority\_number_2, \ldots, priority\_number_n\} \\ u &= \text{contents of } (L, 5) \text{ in “25. Inital Rating”} \end{aligned} \tag{5}$$

For “26. Refresher Rating,” the set of job task items in *Low* have priority numbers greater than or equal to 0 and less than or equal to the integer in *(L, 3)*. As before, the lower bound is 0 because there are no job task items having an initial training priority number that is less than 0; and if the analyst enters a negative integer, the number of job task items in *Low* will be 0. This calculation is specified generally in (6) for n job task items.

$$\begin{aligned} Low &= \{priority\_number : P \mid 0 \leq priority\_number \leq l\} \\ \#Low &= \text{contents of } (O, 4) \text{ in “26. Refresher Rating”} \\ &\text{where:} \\ P &= \{priority\_number_1, priority\_number_2, \ldots, priority\_number_n\} \\ u &= \text{contents of } (L, 3) \text{ in “26. Refresher Rating”} \end{aligned} \tag{6}$$

The set of job task items in *High* have priority numbers greater than or equal to the upper bound in *(L, 4)* and less than or equal to 50. Again, the upper bound is 50 because there are no job task items having a refresher training priority number greater than 50; and if the analyst enters any non-integer string of characters or an integer that is greater than 50, the number of job task items in *High* will be 0. This calculation is specified generally in (7) for n job task items.

$$\begin{aligned} High &= \{priority\_number : P \mid u \leq priority\_number \leq 50\} \\ \#High &= \text{contents of } (O, 6) \text{ in “26. Refresher Rating”} \\ &\text{where:} \\ P &= \{priority\_number_1, priority\_number_2, \ldots, priority\_number_n\} \\ u &= \text{contents of } (L, 5) \text{ in “26. Refresher Rating”} \end{aligned} \tag{7}$$

The outputs of training priority calculations are expressed as percentages within *(P, 4–6)* of both spreadsheets. All values in *(O–P, 4–6)* are also expressed within embedded pie charts that can be saved and exported as image files. Additionally, as indicated in the outline of *(A–W, 1–3)* within Section 2.2, initial and refresher training priorities are shown for each job task item within *(AH–AI, 7–4288)* of “8. Analysis.” Like the contents in “25. Inital Rating” and “26. Refresher Rating,” the contents in *(AH–AI, 7–4288)* of “8. Analysis” update when the analyst changes upper- or lower-bounds on *Medium* job task items.

3 2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx

This workbook contains a subset of the raw and processed data from *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx* and additional exposition that is not in *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1_.xlsx*. Thus, most of what is in Section 2 also applies here. There are 17 spreadsheets, each of which serves a general purpose that can be broadly characterized in terms of its contents (Table 3).

Table 3: Spreadsheets of *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1.xlsx* by tab name and general purpose in terms of its contents. A "★" indicates that the purpose of the spreadsheet is to provide referent content: raw data or exposition

Tab name	Spreadsheet contents	
	Raw data	Exposition
1. Tab Description		★
9. Cur Build Overall Common	★	
10. Cur Build Auto	★	
11. Cur Build Com	★	
12. Cur Build Nav	★	
13. Cur Build Env	★	
14. Cur Build Sur	★	
15. JTA Overall Comm	★	
16. JTA Automation	★	
17. JTA Communication	★	
18. JTA Navigation	★	
19. JTA Environmental	★	
20. JTA Surveillance	★	
21. Proficiency Level		★
22. DIF Criteria and Examples		★
24. Delivery Options		★
27. Instructional Strategies		★

Upon opening the file in Microsoft Excel, a dialog box similar to the one shown in Fig. 3a may appear, indicating that the workbook contains links to data from other files. These linked files are identified in Section 2, and links can be incorporated using the approach described therein.

3.1 Tab Description

This tab has content within (*C–E, 3–39*) that maps to all tabs in this file as well as tabs within *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1_.xlsx*. *C* lists tabs by number (e.g. "2."); *D* lists tabs by name, which is the text immediately following a number in the tab name (e.g. "At-a-Glance Overall Common"); and *E* contains a brief description of the content within each tab (Fig. 4).

A subset of the descriptions in *E* contain boldface text, indicating that the text maps to a tab within this file. Plain text within (*E, 4–10*), (*E, 23*), and (*E, 25–26*) describes tabs within *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1_.xlsx* (discussed in Section 2).

3.2 Cur Build Overall Common–Cur Build Sur

These six tabs have content within (*B–N*) and rows ranging from 4–107 in "Cur Build Nav" to 4–1264 in "Cur Build Sur," where *Cur* is shorthand for *Curriculum*. Each spreadsheet lists raw data about the notional course curriculum for FAA technical operations personnel in a respective sub-area:

"9. Cur Build Overall Common": all personnel

"10. Cur Build Auto": personnel in the automation specialty

"11. Cur Build Comm": personnel in the communication specialty

"12. Cur Build Nav": personnel in the navigation specialty

"13. Cur Build Env": personnel in the environmental specialty

"14. Cur Build Sur": personnel in the surveillance specialty

In all six spreadsheets, cells along *2* contain text identifying categories of course curriculum data. This row remains fixed to the top of Excel's interface while scrolling vertically. The names of these categories, column locations, and cell background color are listed below.

(*B, 2*), **JTA Scalar**: unique identifier for a job task item (explained in Section 2.2.1)

(*C, 2*), **Objective Statement**: a one-sentence statement of the learning objective

(*D, 2*), **Learning Sequence Scalar**: unique identifier for a training curriculum item (explained in Section 2.2.1)

(*E, 2*), **Curriculum Area (Stream) Sequence**: identifies the temporal ordering of training curricula by sub-area, ranging from S1–S6 (first–last). Each identifier corresponds to one sub-area

(*F, 2*), **Curriculum Area (Stream)**: identifies the curriculum sub-area corresponding to an identifier S1–S6. Can be one of the following (corresponding identifiers in parentheses): Overall Common (S1), Automation Common (S2), Communication Common (S3), Navigation Common (S4), Environmental Common (S5), Surveillance Common (S6)

(*G, 2*), **Course Sequence**: identifies the temporal ordering of courses taught in the curriculum sub-area from 1–9 (first–last). Note that some curricula have fewer than nine courses

(*H, 2*), **Course Title**: name of the course

(textitI, 2), **Module Sequence**: identifies the temporal ordering of modules taught in the course from 1–17 (first–last). Note that some courses have fewer than 17 modules

(*J, 2*), **Module Title**: name of the module

(*K, 2*), **Lesson Sequence**: identifies the temporal ordering of lessons taught in the module from 1–21 (first–last). Note that some modules have fewer than 21 lessons

(*L, 2*), **Lesson Title**: name of the lesson

(*M, 2*), **Topic Sequence**: identifies the temporal ordering of topics taught in the lesson from 1–16 (first–last). Note that some lessons have fewer than 16 topics, and some topics are decomposed (e.g. 1.1, 1.2, 1.3). The term "sub-topic" is not utilized to describe such a decomposition

(*N, 2*), **Topic Title**: name of the topic

Below *3*, each row applies to one course curriculum item, and rows are organized in a hierarchical-heterarchical way. This organization is very similar to that of "8. Analysis" in *2013-09-16 Tech Ops_-JTA_COMMON_Part1_V02.1.xlsx* (see Section 2.2.1); however, job task verbal identifiers are not included.

3.3 JTA Overall Comm–JTA Surveillance

These six tabs have content within (*B–D*) and a variable number of rows from 2–106 in "JTA Navigation" to 2–1263 in "JTA Surveillance." Each spreadsheet lists raw JTA data for FAA technical operations personnel in a respective sub-area:

"15. JTA Overall Comm": all personnel

"16. JTA Automation": personnel in the automation specialty

"17. JTA Communication": personnel in the communication specialty

"18. JTA Navigation": personnel in the navigation specialty

"19. JTA Environmental": personnel in the environmental specialty

"20. JTA Surveillance": personnel in the surveillance specialty

In all six spreadsheets, (*B–D, 2*) identifies the top-level job task sub-area (e.g. **Raytheon - Navigation Activities, Sub-Activities, and Tasks** in "18. JTA Navigation"), while cells in (*B–D, 3*) identify subcategories of data within subsequent rows. As in the "Cur Build" spreadsheets, these top rows remain fixed to the top of Excel's interface while scrolling vertically. Category names, column locations, and cell background color are listed below. Because the contents of (*B–D, 2*) vary by sub-area (i.e., by spreadsheet), sub-area name is listed generally as **Sub-area$_i$**.

(*B, 2*), **Raytheon - Sub-area$_i$ Activities, Sub-Activities, and Tasks**: sub-area for which job task items are applicable

(*B, 3*), **Scalar**: unique identifier for a job task item (explained in Section 2.2.1)

(*C, 3*), **Activity, Sub-Activity, Task**: granularity identifier for the job task item (explained in Section 2.2.1)

(*D, 3*), **Task Statement**: granularity identifier for the job task item (explained in Section 2.2.1)

Below *3*, each row applies to one job task item, and rows are organized in a hierarchical-heterarchical way. As in Section 3.2, this organization is very similar to that of "8. Analysis" in *2013-09-16 Tech Ops_-JTA_COMMON_Part1_V02.1.xlsx* (see Section 2.2.1); however, course curriculum item verbal identifiers are not included.

3.4 Proficiency Level

This tab has expository text within (*B–D, 3–14*) explaining the ordinal scales of task knowledge and performance levels for job task items. For task performance levels, scale values are listed in (*C, 4–7*) and explanations of each level are listed in (*D, 4–7*). For task knowledge levels, scale values are listed in (*C, 11–14*) and explanations of each level are listed in (*D, 11–14*). This content aids in understanding a subset of data in "8. Analysis" within the workbook file *2013-09-16 Tech Ops_JTA_COMMON_-Part1_V02.1.xlsx*. The mappings between text in "21. Proficiency Level" and data in "8. Analysis" of *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx* are listed in outline form below. Text from an outline in "8. Analysis" (Section 2.2) is added for reference.

Content in (*B–D, 3–7*) of "21. Proficiency Level" aids in understanding data within the category:

(*AB, 3*), **Proficiency (expected output proficiency** [sic][4]: Level of performance proficiency, measured on an ordinal scale of 1–4 from least to most proficient

Content in (*B–D, 10–14*) of "21. Proficiency Level" aids in understanding data within the category:

(*AA, 3*), **Knowledge Proficiency**: level of knowledge proficiency, measured on an ordinal scale of a–d from least to most proficient

[4]There should be a closing parenthesis here, but it is omitted

3.5 DIF Criteria and Examples

This tab has expository text within (*B–E, 2–29*) explaining the meanings of **D**ifficulty, **I**mportance, and **F**requency (DIF) criteria. Each explanation maps to a subset of data in "8. Analysis" within the workbook file *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*. These mappings are listed in outline form below, and text from an outline in "8. Analysis" (Section 2.2) is added for reference.

Content in (*B–E, 3–8*) of "22. DIF Criteria and Examples" aids in understanding data within the category:

(*U, 3*), **DL**: **D**ifficulty to **L**earn

Content in (*B–E, 10–15*) of "22. DIF Criteria and Examples" aids in understanding data within the category:

(*V, 3*), **DP**: **D**ifficulty to **P**erform

Content in (*B–E, 17–22*) of "22. DIF Criteria and Examples" aids in understanding data within the category:

(*W, 3*), **I**: **I**mportance

Content in (*B–E, 24–29*) of "22. DIF Criteria and Examples" aids in understanding data within the category:

(*X, 3*), **F**: **F**requency

3.6 Delivery Options

This tab has expository text within (*B–D, 2–12*) explaining delivery options for course curriculum items. A legend in (*B–C, 14–16*) explains the meanings of facilitated (FAC), self paced (SP), and blended delivery methods. Explanations aid in understanding data within the workbook file *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*:

- "8. Analysis": data within the category **Recommended Delivery Method (SME)** (*BQ, 3*)
- "2. At a Glance Overall Common"–"7. At a Glance Overall Sur":
  - IDMs listed within (*B–D, 6–12*)
  - all cells in the category **MEDIA** (*E, 15*)

3.7 Instructional Strategies

This tab has expository text within (*B–L, 3–35*) explaining what extant theories of learning the notional course curriculum IDMs leverage. It could aid in understanding data within the category *Suggested Instructions Method Rationale* (*BQ, 3*) in "8. Analysis" of *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*.

4 2013-09-24 Tech Ops JTA Specific Part1.xlsx

This workbook is organized in the same way as *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*. It contains the same tabs having the same names, similar numbers, and analogous, non-overlapping contents (Table 4) that apply to FAA technicians based on their assigned facility and equipment. Thus, the reader should use Section 2 to aid in understanding how this workbook is organized, noting that:

- No data overlap with *2013-09-16 Tech Ops_JTA_COMMON_Part1_V02.1.xlsx*
- There are no tabs for "Overall Common," since the specific job tasks are not for all FAA technicians
- There are no sub-areas
- (*row, column*) references may not map exactly

Table 4: Spreadsheets of *2013-09-24 Tech Ops JTA Specific Part1.xlsx* by tab name and general purpose in terms of its contents. A "★" indicates that the purpose of the spreadsheet is to provide referent content: raw data, processed data, or exposition

Tab name	Spreadsheet contents		
	Raw data	Processed data	Exposition
1. Tab Description			★
2. At-a-Glance Auto		★	
3. At-a-Glance Com		★	
4. At-a-Glance Nav		★	
5. At-a-Glance Env		★	
6. At-a-Glance Sur		★	
7. Analysis	★		
20. Verbs			★
22. Initial Rating		★	
23. Refresher Rating		★	

5 2013-09-24 Tech Ops JTA Specific Part2.xlsx

This workbook is organized in the same way as *2013-09-16 Tech Ops_JTA_COMMON_Part2_V01.1.xlsx*. It contains the same tabs having the same names, similar numbers, and analogous contents (Table 5), which apply to FAA technicians based on their assigned facility and equipment. Thus, the reader should use Section 3 to aid in understanding how this workbook is organized, noting the same differences listed in Section 4.

Table 5: Spreadsheets of *2013-09-24 Tech Ops JTA Specific Part2.xlsx* by tab name and general purpose in terms of its contents. A "★" indicates that the purpose of the spreadsheet is to provide referent content: raw data or exposition

Tab name	Spreadsheet contents	
	Raw data	Exposition
1. Tab Description		★
8. Cur Build Auto	★	
9. Cur Build Com	★	
10. Cur Build Nav	★	
11. Cur Build Env	★	
12. Cur Build Sur	★	
13. JTA Automation	★	
14. JTA Communication	★	
15. JTA Navigation	★	
16. JTA Environmental	★	
17. JTA Surveillance	★	
18. Proficiency Level		★
19. DIF Criteria and Examples		★
20. Delivery Options		★
21. Instructional Strategies		★

A Color Reference

Cell background color shown in this document	RGB value (`red, green, blue`)	
	Apple Retina display	Most other displays
	(255, 255, 255)	(255, 255, 255)
	(229, 184, 184)	(280, 184, 183)
	(190, 7, 18)	(192, 0, 0)
	(224, 107, 33)	(228, 108, 10)
	(233, 170, 79)	(255, 128, 128)
	(248, 178, 124)	(250, 191, 143)
	(148, 137, 87)	(148, 138, 84)
	(248, 198, 46)	(255, 204, 0)
	(251, 241, 54)	(255, 255, 0)
	(253, 252, 157)	(255, 255, 153)
	(204, 142, 142)	(203, 244, 136)
	(42, 178, 38)	(0, 255, 0)
	(130, 201, 63)	(146, 208, 80)
	(205, 254, 206)	(204, 255, 204)
	(62, 204, 203)	(51, 204, 204)
	(155, 205, 243)	(146, 205, 220)
	(28, 172, 231)	(0, 171, 234)
	(86, 142, 211)	(85, 142, 213)
	(24, 55, 91)	(23, 55, 94)
	(96, 74, 121)	(96, 74, 123)
	(203, 156, 252)	(204, 153, 255)