Information Storage and Retrieval

Collection Management of Tobacco Settlement Documents

Authors

Alon Bendelac
Debasmita Biswas
Sushmethaa Muhundan
Andrei Svetovidov
Ashin Marin Thomas
Yan Zhao

Instructor

Dr. Edward A. Fox

Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
December 21, 2019

CS5604: Information Storage and Retrieval, Fall 2019

Team CMT: Alon Bendelac, Sushmethaa Muhundan, Debasmita Biswas, Andrei Svetovidov, Ashin Marin Thomas, and Yan Zhao

This research was done under the supervision of Dr. Edward A. Fox as part of the course CS5604: Information Storage and Retrieval at Virginia Tech, Fall 2019.

1st edition, September 19, 2019

2nd edition, October 11, 2019

3rd edition, November 1, 2019

4th edition, December 11, 2019

5th edition, January 8, 2020

Contents

Abstract

Consumption of tobacco causes health issues, both mental and physical. Despite this widely known fact, tobacco companies sustained their huge presence in the market over the past century owing to a variety of successful marketing strategies. This report documents the work of the Collection Management Tobacco Settlement Documents (CMT) team, the data ingestion team for the tobacco documents. We deal with an archive of tobacco documents that were produced during litigation involving 39 of the states in the USA, and seven major tobacco industry organizations. Our aim is to process these documents and assist Dr. David M. Townsend, an assistant professor at Virginia Polytechnic Institute and State University (Virginia Tech) Pamplin College of Business, in his research towards understanding the marketing strategies of the tobacco companies. The team is part of a larger initiative: to build a state-of-the-art information retrieval and analysis system.

We handle over 14 million tobacco settlement documents as part of this project. Our tasks include extracting the data as well as metadata from these documents. We cater to the needs of the ElasticSearch (ELS) team and the Text Analytics and Machine Learning (TML) team. We provide tobacco settlement data in suitable formats to enable them to process and feed the data into the information retrieval system.

We have successfully processed both the metadata and the document texts into a usable format. For metadata, this involved collaborating with the above-mentioned teams to come up with a suitable format. We retrieved the metadata from a MySQL database and converted it into a JSON for ElasticSearch ingestion. For the data, this involved lemmatization, tokenization, and text cleaning.

We have supplied the entire dataset to the ELS and TML teams. Data, as well as metadata of these documents, were cleaned and provided. Python scripts were used to query the database and output the results in the required format.

We also closely interacted with Dr. Townsend to understand his research needs in order to guide the Front-end and Kibana (FEK) team in terms of insights about features that can be used for visualizations. This way, the information retrieval system we build would add more value to our client.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Objective

The objective of this project is to build a state-of-the-art information retrieval system. Our team's role is to process tobacco settlement documents and organize them in a way that would be suitable for ingestion into the system. Then they would feed as input to the information retrieval system. This project aims to provide a suitable interface to query tobacco settlement documents to aid Dr. Townsend's research.

The source of documents is UCSF's Industry Documents Library (IDL). The IDL hosts documents pertaining to 5 different industries, namely tobacco, drug, chemical, food, and fossil fuel. Through the course of this project, we utilize the 14 million tobacco settlement documents housed by IDL. These documents were created by tobacco companies and come in varying formats like depositions, memos, company letters, etc.

The aim is to organize the documents, parse them, and provide metadata along with the data to the ElasticSearch (ELS) and the Text Analytics and Machine Learning (TML) teams.

We are working on the data ingestion pipeline, where we load the entire data, process all 14 million documents, and provide relevant data that can be processed by the ELS and front-end teams to provide the final product: an information retrieval system that can be used to query the tobacco dataset.

Our aim is to ensure that relevant and correct data is being returned while querying. We have worked closely with the above-mentioned teams to ensure that this happens.

1.2 Client

The client of this project is Dr. David M. Townsend, an assistant professor in the Virginia Polytechnic Institute and State University (Virginia Tech) Pamplin College of Business. Dr. Townsend's research focuses on entrepreneurship, strategic management, and technology management. He is working on analyzing the documents publicly released by tobacco companies as a result of a number of lawsuits launched against them. As an active user of the document library, Dr. Townsend is personally interested in having a specialized information retrieval system that could assist him in his research endeavors.

1.3 Project Management

To provide effective communication between team members and representatives, we use Slack, an easy-to-use chat system with a user-friendly interface, which allows for creating and maintaining chat groups for instant sharing of updates and suggestions both within the team and among participants from different teams. This was extremely useful to collaborate with members of the ELS and TML teams. We were able to work closely with them to understand their requirements and deliver the requirements in a timely manner.

To effectively distribute tasks and track progress, we use Trello, a task management application that gives a visual overview of the progress made.

1.4 Challenges

There were a number of challenges faced during the course of this project. One challenge was the nature of the data. We started with 14 million documents. There was little documentation regarding the structure of the data. We initially used MySQL queries and a sample set of data to understand the attributes and metadata. We also reached out to the technical support team of UCSF library and they were extremely helpful. We were then able to fully understand the structure of data and their attributes.

Another challenge we faced was that we were unable to distribute the tasks among team members in a timely manner, given a range of levels of technical and programming expertise amongst the participants. Notwithstanding, we constantly worked on effectively parallelizing work, addressing critical questions, and supporting each other in leveraging appropriate skills.

Chapter 2

Literature Review

The prescribed textbook *"An Introduction to Information Retrieval"* [4] has been immensely useful during the course of this project. The book helped us understand the basics of tokenization and lemmatization, key steps in pre-processing.

2.1 Overview and Expectations

We closely interacted with Dr. Townsend to understand his requirements. His research focuses on understanding the organizational strategies and corporate tactics utilized by tobacco companies during the second half of the 20th century to fight through the tobacco settlement cases. The aim of his research is to construct a timeline and uncover the most prominent characters and analyze the roles companies have played in these cases. Dr. Townsend was expecting an easily navigable search engine that would return noise-free results, i.e., an interface that facilitates mass retrieval of files based on certain criteria as opposed to them being fetched individually. Examples include downloading all documents on a page filtered by conditions, and seeing results that bundle documents around a certain case. It should also be able to identify cases with a common witness. During subsequent meetings with Dr. Townsend, he expected the teams to provide him with some features in the user interface provided by the front-end team. This included export functionality and visualization for the tobacco documents showing the data features and the document clusters.

2.2 Tobacco Settlement Documents

UCSF's Industry Documents Library (IDL) hosts an archive of 14 million documents created by tobacco companies about their advertising, manufacturing, marketing, scientific research, and political activities [13]. Truth Tobacco Industry Documents were created in 2002 by the UCSF Library. The main intention was to provide access to the tobacco industry documents produced during litigation involving 39 states and the seven major tobacco industry organizations. These documents come in various types and could be any of the following: company memos, letters, depositions, etc. The system we build needs to cater to all these variants.

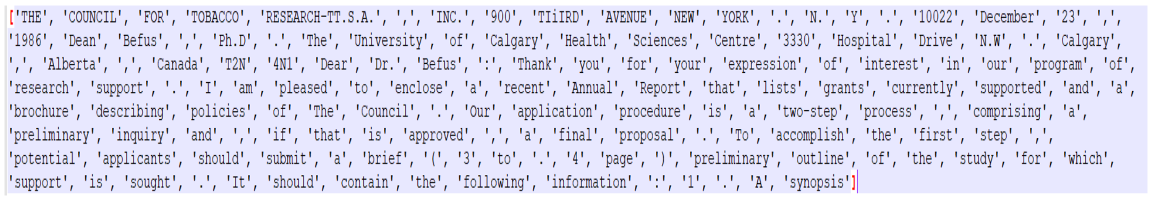
UCSF also provides a list of exhaustive field names [12] that depicts the metadata of the documents. This is being used as a reference point, and we are using this to parse and get relevant information out of the data since it is not well-documented.

The tobacco settlement document report [8] by the team that worked on this project for CS 4624 (Multimedia, Hypertext, and Information Access) during the spring 2019 semester, has also been a point of reference. Previous work in this area had been done in June 2019 and we have been studying the team reports and presentations as well, in order to get a better understanding of the effort in this space. It contains information about various methods incorporated, such as creating scripts for parsing the various documents and coming up with the metadata and doc2vec model implementation.

We had used these resources as a starting point for analyzing the data to come up with a proper knowledge base for the metadata and text extraction from the 14M documents.

2.3 Tokenization and Lemmatization

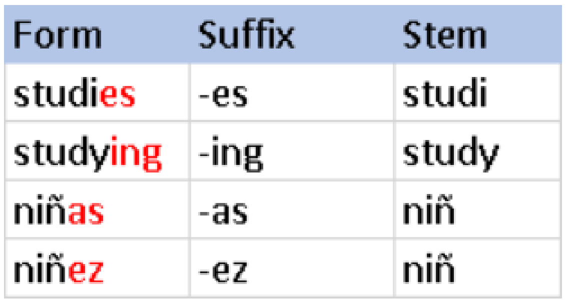
Tokenization refers to chopping a given sentence or document unit into pieces, called tokens, perhaps throwing away certain characters such as punctuation. A token is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing. Tokenization accomplishes a very crucial task in pre-processing, to enable the handling of semantic issues in subsequent stages of machine processing. It also contributes to a structural description of an input sentence. The tokens become the input for other processes like parsing and text mining. Tokenization plays a large part in the process of lexical analysis [10]. Figure 2.1 shows the result of the tokenization undertaken for a sample deposition file.



```
['THE', 'COUNCIL', 'FOR', 'TOBACCO', 'RESEARCH-TT.S.A.', ',', 'INC.', '900', 'TIiIRD', 'AVENUE', 'NEW', 'YORK', '.', 'N.', 'Y', '.', '10022', 'December', '23', ',',
'1986', 'Dean', 'Befus', ',', 'Ph.D', '.', 'The', 'University', 'of', 'Calgary', 'Health', 'Sciences', 'Centre', '3330', 'Hospital', 'Drive', 'N.W', '.', 'Calgary',
',', 'Alberta', ',', 'Canada', 'T2N', '4N1', 'Dear', 'Dr.', 'Befus', ':', 'Thank', 'you', 'for', 'your', 'expression', 'of', 'interest', 'in', 'our', 'program', 'of',
'research', 'support', '.', 'I', 'am', 'pleased', 'to', 'enclose', 'a', 'recent', 'Annual', 'Report', 'that', 'lists', 'grants', 'currently', 'supported', 'and', 'a',
'brochure', 'describing', 'policies', 'of', 'The', 'Council', '.', 'Our', 'application', 'procedure', 'is', 'a', 'two-step', 'process', ',', 'comprising', 'a',
'preliminary', 'inquiry', 'and', ',', 'if', 'that', 'is', 'approved', ',', 'a', 'final', 'proposal', '.', 'To', 'accomplish', 'the', 'first', 'step', ',',
'potential', 'applicants', 'should', 'submit', 'a', 'brief', '(', '3', 'to', '.', '4', 'page', ')', 'preliminary', 'outline', 'of', 'the', 'study', 'for', 'which',
'support', 'is', 'sought', '.', 'It', 'should', 'contain', 'the', 'following', 'information', ':', '1', '.', 'A', 'synopsis']
```

Figure 2.1: Tokens of fghb0000.ocr

For grammatical reasons, documents tend to use different forms of a word, such as *organize*, *organizes*, and *organizing*. Additionally, there are families of derivationally related words with similar meanings, such as *democracy*, *democratic*, and *democratization*. In many situations, it would be useful for a search for one of these words to return documents that contain another word in the set. Stemming and lemmatization refer to reducing inflectional forms and sometimes derivationally related forms of a word to a common base form [4]. The difference between stemming and lemmatization is that a stem might not be an actual word, whereas a lemma is an actual language word or root form. Stemming follows an algorithm with steps to perform on the words, which makes it faster. Figure 2.2 shows the result of the stemming on a sample set.

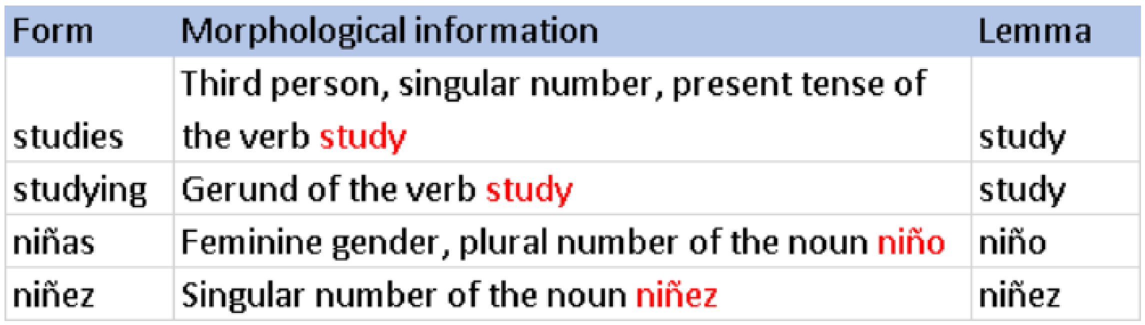


Form	Suffix	Stem
studies	-es	studi
studying	-ing	study
niñas	-as	niñ
niñez	-ez	niñ

Figure 2.2: Stemming sample[3]

In contrast, lemmatization uses a lexicon, such as in the WordNet corpus, to produce lemma, which makes it slower than stemming. If we are building a language application in which language is important one should use lemmatization as it uses a corpus to match root forms [6]. If confronted with the token *saw*, poor stemming might return just *s*, whereas lemmatization would attempt to return either *see* or *saw* depending on whether the use of the token was as a verb or a noun.

Lemmatization takes into consideration the morphological analysis of the words. To do so, it is necessary to have detailed dictionaries that the algorithm can look through to link the form back to its lemma [3]. Lemmatization is used in text mining, which involves analyzing the texts written in natural language, and extracting high-quality information. Text mining includes text categorization, text clustering, sentiment analysis, document summarization, and entity relation modeling (i.e., learning relations between named entities). For document clustering, lemmatization reduces the number of tokens that contain the same information and hence speeds up the whole process. Figure 2.3 shows the result of the lemmatization on a sample set.



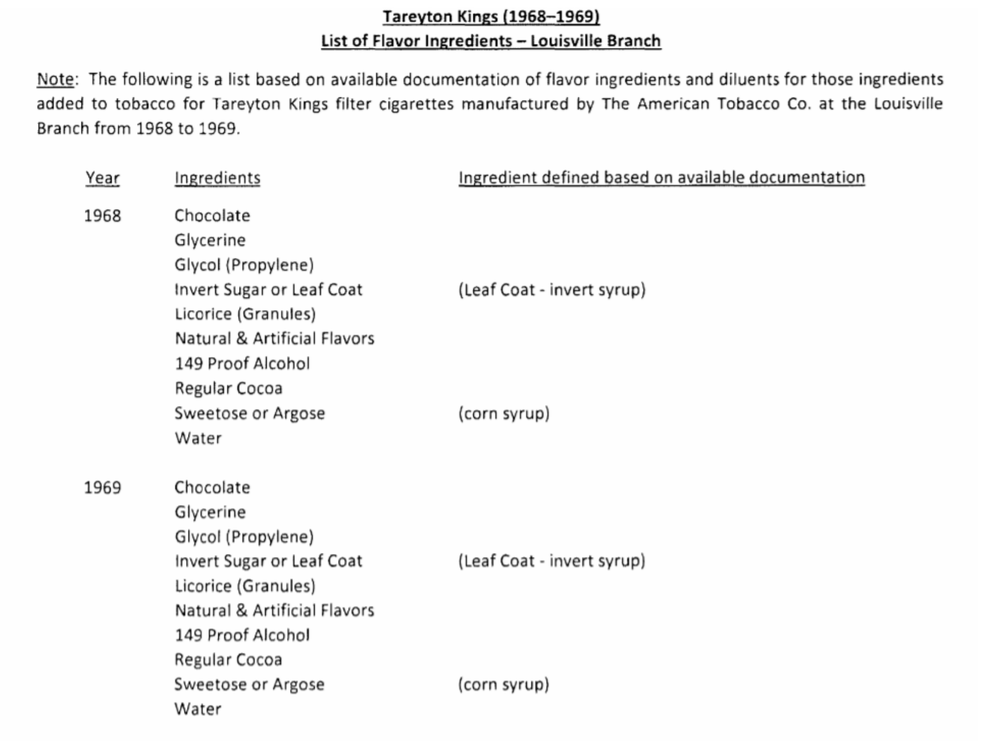
Form	Morphological information	Lemma
studies	Third person, singular number, present tense of the verb study	study
studying	Gerund of the verb study	study
niñas	Feminine gender, plural number of the noun niño	niño
niñez	Singular number of the noun niñez	niñez

Figure 2.3: Lemmatization sample[3]

Examples of lemmatization include:
am, are, is => be
car, cars, car's, cars' => car
been had done languages cities mice => be have do language city mouse

2.4 PDF Processing Techniques

PDF is one of the most important and widely used digital formats, used to present and exchange documents. The tobacco dataset thus is shared as 14M PDF files. The current dataset also contains OCR files, but the existing ones have garbage characters and line numbers included. Some .ocr files have text in the format of JSON data. Hence, we explored many PDF processing techniques mentioned below to improve the PDF processing step. Figure 2.4 is the original PDF file: jydp0228.pdf that has been processed by the following techniques.


Tareyton Kings (1968–1969)
List of Flavor Ingredients – Louisville Branch

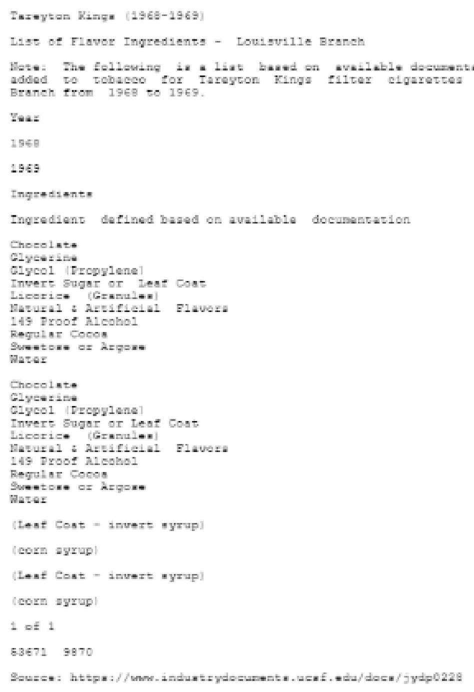
Note: The following is a list based on available documentation of flavor ingredients and diluents for those ingredients added to tobacco for Tareyton Kings filter cigarettes manufactured by The American Tobacco Co. at the Louisville Branch from 1968 to 1969.

Year	Ingredients	Ingredient defined based on available documentation
1968	Chocolate	
	Glycerine	
	Glycol (Propylene)	
	Invert Sugar or Leaf Coat	(Leaf Coat - invert syrup)
	Licorice (Granules)	
	Natural & Artificial Flavors	
	149 Proof Alcohol	
	Regular Cocoa	
	Sweetose or Argose	(corn syrup)
	Water	
1969	Chocolate	
	Glycerine	
	Glycol (Propylene)	
	Invert Sugar or Leaf Coat	(Leaf Coat - invert syrup)
	Licorice (Granules)	
	Natural & Artificial Flavors	
	149 Proof Alcohol	
	Regular Cocoa	
	Sweetose or Argose	(corn syrup)
	Water	

Figure 2.4: Sample scanned document fetched from the UCSF site: jydp0228.pdf

2.4.1 PDFMiner

PDFMiner focuses entirely on getting and analyzing text data from PDF documents. It allows obtaining the exact location of text on a page, as well as other information such as fonts or lines. It includes a PDF converter that can transform PDF files into other text formats (such as HTML) [7]. This method tries to maintain the data format, for example, tabular format, but data in different columns gets merged into a single column. It also takes into consideration the structure of the data; this is shown in Figure 2.5.

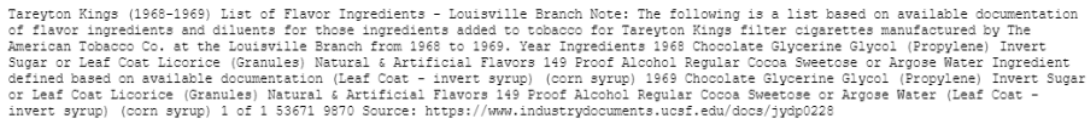


```
Tareyton Kings (1968-1969)

List of Flavor Ingredients -  Louisville Branch

Note:  The following  is a list  based on  available documentation  of flavor  ingredients and diluents for those  ingredients
added  to  tobacco  for  Tareyton  Kings  filter  cigarettes  manufactured  by  The  American  Tobacco  Co. at  the  Louisville
Branch from  1968 to 1969.

Year

1968

1969

Ingredients

Ingredient  defined based on available  documentation

Chocolate
Glycerine
Glycol (Propylene)
Invert Sugar or  Leaf Coat
Licorice  (Granules)
Natural & Artificial  Flavors
149 Proof Alcohol
Regular Cocoa
Sweetose or Argose
Water

Chocolate
Glycerine
Glycol (Propylene)
Invert Sugar or Leaf Coat
Licorice  (Granules)
Natural & Artificial  Flavors
149 Proof Alcohol
Regular Cocoa
Sweetose or Argose
Water

(Leaf Coat - invert syrup)

(corn syrup)

(Leaf Coat - invert syrup)

(corn syrup)

1 of 1

53671  9870

Source: https://www.industrydocuments.ucsf.edu/docs/jydp0228
```

Figure 2.5: Sample scanned document processed by PDFMiner

2.4.2 PyPDF2

PyPDF2 is capable of splitting, merging together, cropping, and transforming the pages of PDF files. It can also add custom data, viewing options, and passwords to PDF files. It can retrieve text and metadata from PDFs as well as merge entire files together [7]. This technique extracts just the text from the PDF but does not maintain the format of the data, for example, tabular format. It does not take into consideration the field structure, for example, the title is merged with the rest of the text as shown in Figure 2.6.

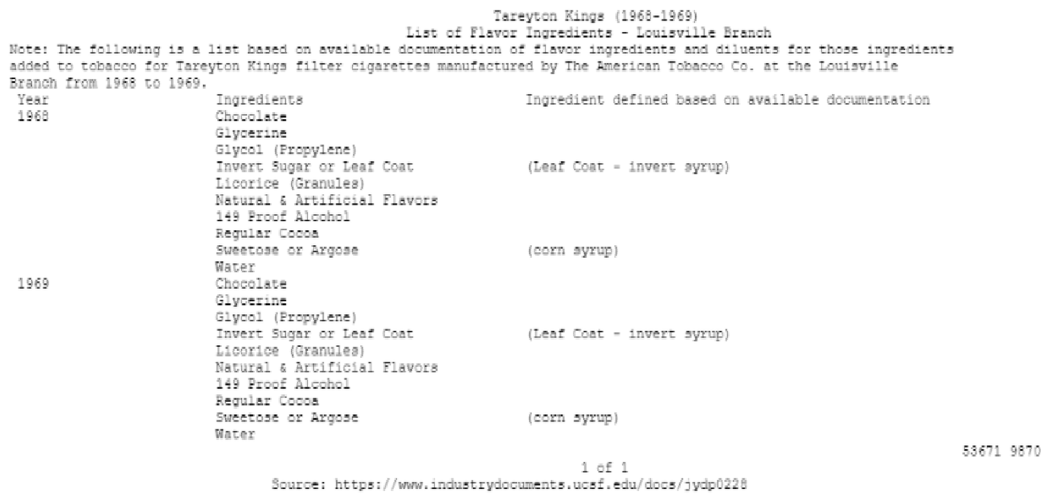


```
Tareyton Kings (1968-1969) List of Flavor Ingredients - Louisville Branch Note: The following is a list based on available documentation
of flavor ingredients and diluents for those ingredients added to tobacco for Tareyton Kings filter cigarettes manufactured by The
American Tobacco Co. at the Louisville Branch from 1968 to 1969. Year Ingredients 1968 Chocolate Glycerine Glycol (Propylene) Invert
Sugar or Leaf Coat Licorice (Granules) Natural & Artificial Flavors 149 Proof Alcohol Regular Cocoa Sweetose or Argose Water Ingredient
defined based on available documentation (Leaf Coat - invert syrup) (corn syrup) 1969 Chocolate Glycerine Glycol (Propylene) Invert Sugar
or Leaf Coat Licorice (Granules) Natural & Artificial Flavors 149 Proof Alcohol Regular Cocoa Sweetose or Argose Water (Leaf Coat -
invert syrup) (corn syrup) 1 of 1 53671 9870 Source: https://www.industrydocuments.ucsf.edu/docs/jydp0228
```

Figure 2.6: Sample scanned document processed by PyPDF2

2.4.3 Abbyy Cloud OCR SDK

It is a web-based AI-powered cloud OCR SDK (Software Development Kit) which provides excellent text recognition, PDF conversion, and data capture functionalities, enabling it to convert scans into searchable PDF documents [1]. It maintains the data format and structure as shown in Figure 2.7.



```
                                          Tareyton Kings (1968-1969)
                                  List of Flavor Ingredients - Louisville Branch
Note: The following is a list based on available documentation of flavor ingredients and diluents for those ingredients
added to tobacco for Tareyton Kings filter cigarettes manufactured by The American Tobacco Co. at the Louisville
Branch from 1968 to 1969.
 Year                  Ingredients                        Ingredient defined based on available documentation
 1968                  Chocolate
                       Glycerine
                       Glycol (Propylene)
                       Invert Sugar or Leaf Coat          (Leaf Coat - invert syrup)
                       Licorice (Granules)
                       Natural & Artificial Flavors
                       149 Proof Alcohol
                       Regular Cocoa
                       Sweetose or Argose                 (corn syrup)
                       Water
 1969                  Chocolate
                       Glycerine
                       Glycol (Propylene)
                       Invert Sugar or Leaf Coat          (Leaf Coat - invert syrup)
                       Licorice (Granules)
                       Natural & Artificial Flavors
                       149 Proof Alcohol
                       Regular Cocoa
                       Sweetose or Argose                 (corn syrup)
                       Water
                                                                                                       53671 9870
                                                  1 of 1
                           Source: https://www.industrydocuments.ucsf.edu/docs/jydp0228
```

Figure 2.7: Sample scanned document processed by Abbyy Cloud OCR SDK

Table 2.1 summarizes the performance of the above mentioned OCR methods for different document types.

Table 2.1: OCR methods comparison

	UCSF	**PyPDF2**	**PDFMiner**	**Abby Cloud OCR**
Two Column PDF’s	Poor to Average (comparable to pdfminer)	Poor	Average (comparable to UCSF)	Good
Newspaper Articles	Very poor	Extremely poor / No result	Very poor	Average
Tables	Average (comparable to pdfminer)	Poor	Average (comparable to UCSF)	Excellent
Plain texts, letters	Excellent	Good	Excellent	Excellent
Handwritten Texts	Very Poor	Extremely poor	Very Poor	Poor to Average

Chapter 3

Requirements

To provide the teams working on ingesting data into Elasticsearch with a certain subset (Phase I) and a complete set (Phase II) of text data with accompanying metadata information, the following requirements and conditions should be taken into consideration.

1. **Text data from the documents should be provided in its entirety and with completeness.**

   In other words, all the text information contained in processed documents is supposed to be saved and transferred to the other teams without any loss, unless otherwise indicated. In Phase I of the project, only documents of the "Deposition" type were considered. Therefore, this needed to be extended to support the other document types as well. Phase II of the project involved processing all the documents from the database.

2. **Only requested meta-information fields are to be provided for further processing.**

   This means that the final metadata file should only comprise two sets of fields: the ones which are referred to by the ELS team and necessary for the search engine, and the fields that are not involved in the search process directly but can be used by the FEK team for visualization and demonstration purposes.

3. **The metadata files should follow the format requirements.**

   Depending on what restrictions the ELS team imposes on the format of the to-be-ingested files, it was our responsibility to follow these restrictions. For example, certain metadata files, describing a similar topic or several related ones (such as articles, questionnaires, etc.), might be incorporated into a single file for potential improvement of search engine capabilities.

4. **The CMT team should communicate with the ELS team and other teams on a regular basis.**

   This was necessary in order to understand current updates in constraints imposed on the file structure and size so that the CMT team is able to follow the 2nd and 3rd requirements. This was also necessary to provide other groups with updates regarding our progress.

5. **The CMT team must pre-process file content for the TML team appropriately**

   As various intelligent text analysis algorithms require data to be ingested in different formats, we also provided the TML team with full-text data in desired formats. The reader may find further details in Section 5.3.2.

6. **Potential storage of the data on cloud.cs.vt.edu for future project organization.**

   Although it is not required to store all the documents and metadata tables in the cloud storage for the purpose of pre-processing, this could be a potential direction for the sake of persisting files to be utilized by future project contributors.

Chapter 4

Design

4.1 Approach

We broke down our research objectives into the following research questions:

RQ1: *How do we structure the metadata in a format that ElasticSearch can ingest?*

RQ2: *What text pre-processing should be done on the OCR'ed documents to make them useful for the TML team?*

RQ3: *How does UCSF's proprietary OCR tool compare with open-source OCR tools on a subset of the tobacco documents?*

We initially started by investigating these research questions on a subset of the tobacco documents. More specifically, we used the roughly 8,000 deposition documents in this work. We later expanded the scope to cover all 14 million documents.

In order to be able to provide other teams with the necessary information in the required format, the following steps were followed:

1. Download data from the UCSF database onto the tobacco.cs.vt.edu virtual machine, including files and corresponding metadata. This is illustrated in Figure 4.1.

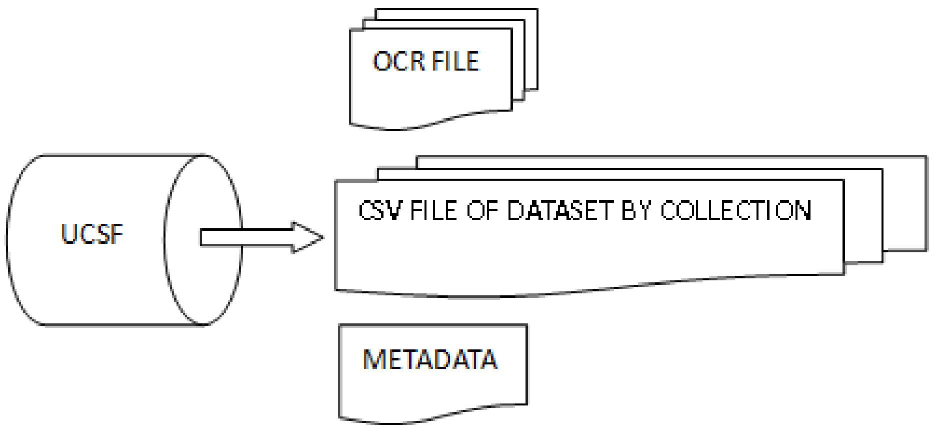


Figure 4.1: Files downloaded from the UCSF site

2. Import metadata into a database and analyze the structure of data.

3. Process a small collection of data first. Specifically, locate deposition files based on file type, then process the deposition files.

4. Convert metadata and files into JSON format, as shown in Figure 4.2.

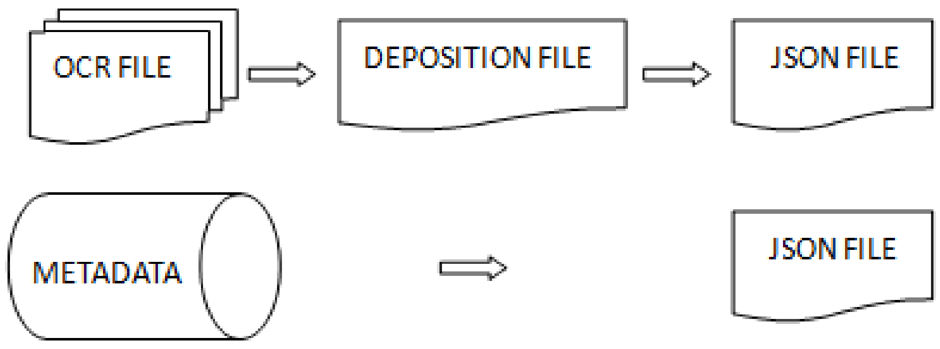


Figure 4.2: Process data to be formatted

5. Pre-process texts of deposition files, as shown in Figure 4.3. This involved the following steps:

- Text cleaning: Delete the invalid characters which can not be decoded to Unicode. Delete the blank characters, like CRLF line terminators, and so on.
- Tokenization: Split the given text into smaller pieces. Words, numbers, punctuation marks, and others are considered as tokens.
- Lemmatization: Reduce inflectional forms to get the correct base forms of words by using lexical knowledge. The aforementioned processes are described in Chapter 5 in detail.
- Concatenation of Q's and A's in one line for easy reading. This process is described in Chapter 5 in detail.

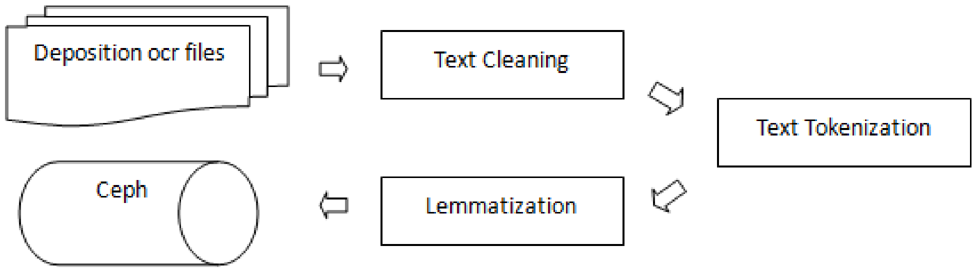


Figure 4.3: Text preprocessing

6. Provide the relevant JSON files to the ElasticSearch team.
7. Migrate preprocessed deposition files and JSON files to Ceph.

The next step was to apply the above-stated process to all types of documents. We have extracted the metadata and content for all the documents and sent them to the ELS team for processing, and to the TML team for summarization.

4.2 Tools

- MariaDB: MariaDB Server, with its continual open source innovation, is a modern relational database. We store and view all the metadata of files in MariaDB 5.5.

- Python: We used Python 3 to develop scripts to process files, including JSON packages, Codec packages, and so on.

- NLTK: Natural Language ToolKit[9] is a leading platform for building Python programs to work with human language data. We used NLTK to do text tokenization and lemmatization, and to provide high-quality text to other teams.

Chapter 5

Implementation

To manage our work, we identified three major tasks:

1. Identifying different document types
2. Processing metadata
3. Processing document text

These tasks are described in detail below. In order to achieve our goals in a reasonable amount of time, along with preparing and executing MySQL queries, we focused on developing code in Python because of its popularity and ease-of-use. We started by working with only the deposition documents, and then we transitioned to working with all document types.

5.1 Identifying Different Document Types

The tobacco documents within the dataset are categorized by type, such as memo, letter, and deposition. Some of the documents belong to several types. The information about each document's type is encoded in the MySQL database (further details can be found in Section 7.1). We identified different document types and decided to process deposition documents. The number of depositions documents is relatively small: 7995 files overall as opposed to the entire collection size (about 14 million files). We developed a Python script to produce a file containing the record keys (unique identifiers) of all the deposition documents. The record keys were then used to retrieve the deposition files from the database directly, as the record keys correspond to file names.

5.2 Processing Metadata

The objective of the metadata processing task is to transform the metadata from a MySQL database to a JSON file that ElasticSearch can ingest. Because ElasticSearch could not ingest the metadata of all 14 million documents in one file, we have transformed the metadata into a collection of 31 files, each containing the metadata of at most half a million documents.

As shown in Figure 5.1, we encode a document's metadata as two JSON objects: a header and a body. The header object identifies the document with its unique ID. The body object contains the document's metadata as a collection of key-value pairs, where the key identifies the metadata field, and the value specifies the corresponding value for the given document.

Figure 5.1: JSON object representing metadata

```
{"index": {"_id": 1, "_index": "tobacco"}}
{"url": "https://s3-us-west-2.amazonaws.com/edu.ucsf.library.iddl.
 artifacts/f/z/w/d/fzwd0000/fzwd0000.pdf",
 "Legacy_(LTDL2)_Tobacco_Id": "gbu00a00",
 "Title": "RISING MEDICAL COSTS REQUIRE G.H.C. DUES INCREASE IN 1967
  ↪  VIEW",
 "Document_Date": "1967-02-28 00:00:00",
 "Author": "NEWMAN HF;SIEGAL A",
 "Case": "MNAG",
 "Description": "DISCUSSES DUES INCREASE",
 "Date_Added_UCSF": "2002-02-01 00:00:00",
 "Document_Type": "article",
 "availablility": "public",
 "availablilitystatus": "no restrictions",
 "Mentioned": "GROUP HEALTH COOPERATIVE OF PUGET SOUND;...
  "Attached_Artifacts":
    [{"name":"fzwd0000.ocr",
          "mediaType":"text/plain",
          "size":7730},
         {"name":"fzwd0000.pdf",
          "mediaType":"application/pdf",
          "size":308035},
         {"name":"fzwd0000.tif",
          "mediaType":"image/tiff",
          "size":314038},
         {"name":"fzwd0000_thumb.png",
          "mediaType":"image/png",
          "size":100478}],
   "Page_Map": "60025743/5745",
   "Numeric_start_bates": "60025743",
   "Numeric_end_bates": "60025745",
   "Page_Count": "3"
}
```

The following steps were applied to transform metadata into the required JSON format. First, connect to the MySQL database in order to retrieve metadata. Next, execute a MySQL query to retrieve all metadata entries, ordered by document ID, with document IDs between the given minimum and maximum ID values. Then, retrieve the result of this query. The result is a list of metadata entries, each consisting of a document ID, a metadata field name, and a metadata value. Extract the metadata entries for each document in an iterative fashion. Generate the header JSON object followed by the body obejct for each document.

We later included a text field, which is provided in a page-wise fashion: an additional key "text content" consists of two key-value pairs, where the keys are "content" and "page". This would facilitate full-text search by the search engine.

5.3 Processing Document Text

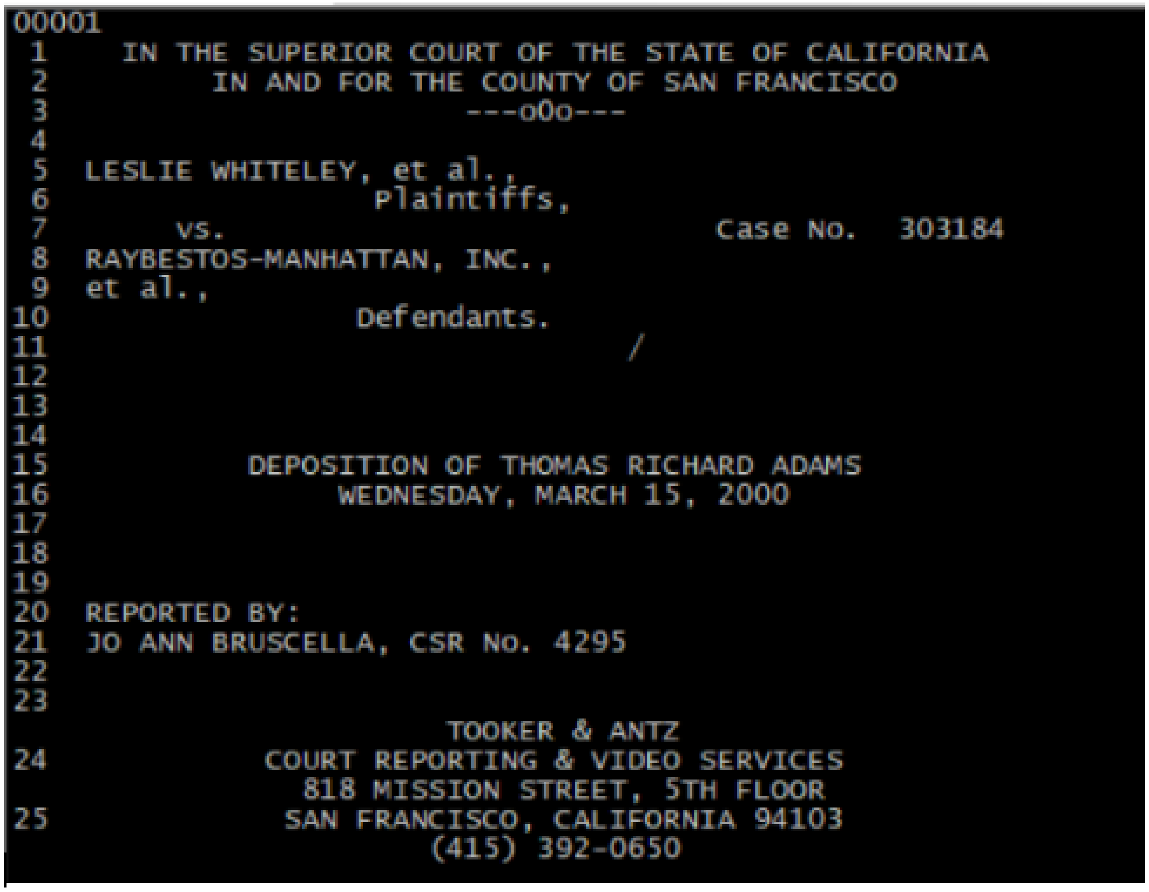
The task of processing the content of the documents can be divided into 2 parts based on the requests from the teams:

1. Preparing documents for the ELS and FEK teams
2. Pre-processing documents for applying of intelligent algorithms for TML team

These sub-tasks are described in Sections 5.3.1 and 5.3.2.

5.3.1 Preparing document text for the ElasticSearch and Front-End Kibana Teams

All the files stored in the virtual machine have a .ocr format. In order to make them digestible by the ElasticSearch engine, we developed and subsequently updated a Python script to convert each of the deposition .ocr files into a JSON file. The typical structure of a deposition file is shown in Figure 5.2.



```
00001
     IN THE SUPERIOR COURT OF THE STATE OF CALIFORNIA
          IN AND FOR THE COUNTY OF SAN FRANCISCO
                        ---o0o---

 LESLIE WHITELEY, et al.,
                   Plaintiffs,
      vs.                               Case No.  303184
 RAYBESTOS-MANHATTAN, INC.,
 et al.,
                  Defendants.
                              /



          DEPOSITION OF THOMAS RICHARD ADAMS
              WEDNESDAY, MARCH 15, 2000




 REPORTED BY:
 JO ANN BRUSCELLA, CSR No. 4295


                    TOOKER & ANTZ
            COURT REPORTING & VIDEO SERVICES
              818 MISSION STREET, 5TH FLOOR
             SAN FRANCISCO, CALIFORNIA 94103
                   (415) 392-0650
```

Figure 5.2: The structure of the first page of the deposition document

We structured the data to extract page contents separately as shown below:

```
{
"Page No." : "Page content"
}
```

The structure of the final file generated by the Python script is shown below:

```
{"index": {"_id": 0, "_index": "tobacco"}}
{"Title": "Deposition of THOMAS RICHARD ADAMS, March 15, 2000, WHITELEY
↪  v. RAYBESTOS-MANHATTAN INC.",
"Document_Date": "2000-03-15",
"text_content":[{"page":"1","content":"IN THE SUPERIOR COURT OF THE
↪  STATE OF CALIFORNIA"},
{"page": "2","content":"DEPOSITION OF THOMAS RICHARD ADAM"}]}
```

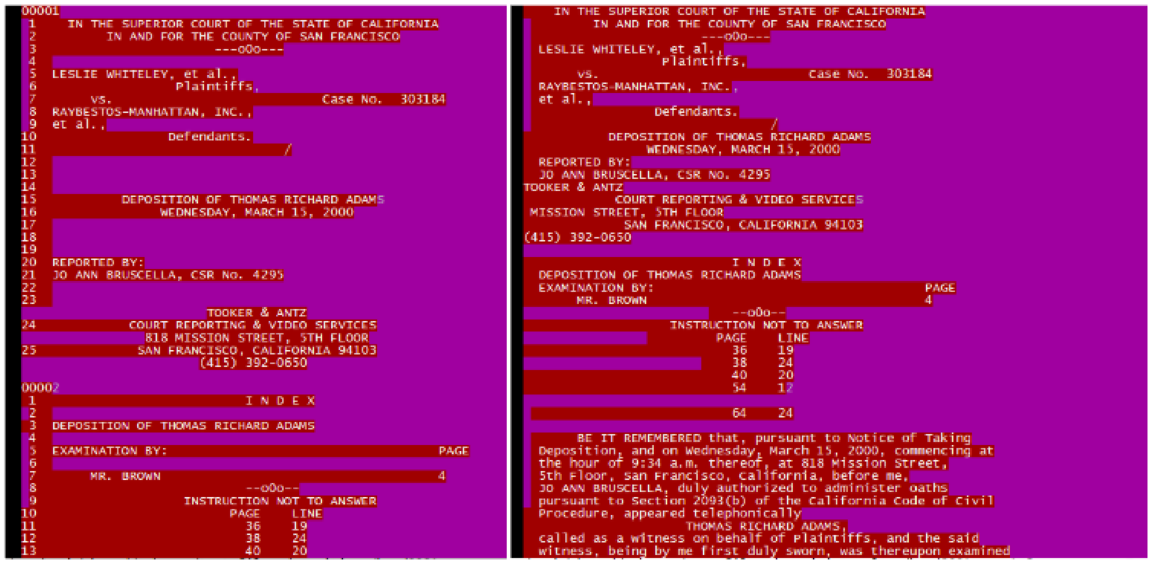
Besides the text itself, the file represents a page-wise distribution of the content, as well as the document date, as requested by the ELS and FEK teams.

5.3.2 Pre-processing documents for text analytics and machine learning (TML)

To ensure the correct operation of different automatic text analysis and ML approaches, we follow the specific requirements for the document text preprocessing provided by the TML team. The tasks include the following steps:

1. Cleaning documents

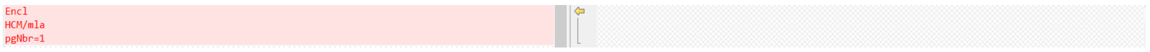
   The documents stored in the virtual machine were received as a result of preliminary performing of the Optical Character Recognition (OCR) procedure. A common issue about automatic extraction of characters from document scans is that the latter may contain a lot of specialized information, such as line numbers and handwritten inscriptions. Although some of them might be valuable for retrieval purposes, a significant portion of these characters and strings can hinder the performance of ML algorithms, so cleaning the documents is important. Moreover, the recognizing algorithm may produce service symbols, which can become additional noise for ML algorithms. In order to reduce the negative impact of such characters, we prepared a cleaning algorithm in Python and used it to pre-process a set of deposition documents. The examples of the comparative file contents (before and after the pre-processing accordingly) are demonstrated in Figures 5.3, 5.4, and 5.5.



```
00001
 1     IN THE SUPERIOR COURT OF THE STATE OF CALIFORNIA
 2          IN AND FOR THE COUNTY OF SAN FRANCISCO
 3                        ---oOo---
 4
 5   LESLIE WHITELEY, et al.,
 6                  Plaintiffs,
 7        vs.                                 Case No.  303184
 8   RAYBESTOS-MANHATTAN, INC.,
 9   et al.,
10                 Defendants.
11                                  /
12
13
14
15              DEPOSITION OF THOMAS RICHARD ADAMS
16                   WEDNESDAY, MARCH 15, 2000
17
18
19
20   REPORTED BY:
21   JO ANN BRUSCELLA, CSR No. 4295
22
23
                         TOOKER & ANTZ
24               COURT REPORTING & VIDEO SERVICES
                   818 MISSION STREET, 5TH FLOOR
25                SAN FRANCISCO, CALIFORNIA 94103
                         (415) 392-0650

00002
 1                             I N D E X
 2
 3   DEPOSITION OF THOMAS RICHARD ADAMS
 4
 5   EXAMINATION BY:                                         PAGE
 6
 7        MR. BROWN                                             4
 8                             --oOo--
 9                   INSTRUCTION NOT TO ANSWER
10                           PAGE    LINE
11                             36    19
12                             38    24
13                             40    20
```

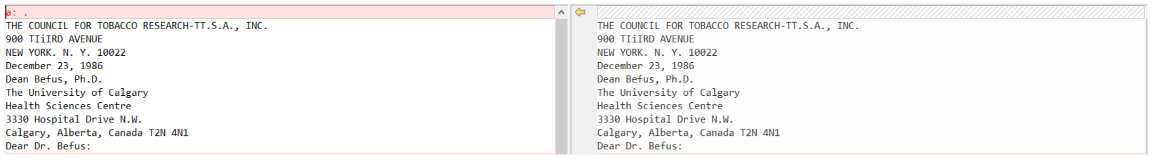
```
     IN THE SUPERIOR COURT OF THE STATE OF CALIFORNIA
          IN AND FOR THE COUNTY OF SAN FRANCISCO
                        ---oOo---
  LESLIE WHITELEY, et al.,
                 Plaintiffs,
       vs.                                 Case No.  303184
  RAYBESTOS-MANHATTAN, INC.,
  et al.,
                Defendants.
                               /
           DEPOSITION OF THOMAS RICHARD ADAMS
                WEDNESDAY, MARCH 15, 2000
  REPORTED BY:
  JO ANN BRUSCELLA, CSR No. 4295
TOOKER & ANTZ
            COURT REPORTING & VIDEO SERVICES
 MISSION STREET, 5TH FLOOR
             SAN FRANCISCO, CALIFORNIA 94103
(415) 392-0650

                         I N D E X
  DEPOSITION OF THOMAS RICHARD ADAMS
  EXAMINATION BY:                                         PAGE
       MR. BROWN                                             4
                         --oOo--
                 INSTRUCTION NOT TO ANSWER
                       PAGE    LINE
                         36    19
                         38    24
                         40    20
                         54    12

                         64    24

       BE IT REMEMBERED that, pursuant to Notice of Taking
  Deposition, and on Wednesday, March 15, 2000, commencing at
  the hour of 9:34 a.m. thereof, at 818 Mission Street,
  5th Floor, San Francisco, California, before me,
  JO ANN BRUSCELLA, duly authorized to administer oaths
  pursuant to Section 2093(b) of the California Code of Civil
  Procedure, appeared telephonically
                    THOMAS RICHARD ADAMS,
  called as a witness on behalf of Plaintiffs, and the said
  witness, being by me first duly sworn, was thereupon examined
```

Figure 5.3: Cleaning page and line numbers from depositions (before vs. after)



```
Encl
HCM/mla
pgNbr=1
```

Figure 5.4: The results of cleaning garbage characters, excess spaces, and service page numbers

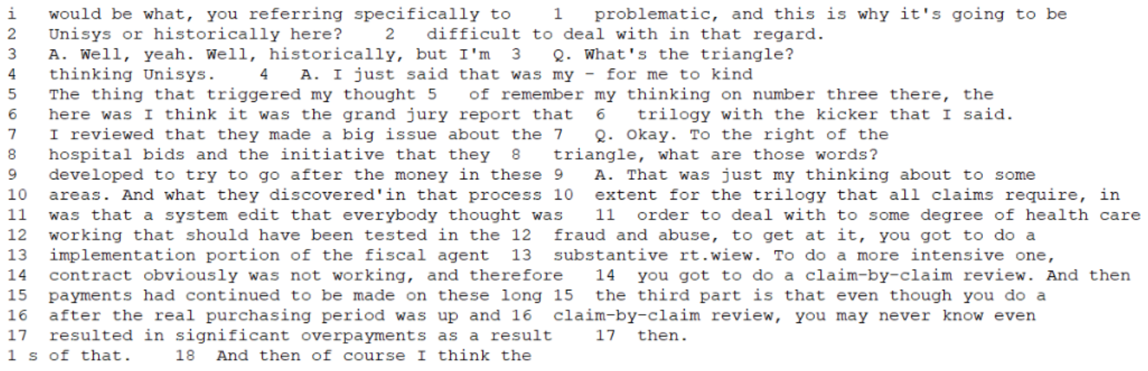


```
a: .
THE COUNCIL FOR TOBACCO RESEARCH-TT.S.A., INC.
900 TIiIRD AVENUE
NEW YORK. N. Y. 10022
December 23, 1986
Dean Befus, Ph.D.
The University of Calgary
Health Sciences Centre
3330 Hospital Drive N.W.
Calgary, Alberta, Canada T2N 4N1
Dear Dr. Befus:
```

```
THE COUNCIL FOR TOBACCO RESEARCH-TT.S.A., INC.
900 TIiIRD AVENUE
NEW YORK. N. Y. 10022
December 23, 1986
Dean Befus, Ph.D.
The University of Calgary
Health Sciences Centre
3330 Hospital Drive N.W.
Calgary, Alberta, Canada T2N 4N1
Dear Dr. Befus:
```

Figure 5.5: The results of cleaning garbage characters, excess spaces

It should be mentioned that some files may contain line numbers not only in the beginning of the line, but within the text as well. An example of such a file is given in Figure 5.6. Also, garbage characters that cannot be identified automatically exist, such as recognized handwritten phrases in certain documents, which might not appear in any other file.

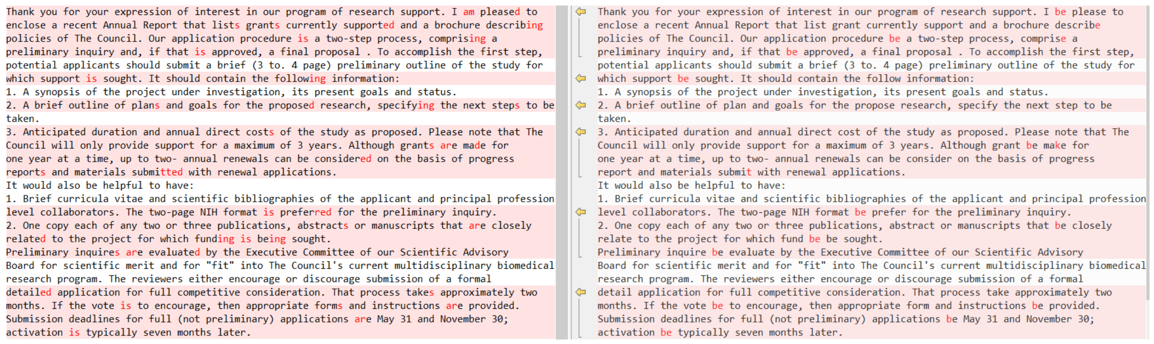


```
i   would be what, you referring specifically to    1   problematic, and this is why it's going to be
2   Unisys or historically here?    2   difficult to deal with in that regard.
3   A. Well, yeah. Well, historically, but I'm  3   Q. What's the triangle?
4   thinking Unisys.     4   A. I just said that was my - for me to kind
5   The thing that triggered my thought 5   of remember my thinking on number three there, the
6   here was I think it was the grand jury report that  6   trilogy with the kicker that I said.
7   I reviewed that they made a big issue about the 7   Q. Okay. To the right of the
8   hospital bids and the initiative that they  8   triangle, what are those words?
9   developed to try to go after the money in these 9   A. That was just my thinking about to some
10  areas. And what they discovered'in that process 10  extent for the trilogy that all claims require, in
11  was that a system edit that everybody thought was   11  order to deal with to some degree of health care
12  working that should have been tested in the 12  fraud and abuse, to get at it, you got to do a
13  implementation portion of the fiscal agent  13  substantive rt.wiew. To do a more intensive one,
14  contract obviously was not working, and therefore   14  you got to do a claim-by-claim review. And then
15  payments had continued to be made on these long 15  the third part is that even though you do a
16  after the real purchasing period was up and 16  claim-by-claim review, you may never know even
17  resulted in significant overpayments as a result    17  then.
1 s of that.    18  And then of course I think the
```

Figure 5.6: Line numbers within the document

2. Tokenization and stemming

   Some natural language processing approaches require the input text to be represented in the form of separate instances, i.e., tokens. For other tasks, it might be necessary to bring the tokens into their initial form. This operation is also known as stemming or lemmatization. This topic is discussed in detail in Section 2.3. Following the received requests and anticipating potential needs, we prepared a Python script that breaks down a single .ocr file into tokens and returns a file with the original lemmas of the tokens. A screenshot with the processing outcome is shown in Figure 5.7.



```
Thank you for your expression of interest in our program of research support. I am pleased to
enclose a recent Annual Report that lists grants currently supported and a brochure describing
policies of The Council. Our application procedure is a two-step process, comprising a
preliminary inquiry and, if that is approved, a final proposal . To accomplish the first step,
potential applicants should submit a brief (3 to. 4 page) preliminary outline of the study for
which support is sought. It should contain the following information:
1. A synopsis of the project under investigation, its present goals and status.
2. A brief outline of plans and goals for the proposed research, specifying the next steps to be
taken.
3. Anticipated duration and annual direct costs of the study as proposed. Please note that The
Council will only provide support for a maximum of 3 years. Although grants are made for
one year at a time, up to two- annual renewals can be considered on the basis of progress
reports and materials submitted with renewal applications.
It would also be helpful to have:
1. Brief curricula vitae and scientific bibliographies of the applicant and principal profession
level collaborators. The two-page NIH format is preferred for the preliminary inquiry.
2. One copy each of any two or three publications, abstracts or manuscripts that are closely
related to the project for which funding is being sought.
Preliminary inquires are evaluated by the Executive Committee of our Scientific Advisory
Board for scientific merit and for "fit" into The Council's current multidisciplinary biomedical
research program. The reviewers either encourage or discourage submission of a formal
detailed application for full competitive consideration. That process takes approximately two
months. If the vote is to encourage, then appropriate forms and instructions are provided.
Submission deadlines for full (not preliminary) applications are May 31 and November 30;
activation is typically seven months later.
```

```
Thank you for your expression of interest in our program of research support. I be please to
enclose a recent Annual Report that list grant currently support and a brochure describe
policies of The Council. Our application procedure be a two-step process, comprise a
preliminary inquiry and, if that be approved, a final proposal . To accomplish the first step,
potential applicants should submit a brief (3 to. 4 page) preliminary outline of the study for
which support be sought. It should contain the follow information:
1. A synopsis of the project under investigation, its present goals and status.
2. A brief outline of plan and goals for the propose research, specify the next step to be
taken.
3. Anticipated duration and annual direct cost of the study as proposed. Please note that The
Council will only provide support for a maximum of 3 years. Although grant be make for
one year at a time, up to two- annual renewals can be consider on the basis of progress
report and materials submit with renewal applications.
It would also be helpful to have:
1. Brief curricula vitae and scientific bibliographies of the applicant and principal profession
level collaborators. The two-page NIH format be prefer for the preliminary inquiry.
2. One copy each of any two or three publications, abstract or manuscripts that be closely
relate to the project for which fund be be sought.
Preliminary inquire be evaluate by the Executive Committee of our Scientific Advisory
Board for scientific merit and for "fit" into The Council's current multidisciplinary biomedical
research program. The reviewers either encourage or discourage submission of a formal
detail application for full competitive consideration. That process take approximately two
months. If the vote be to encourage, then appropriate form and instructions be provided.
Submission deadlines for full (not preliminary) applications be May 31 and November 30;
activation be typically seven months later.
```

Figure 5.7: Results of tokenization and lemmatization (before vs. after)

5.4 Deliverables and Timeline

In order to keep track of our progress, we developed a list of deliverables. The deliverables are summarized in Table 5.1.

Table 5.1: Deliverables

Task	Completion date
Sample of metadata JSON files and document text JSON files to be consumed by the ELS team	09.17.19
Complete structure of the MySQL metadata database	09.24.19
JSON text files for deposition documents (ELS)	10.02.19
JSON metadata files for deposition documents (ELS)	10.03.19
Tokenization and lemmatization done for the TML team	10.08.19
Cleaned content of deposition documents (TML)	10.24.19
Content of article documents extracted for TML team	10.24.19
Different OCR approaches tested on a single PDF	10.28.19
Extracted metadata for all the document types for the ELS team	11.07.19
Cleaned and pre-processed article-style documents (TML)	11.15.19
Generated JSON metadata files for 1M documents	11.18.19
Different OCR approaches tested on a sub-set of PDFs	11.20.19
Generated JSON data files for 1M documents	11.20.19
Recommendations on choosing OCR tools, including comparative table	12.02.19
Script for generating JSON metadata files with the field containing cleaned document content	12.03.19
Script for UNIT testing and validating of JSON files	12.03.19
JSON metadata files for all 14M documents	12.06.19
JSON metadata+text files for 100K documents as requested by the ELS team	12.07.19
All the scripts uploaded to the GitLab repository	12.09.19

Chapter 6

User Manual

6.1 Data Source

UCSF's IDL dataset can be downloaded free of charge. The dataset contains a README file that explains how to set up the database.

Links to official resources:

- Official IDL Website:
  https://www.industrydocuments.ucsf.edu/
- Downloadable IDL dataset:
  https://ucsf.app.box.com/v/IDL-DataSets

6.2 MariaDB

We used a CentOS Linux 7 machine for development. To install MariaDB 5.5 on CentOS 7, follow these steps:

- Install the MariaDB package

  ```
  sudo yum install mariadb-server
  ```

- Start and enable the MariaDB service

  ```
  sudo systemctl start mariadb
  ```

```
sudo systemctl enable mariadb
```

- Verify that the installation was successful. The output should state "`Active: active (running)`".

```
sudo systemctl status mariadb
```

6.3 Python

We used Python 3, which can be installed on CentOS 7 by following these steps:

- Install IUS (Inline with Upstream Stable)

```
sudo yum -y install
 ↪  https://centos7.iuscommunity.org/ius-release.rpm
```

- Install the most recent version of Python

```
sudo yum -y install python36u
```

- Verify that the installation was successful. The output should be "`Python 3.6.8`".

```
python3.6 -V
```

6.4 Useful Linux Commands

We have found the following Linux commands to be useful in our work.

1. `ssh user1@tobacco.cs.vt.edu`

   This command is used to securely log into the remote machine that we used for development.

2. `mysql -u root`

   This command is used to launch the MySQL shell and enter it as the root user.

6.5 Interaction with other teams

During the course of this project, we interacted with multiple teams in order to understand the requirements and fulfill them in a timely manner.

6.5.1 ElasticSearch team

We worked closely with the ElasticSearch team to come up with a suitable contract for the metadata and data fields that could be ingested into ElasticSearch. We retrieved the metadata and data from a MySQL database and converted it into a JSON for ElasticSearch ingestion.

6.5.2 Text Analytics and Machine Learning team

We worked with the Text Analytics and Machine Learning team to understand their requirements and provided them with suitable data. We pre-processed the raw data extracted and handed it over to the TML team for further processing. The TML team used the data provided to apply machine learning algorithms to extract patterns, summaries, and recommendations.

6.5.3 Front-end and Kibana team

We worked closely with our client, Dr. Townsend to understand his research needs in order to guide the Front-end and Kibana (FEK) team in terms of insights about features that can be used for visualizations.

Chapter 7

Developer's Manual

The aim of this section is to frame it in a manner such that it may be used as a manual to redevelop the project. We also provide links to external references and points of contact.

We began by reading a tutorial drawn up by Saurabh Chakrabarty [11], who had undertaken parts of this project earlier. The instructions therein provided us with an initial idea of what could be expected to be achieved over the course of this project.

The UCSF library website hosting 14 million tobacco documents was the data source. The site contained a list with definitions for the various topics and keywords one can use to navigate through these documents. The definitions proved to be very helpful in understanding the data structure.

7.1 Metadata

The IDL dataset contains 76,014 types of documents. Table 7.1 lists the top five document types with the highest number of documents.

Table 7.1: Document types: Frequent

Document Type	Number of Documents
Letter	1954308
Report	934164
Memo	795282
Note	660035
Email	587367

Table 7.2 lists some document types with a moderate number of documents. These are the types of documents that occupy a chunk of the tobacco dataset but are individually lesser in number as compared to the types of documents in Table 7.1.

Table 7.2: Document types: Moderate

Document Type	Number of Documents
proposal	25203
personnel information	12449
collage	7768
patent application	5988

Table 7.3 lists some document types with a smaller number of documents. These are the types of documents that appear infrequently in the tobacco dataset, so the total number of their instances is much less when compared with the types of documents in Table 7.1.

Table 7.3: Document types: Sparse

Document Type	Number of Documents
microfilm	1146
application	774
flow chart	542
survey	310
legislation	207
telex message	17

Table 7.4 lists some document types that are actually combinations of discrete document types. By inspecting some of these documents, it was found that these generally contain text that can be categorized as belonging to two or more document types. For example, a single PDF of 2 pages where the first page contains a letter and the second page contains an article, can be found to be stored under the document type 'letter; article.' Also, these files are mostly longer, when compared to other tobacco documents which are classified as of a certain type. The number of documents belonging to a certain combination of discrete file types can vary between moderate to very low.

Table 7.4: Document types: Combined

Document Type	Number of Documents
bibliography; chart; graph; map; report, scientific	3766
consumer response; letter; letter, consumer	2545
budget; budget review; chart; graph; map; memo; table	365
brand plan; chart; graph; map; memo; promotional material; table	323
handwritten; report; report, market research	80
email; revision; speech	77
advertisement; agenda; brand review; presentation; promotional material; speech	2

Table 7.5 lists examples of a few discrete document types which are extremely rare in the tobacco dataset, with occurrence counts like 1 or 2.

Table 7.5: Document types: Rare

Document Type	Number of Documents
study proposal	1
submission, industry commission inquiry	1
response statement	1
disclosure of invention	1
19641014	1

Table 7.6 lists the metadata fields along with their descriptions.

Table 7.6: Metadata field descriptions

Begin of Table		
ID	Code	Description
0	IGNORE	IGNORE
1	tid	Legacy (LTDL2) Tobacco Id
2	cn	Collection
3	ti	Title
4	dd	Document Date
5	au	Author

Continuation of Table 7.6		
ID	Code	Description
6	recommend	Recommend
7	auo	Organization Author
8	aup	Person Author
10	refdoc	Referenced Document
11	at	Attending
12	ato	Organization Attending
13	atp	Person Attending
14	brd	Brands
15	bnalias	Alternate Bates Range
16	mbn	Bates Mater
17	othernum	Other Bates Range
18	bn	Bates Number
19	cc	Copied
20	cco	Organization Copied
21	ddi	Date Added Industry Site
24	case	Case
25	desc	Description
27	ddu	Date Added UCSF
28	ddprod	Date Produced
29	dt	Document Type
32	availability	availablility
33	availabilitystatus	availablilitystatus
35	fn	File Number
36	grantnum	Grant Number
38	men	Mentioned
39	meno	Organization Mentioned
40	menp	Person Mentioned
45	pgdisp	Page Count Display
46	attach	Attached Artifacts
48	box	Box Number
49	rc	Recipient

Continuation of Table 7.6		
ID	Code	Description
51	rco	Organization Recipient
52	rcp	Person Recipient
53	redact	Redacted
54	rnm	Minnesota Request Number
55	rno	Other Request Number
56	reqno	Primary Request Number
57	area	Area
58	speccoll	Special Collection
59	ddship	Ship Date
60	mm	Multimedia
61	st	Status
63	topic	Topic
64	food	Food
65	pgmap	Page Map
66	ccp	Person Copied
67	kw	Keyword
68	access	Availability
69	rt	Run Time
70	genre	Genre
76	per	comp:persons
78	org	comp:org
83	mbn_begin	Master Bates begin
85	mbn_end	Master Bates End
87	w	Witness Name
89	journal	Journal Citation
90	folder	Folder
91	series	Series
92	rights	Rights
93	alias_begin	Alias Begin
94	alias_end	Alias End
95	bns	Numeric start Bates

Continuation of Table 7.6		
ID	Code	Description
96	bne	Numeric end Bates
97	bnp	Bates Number prefix
100	pg	Page Count
101	privilegecode	Privilege Code
103	ct	Country
104	lg	Language
105	dupes	Duplicates
106	chemical	chemical
107	ddm	Date Modified
108	exw	Express Waiver
109	adr	Adverse Ruling
110	dpl	Privilege Log Date
111	source	Source
112	crt	Court
113	df	Document Format
114	ddmu	Date Modified UCSF
115	dpdt	Deposition Date
116	co	Company
117	dg	Drug
118	en	Exhibit Number
119	id	IDDL ID
121	ot	ocr text
122	md	metadata
End of Table		

7.2 Remote Machine

For development purposes, we have used a Virtual Machine (tobacco.cs.vt.edu) hosted by the Department of Computer Science at Virginia Tech, with storage capacity of 2000 GB.

It came from others, pre-installed with MySQL and Python libraries. To connect to the VM, ssh into tobacco.cs.vt.edu as user1.

Run the following command in a terminal:

ssh -l user1 tobacco.cs.vt.edu

When prompted for a password, type that, and press enter to login. You may contact Saurabh Chakrabarty or one of the authors of this report for the password.

We created a directory 'fall2019' to store this group's files.

7.3 IDL Dataset

UCSF hosts another site which stores all the metadata and OCR-ed versions of the different files related to the tobacco settlement cases. We downloaded the metadata and created the database 'data' in our VM.

Next, we created a directory named 'data'. We downloaded all the above mentioned OCR-ed files under this directory. We created the script 'download.sh' to download the zip files to the VM (./data/rawdata) directly from the UCSF site using the wget command.

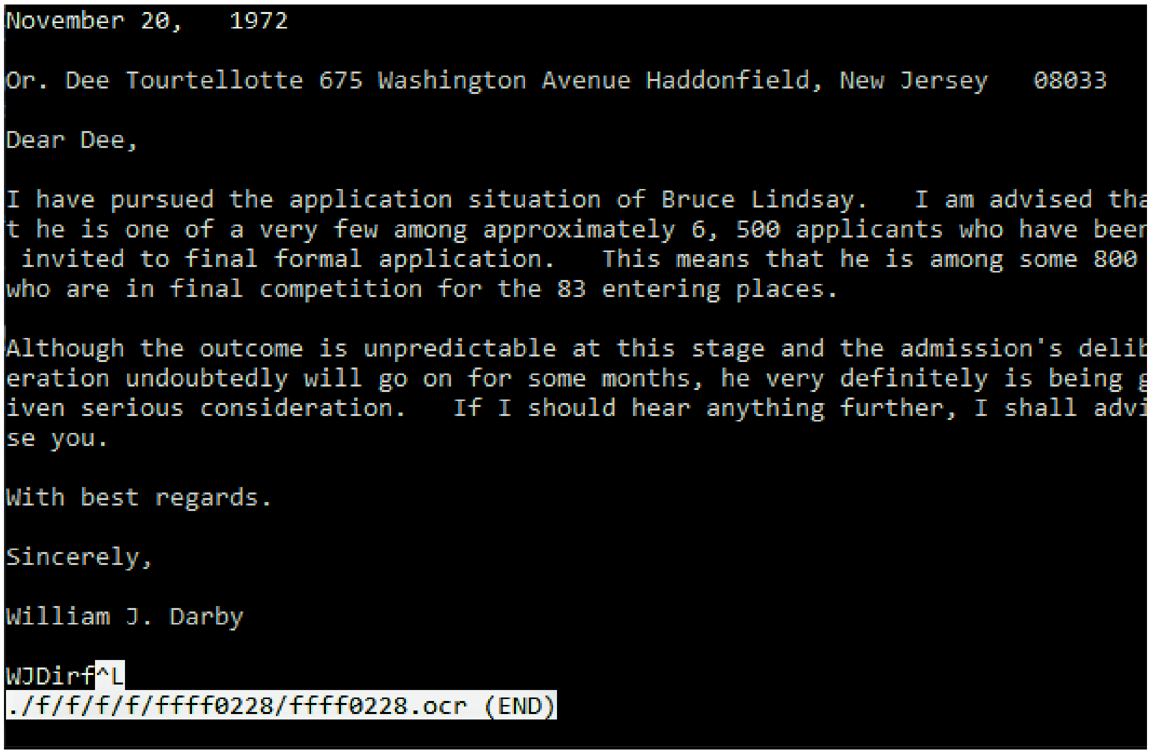
These files were then unzipped at ./fall2019/data using the script saved as 'untar.sh'.

Code used: **tar -xzvf f-j.tar.gz**

These files are stored in a tree like structure with 16 base directories, with 4 levels under each. At each level, there are 16 more directories which are subdivided into 16 further directories. The process is repeated until the last level is reached, where the final folders containing the .ocr files are stored. When we want to access an OCR-ed file with record key ffvv0000, we can find it listed as ffvv0000.ocr under the file path /fall2019/data/f/f/v/v/ffvv0000/

As an example, to view ffff0228.ocr, run the following command in terminal:
vim ./fall2019/data/f/f/f/f/ffff0228/ffff0228.ocr

Figure 7.1 shows the output file.



```
November 20,   1972

Or. Dee Tourtellotte 675 Washington Avenue Haddonfield, New Jersey   08033

Dear Dee,

I have pursued the application situation of Bruce Lindsay.   I am advised tha
t he is one of a very few among approximately 6, 500 applicants who have been
 invited to final formal application.   This means that he is among some 800
who are in final competition for the 83 entering places.

Although the outcome is unpredictable at this stage and the admission's delib
eration undoubtedly will go on for some months, he very definitely is being g
iven serious consideration.   If I should hear anything further, I shall advi
se you.

With best regards.

Sincerely,

William J. Darby

WJDirf^L
./f/f/f/f/ffff0228/ffff0228.ocr (END)
```

Figure 7.1: Depicts the text editable version of a tobacco file saved as ffff0228.ocr

7.4 MySQL Database

The UCSF website also contains an instruction manual and README.txt, which helped us recreate the database on our Virtual Machine. We used the database server MariaDB, which is an open source fork of MySQL, for data manipulation.

First of all, a new database was created under the user ‘root’ and name ‘data’.

Login as root user and no password by running the following command:

mysql -u root -p

Create the database ‘data’:

[mysql] CREATE DATABASE data CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;

Also, notice that we have set the character set to utf8mb4 while creating the database. This is because the default utf8 encoding allows a maximum of 3 bytes per characters which may not support some special characters or alphabets which would require 4 bytes.

Using utf8mb4, which allows up to 4 bytes per character, would work in case there were any special characters in the metadata. Again, collation is a set of rules for comparing and sorting data in a character set. To maintain uniformity, we set the collation to utf8mb4_unicode_ci ('ci' designates case insensitive comparisons).

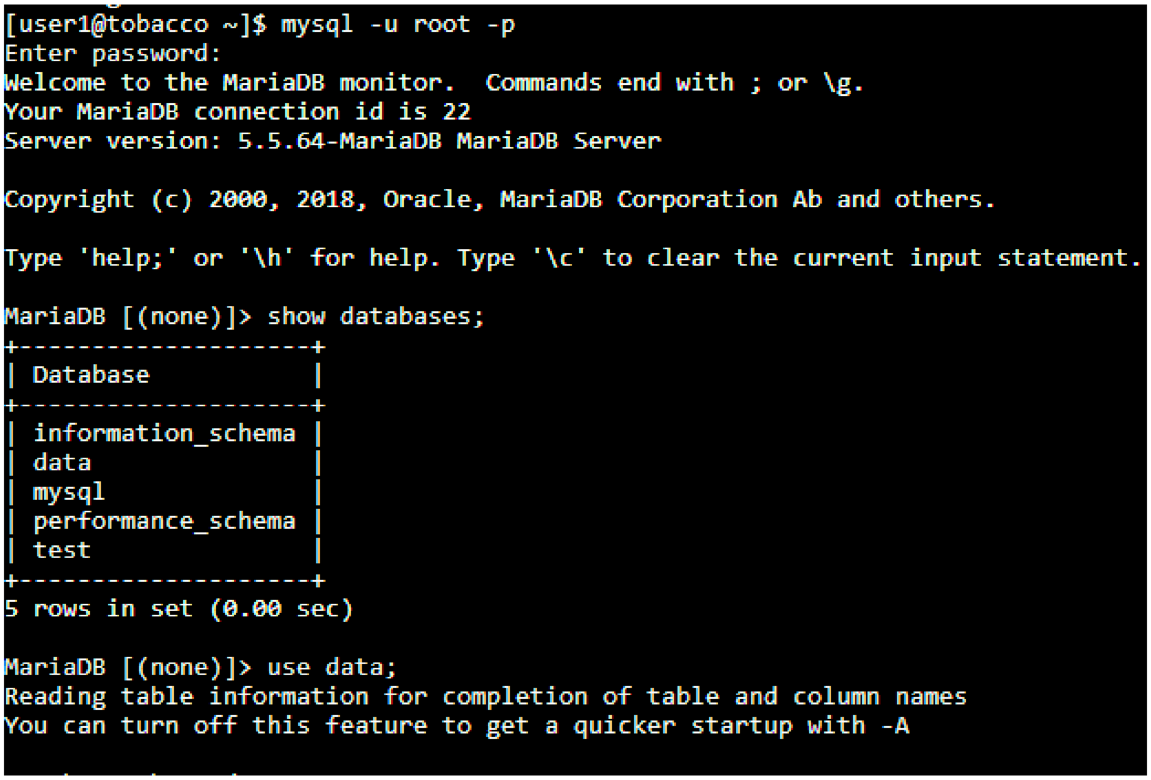
Next, we populated new_data with the metadata from the database dump file. The command used to extract the database dump file is:

tar xzf idl-database-dump.tar.gz

The command used to populate the metadata tables is:

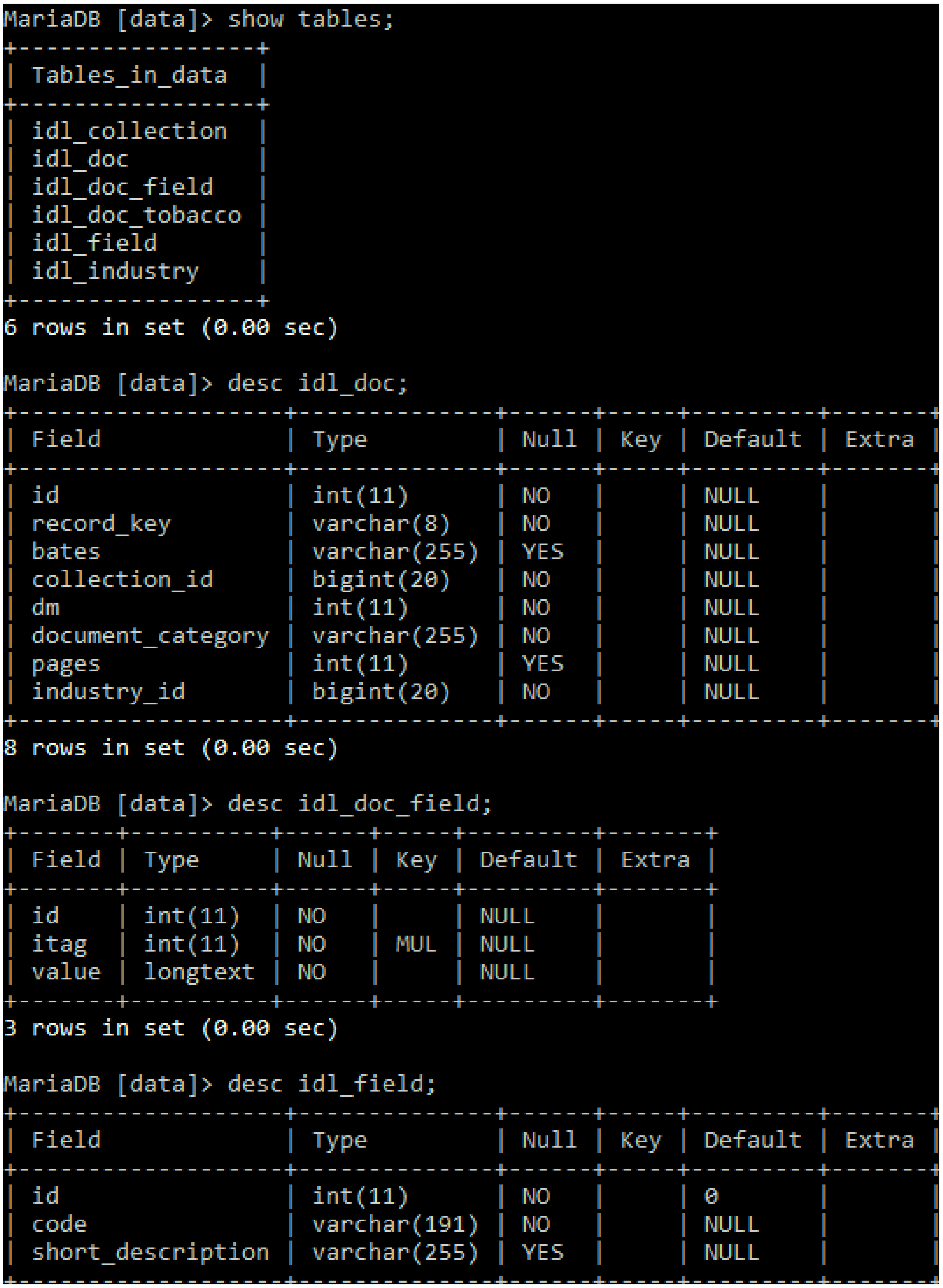
mysql -u root -p data < idl-database-dump.sql

Figure 7.2 shows how one can log into MariaDB as user 'root' and view the databases under it.



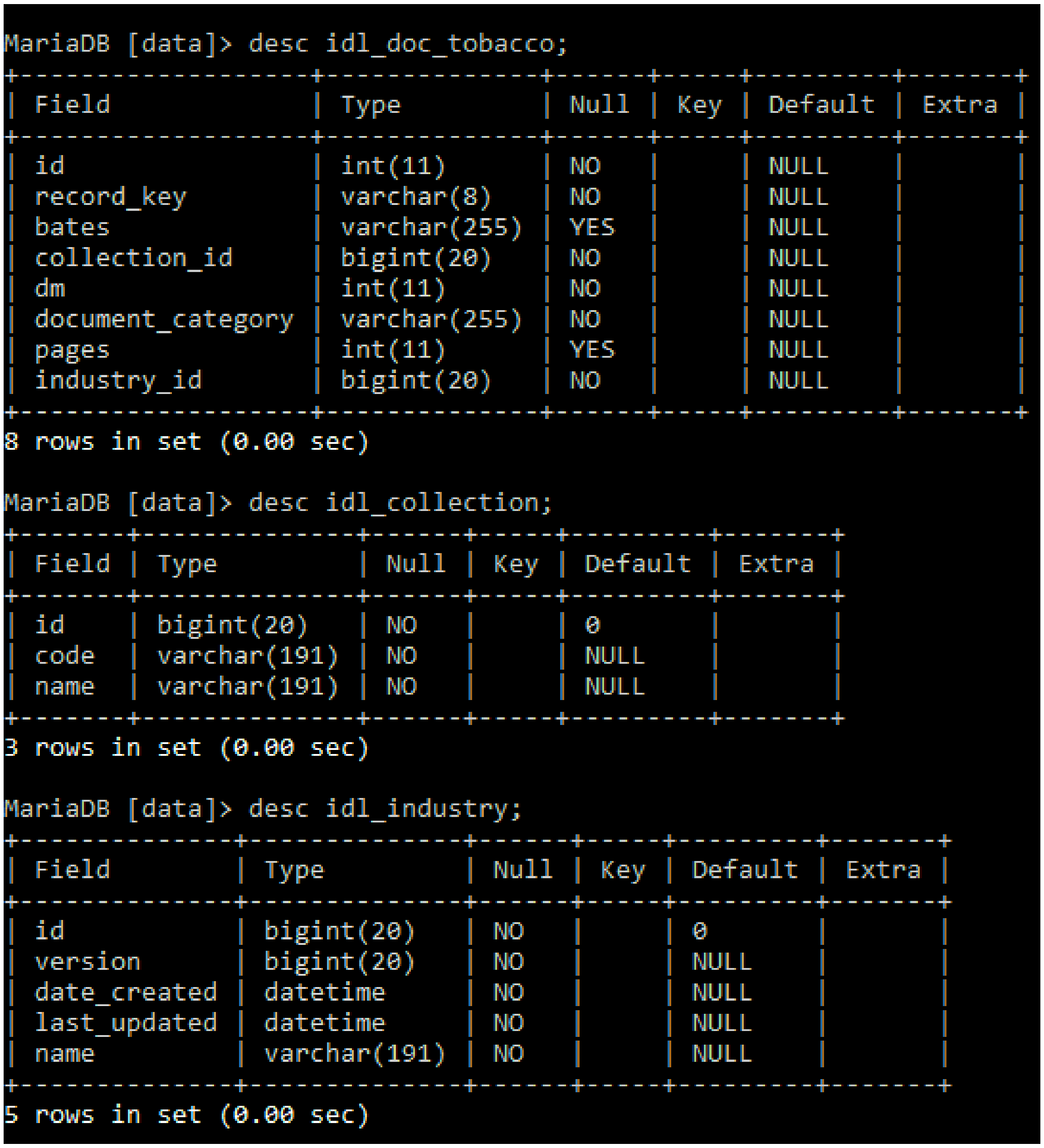
```
[user1@tobacco ~]$ mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 22
Server version: 5.5.64-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| data               |
| mysql              |
| performance_schema |
| test               |
+--------------------+
5 rows in set (0.00 sec)

MariaDB [(none)]> use data;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

Figure 7.2: Logging into MariaDB and viewing data and other databases under it.



```
MariaDB [data]> show tables;
+------------------+
| Tables_in_data   |
+------------------+
| idl_collection   |
| idl_doc          |
| idl_doc_field    |
| idl_doc_tobacco  |
| idl_field        |
| idl_industry     |
+------------------+
6 rows in set (0.00 sec)

MariaDB [data]> desc idl_doc;
+-------------------+--------------+------+-----+---------+-------+
| Field             | Type         | Null | Key | Default | Extra |
+-------------------+--------------+------+-----+---------+-------+
| id                | int(11)      | NO   |     | NULL    |       |
| record_key        | varchar(8)   | NO   |     | NULL    |       |
| bates             | varchar(255) | YES  |     | NULL    |       |
| collection_id     | bigint(20)   | NO   |     | NULL    |       |
| dm                | int(11)      | NO   |     | NULL    |       |
| document_category | varchar(255) | NO   |     | NULL    |       |
| pages             | int(11)      | YES  |     | NULL    |       |
| industry_id       | bigint(20)   | NO   |     | NULL    |       |
+-------------------+--------------+------+-----+---------+-------+
8 rows in set (0.00 sec)

MariaDB [data]> desc idl_doc_field;
+-------+----------+------+-----+---------+-------+
| Field | Type     | Null | Key | Default | Extra |
+-------+----------+------+-----+---------+-------+
| id    | int(11)  | NO   |     | NULL    |       |
| itag  | int(11)  | NO   | MUL | NULL    |       |
| value | longtext | NO   |     | NULL    |       |
+-------+----------+------+-----+---------+-------+
3 rows in set (0.00 sec)

MariaDB [data]> desc idl_field;
+-------------------+--------------+------+-----+---------+-------+
| Field             | Type         | Null | Key | Default | Extra |
+-------------------+--------------+------+-----+---------+-------+
| id                | int(11)      | NO   |     | 0       |       |
| code              | varchar(191) | NO   |     | NULL    |       |
| short_description | varchar(255) | YES  |     | NULL    |       |
+-------------------+--------------+------+-----+---------+-------+
```

Figure 7.3: Viewing the tables idl_doc, idl_doc_field, and idl_field present in the database ‘data’ and the corresponding table specifications.



```
MariaDB [data]> desc idl_doc_tobacco;
+-------------------+--------------+------+-----+---------+-------+
| Field             | Type         | Null | Key | Default | Extra |
+-------------------+--------------+------+-----+---------+-------+
| id                | int(11)      | NO   |     | NULL    |       |
| record_key        | varchar(8)   | NO   |     | NULL    |       |
| bates             | varchar(255) | YES  |     | NULL    |       |
| collection_id     | bigint(20)   | NO   |     | NULL    |       |
| dm                | int(11)      | NO   |     | NULL    |       |
| document_category | varchar(255) | NO   |     | NULL    |       |
| pages             | int(11)      | YES  |     | NULL    |       |
| industry_id       | bigint(20)   | NO   |     | NULL    |       |
+-------------------+--------------+------+-----+---------+-------+
8 rows in set (0.00 sec)

MariaDB [data]> desc idl_collection;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| id    | bigint(20)   | NO   |     | 0       |       |
| code  | varchar(191) | NO   |     | NULL    |       |
| name  | varchar(191) | NO   |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

MariaDB [data]> desc idl_industry;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| id           | bigint(20)   | NO   |     | 0       |       |
| version      | bigint(20)   | NO   |     | NULL    |       |
| date_created | datetime     | NO   |     | NULL    |       |
| last_updated | datetime     | NO   |     | NULL    |       |
| name         | varchar(191) | NO   |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Figure 7.4: Viewing the tables idl_data_tobacco, idl_collection, and idl_industry with their descriptions under data

Figures 7.3 and 7.4 show how one can view the various tables in the database ‘data’ and view the corresponding table descriptions.

To join idl_doc and idl_doc_field, use:

SELECT * FROM idl_doc id, idl_doc_field idf WHERE id.id=idf.id;

To retrieve record keys, use:

SELECT record_key FROM idl_doc WHERE id=‘id_value’;

To view descriptions of different numerical codes under the column idl_doc_field.itag, use:

Select * from idl_field; The database has metadata for documents pertaining to four industry types referenced by distinct ID, as shown in Table 7.7. To view the different types of industries whose files are in the database data, use:

Select * from idl_industry;

Refer to Table 7.7 to view the output of the above query.

ID	Name
1	Drug
2	Tobacco
3	Food
4	Chemical

Table 7.7: Values and meanings of the industry IDs (idl_doc.industry_id)

To view records of files belonging to the tobacco industry only, use:

Select * from idl_doc where industry_id=2;

OR,

Select * from idl_doc_tobacco;

Table idl_doc_tobacco has been created by extracting data from idl_doc where the value of industry_id is ‘2’, i.e., fetching rows from the table pertaining to the tobacco industry. This was done to facilitate faster traversal of the rows related to tobacco case files while executing SQL queries.

To view record_keys of document type deposition, use:

SELECT distinct record_key FROM idl_doc_tobacco id, idl_doc_field idf WHERE id.id=idf.id AND idf.itag=‘29’ AND idf.value=‘deposition’;

7.4.1 Use of the Tables idl_industry and idl_field

In case we want to know which kind of industry a document with idl_doc.record_key='ffvv0000' and idl_doc.industry_id='2' belongs to, we would be able to query the table idl_industry with the industry_id which would return us a row with the meaning of '2.' See Table 7.7 for an example.

The table idl_field contains the significance of all the 'itag' attributes attached to any document we encounter in the table idl_doc_field. For example: A document has itag attribute = '29'. We may query the table idl_field with idl_field.id=idl_doc_field.itag to view its meaning. Refer to Table 7.8 to view the output of the above query.

ID	Code	Short_Description
29	dt	Document Type

Table 7.8: Description of the itag value 29 fetched by querying the table idl_field

Here 'ID'= value of itag and 'Short_Description' is the meaning of the itag.

7.5 Scripts

7.5.1 Test Runs

Having completed the setup, our next job was to start identifying the scope of our work and create short term goals. We decided upon creating a smaller prototype with only the metadata of documents of the type 'deposition' (**select distinct record_key from idl_doc_tobacco id, idl_doc_field idf where id.id=idf.id and idf.itag=29 and idf.value='deposition';**). Typically, we would test out our code on a couple of instances before running it on the full set of deposition related data.

We ended up creating Python scripts with functionalities as follows:

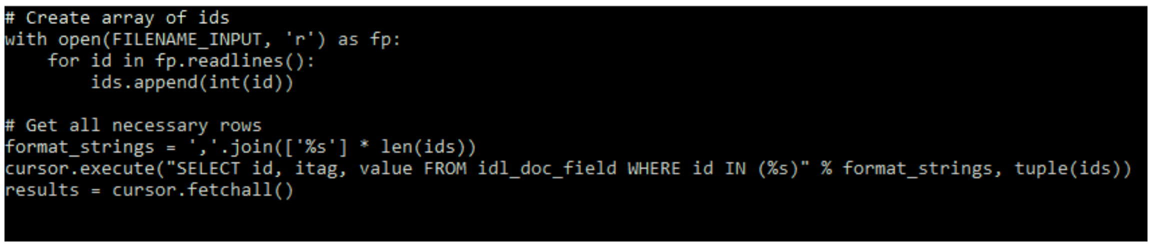
1. **utils.py**: This script stores values of several constants with multiple instances of use throughout our scripts. It stores the constants username, password, hostname, and database name required to connect to the database 'new_data'. Additionally, the function connect_db() defined here can be called from other programs to connect to 'new_data' readily without going through the trouble of writing the code and setting up a connection every time.

2. **determine_depositions.py**: This script was initially used to fetch the record ID (idl_doc_tobacco.id) of all the tobacco files which are of the type 'deposition'. The output was stored in the text file deposition_ids_full.txt. However, in the next iteration, we upgraded the code to create script (3).

3. **determine_ids_by_type.py**: This script is an upgraded version of script (2) and can be used to fetch the record ID (idl_doc_tobacco.id) of all the tobacco files by the document type. Currently, we ran this to fetch the IDs of the document types 'deposition' and 'article' and their corresponding outputs are stored in the text files deposition_ids_full.txt and article_ids_full, respectively. A subset of the deposition IDs are stored in another text file, deposition_ids_100.txt, which was initially used as an input in script (4). After successful execution with the small subset, we moved on to process all the depositions by using deposition_ids_full.txt as input in script (4). The same process was repeated for the articles.

4. **metadata_to_json.py**: The next part was to convert the metadata of the files that we will be using into JSON format. Since we were just dealing with depositions initially, we converted the metadata for only that document type. In the next iteration we also included metadata for document type 'article'. This script helped in achieving that, by using ID (idl_doc_tobacco.ID) of deposition documents and articles as input. We had used the static file deposition_ids_full.txt as the input for depositions, and article_ids_full.txt for articles. One should note that, based on the requirements from the ELS Team, we have included the URL of the source of each file as part of the JSON file containing the metadata. Also, we have altered the date format to match mm/dd/yyyy based on their requirements, in the current code.

   Figure 7.5 shows a code snippet used to read the ID from `deposition_ids_100.txt` and fetch `idl_doc_tobacco.id` from the database.



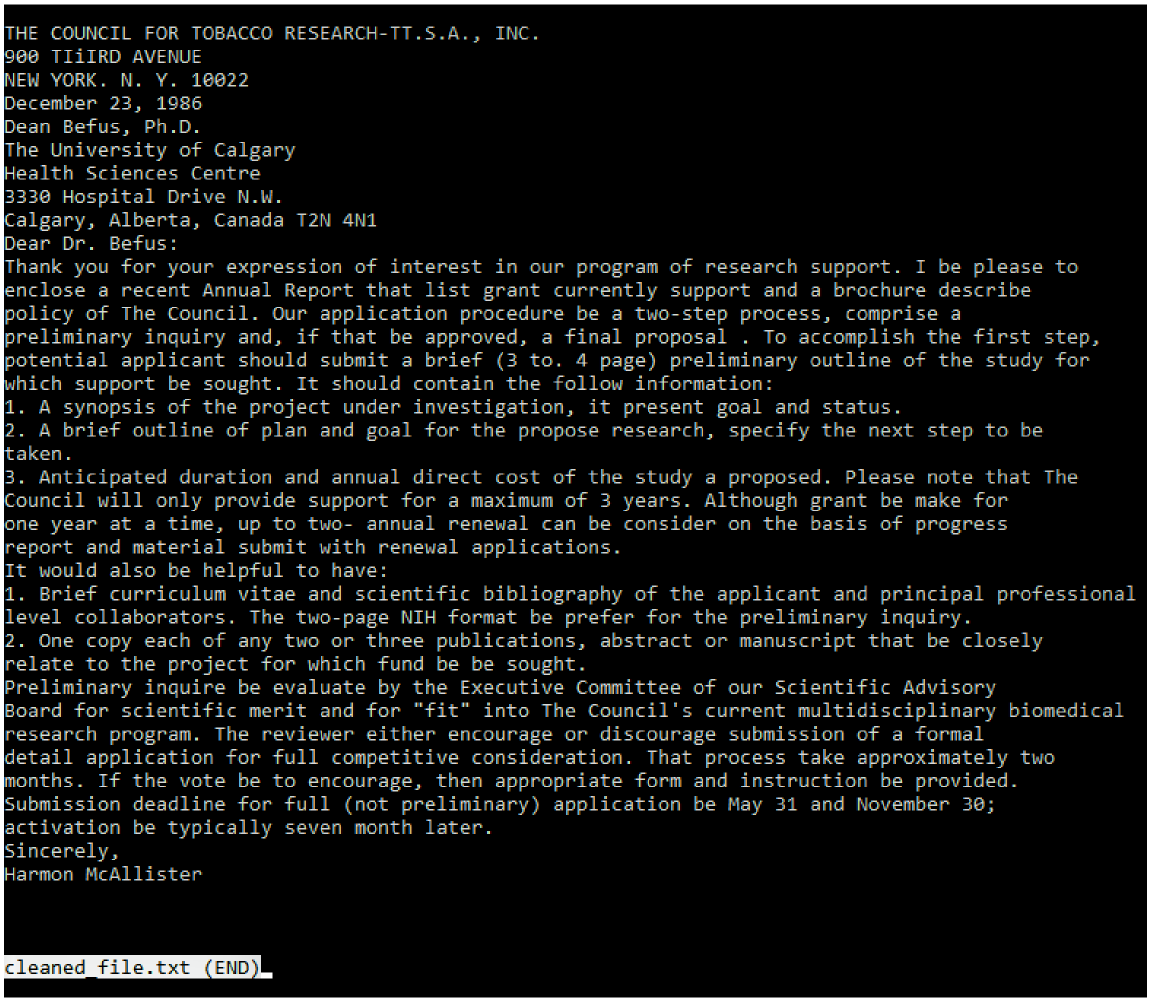
```
# Create array of ids
with open(FILENAME_INPUT, 'r') as fp:
    for id in fp.readlines():
        ids.append(int(id))

# Get all necessary rows
format_strings = ','.join(['%s'] * len(ids))
cursor.execute("SELECT id, itag, value FROM idl_doc_field WHERE id IN (%s)" % format_strings, tuple(ids))
results = cursor.fetchall()
```

Figure 7.5: Section of code from metadata_to_json.py

The output of this script is the generation of the file deposition_metadata.json and article_metadata.json which are stored in Ceph with file path: /mnt/ceph/shared/tobacco/metadata

5. **file2json.py**: This script was initially generated to convert the text content of the OCR-ed tobacco documents into JSON. We were able to find 8136 deposition documents, out of which four thousand were successfully parsed initially. The OCR-ed documents which were generated based on scanned documents contained many anomalous characters. These were due to the presence of handwritten texts, company logos, signatures and other less machine recognizable fonts in these scanned documents. The OCR techniques would fail to recognize these as valid English characters and pass garbage characters like '* '. We found that the presence of these anomalous characters in the OCR-ed documents were causing the misses. We were able to work around the problem by encoding each word in the text and leaving out the ones that could not be encoded from the file to overcome this impediment. Later we were able to parse all of those 8136 'deposition' documents. These JSON files can be found under the folder /fall2019/deposition_json. The record_key of the files serve as their name under this directory.

6. **Preprocess_sample.py**: This script performs the following text processing tasks on the tobacco file texts: tokenization, lemmatization, and removal of unnecessary white spaces, numbers, and garbage characters. To view the processed version of the file fghb0000.ocr, run the following command: **less cleaned_file.txt**. Figure 7.6 shows an example of a processed file. However, in the final run, we ended up not requiring to implement lemmatization and tokenization on our documents; these tasks were designated to be taken care of by the TML team.



```
THE COUNCIL FOR TOBACCO RESEARCH-TT.S.A., INC.
900 TIiIRD AVENUE
NEW YORK. N. Y. 10022
December 23, 1986
Dean Befus, Ph.D.
The University of Calgary
Health Sciences Centre
3330 Hospital Drive N.W.
Calgary, Alberta, Canada T2N 4N1
Dear Dr. Befus:
Thank you for your expression of interest in our program of research support. I be please to
enclose a recent Annual Report that list grant currently support and a brochure describe
policy of The Council. Our application procedure be a two-step process, comprise a
preliminary inquiry and, if that be approved, a final proposal . To accomplish the first step,
potential applicant should submit a brief (3 to. 4 page) preliminary outline of the study for
which support be sought. It should contain the follow information:
1. A synopsis of the project under investigation, it present goal and status.
2. A brief outline of plan and goal for the propose research, specify the next step to be
taken.
3. Anticipated duration and annual direct cost of the study a proposed. Please note that The
Council will only provide support for a maximum of 3 years. Although grant be make for
one year at a time, up to two- annual renewal can be consider on the basis of progress
report and material submit with renewal applications.
It would also be helpful to have:
1. Brief curriculum vitae and scientific bibliography of the applicant and principal professional
level collaborators. The two-page NIH format be prefer for the preliminary inquiry.
2. One copy each of any two or three publications, abstract or manuscript that be closely
relate to the project for which fund be be sought.
Preliminary inquire be evaluate by the Executive Committee of our Scientific Advisory
Board for scientific merit and for "fit" into The Council's current multidisciplinary biomedical
research program. The reviewer either encourage or discourage submission of a formal
detail application for full competitive consideration. That process take approximately two
months. If the vote be to encourage, then appropriate form and instruction be provided.
Submission deadline for full (not preliminary) application be May 31 and November 30;
activation be typically seven month later.
Sincerely,
Harmon McAllister



cleaned_file.txt (END)
```

Figure 7.6: Cleaned file after application of text processing methods like lemmatization and tokenization.

7.5.2 Final Run

1. **metadata_to_json_fast.py:** This script did two tasks for us in the final run of the project:

   (a) Generate the metadata only: The metadata of all the tobacco documents were generated ordered by the range of 'ID' values fed as user input. Typically, we generated the metadata for 1 million documents at a time. Code used to generate metadata for the first million of these documents is: **python metadata_to_json_fast.py 0 1000000**

   (b) In the next step, we modified this script to generate the metadata along with an additional field 'text' that contained text or data within the tobacco documents. The same method as above could be followed to run this script. Also, based upon the ELS team's requirements that were due to constraints with regards to memory allocation, we generated metadata with data for 100k documents and stored them at Ceph. The location in the ceph directory is: **cd /mnt/ceph/shared/tobacco/**

2. **file2json.py:** Unlike in the test runs, we were no longer required to send the JSON files containing the data within the tobacco documents separately to the ELS team. Instead the file generated using the second version of the script metadata_to_json_fast.py would suffice. However, the script file2json.py is imported by metadata_to_json_fast.py, so that it can use the functions to access the OCR-ed files and extract their cleaned versions as value of the metadata key field 'text'.

3. **file2json2.py:** This script was written to access the OCR-ed files and clean their contents. Then, it would generate files containing cleaned versions of data within those documents and save them by their record key.

4. **utils.py:** This remains unchanged and has the same usage as in Test Run defined in Section 7.5.1.

7.6 Alternative OCR Techniques

The presence of garbage characters and the poor quality of some OCR-ed files have lead us to look for alternative approaches to implement OCR on the tobacco documents. This was mainly to improve the quality of text files which were extracted from scanned images of newspaper articles, blurry texts, and handwritten texts. The methods implemented gave us results with a varying degree of success except for recognizing handwritten texts.

The Python scripts and output text files are saved in the folder ocr at the filepath: /fall2019/ocr/

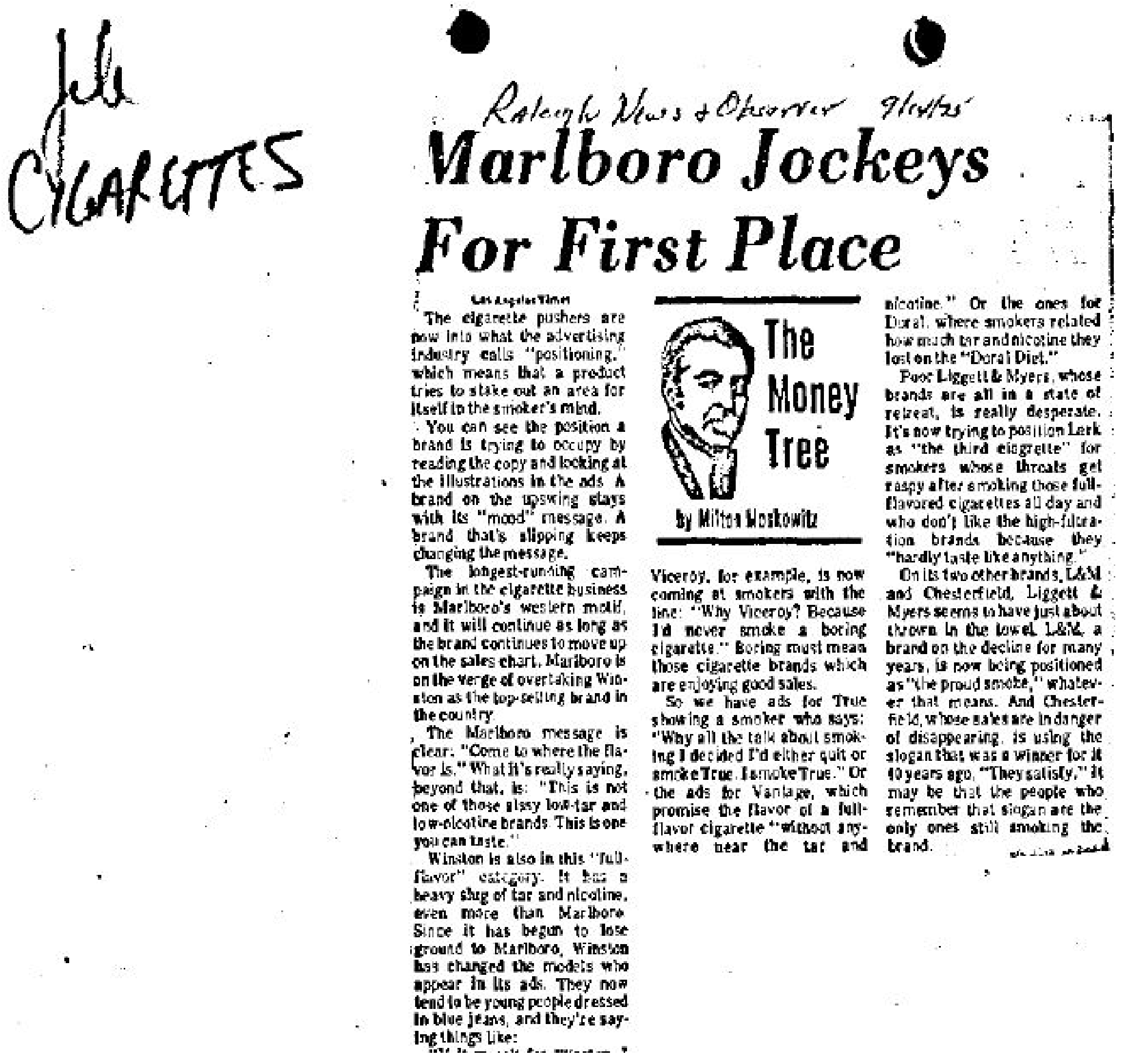
We can do a comparative study based on the file jydp0228. The URL of the original PDF is https://www.industrydocuments.ucsf.edu/tobacco/docs/#id=jydp0228.

1. **PyPDF2**: This approach failed to identify paragraphs and columns within the text. The output had a single paragraph with all of the columns merged together. Figure 2.5 shows the screenshot of the text file extracted by this process. The Python script used can be found with the name test2.py.

2. **PDFMiner**: This package gave us much better results with respect to approach (1). It could identify paragraphs and columns. This would be a fairly efficient approach when we are dealing with PDF files with column texts as it can append the text from the second column on a page below the text from the first column on that page. However, it failed to identify the tabular structure of data in our test file. The columns were simply appended one after the other which resulted in the loss of information to be retrieved from the rows of the table. Figure 2.6 shows a screenshot of the text file retrieved by this process. The Python script used can be found with the name test2.py.

3. **Abbyy Cloud OCR**: This tool was used on the UiPath platform to test the file. We used a trial version which developers can sign up for from Abbyy's official website. This method could extract text with relatively good accuracy. In addition to identifying the columns, it could also recognize the tabular structure of the text. Figure 2.7 shows a screenshot of the text file extracted by this process. We will implement this method by using a Python script next.

We also tested these methods on the scanned version of an old newspaper article belonging to the tobacco dataset, whose OCR-ed file is saved as ffbb0020.ocr in the database. The PDF of the scanned copy can be found at the UCSF website. Figure 7.7 shows a snapshot of the raw document from September 14, 1975.



CIGARETTES

Raleigh News & Observer [illegible]

Marlboro Jockeys For First Place

Los Angeles Times

The cigarette pushers are now into what the advertising industry calls "positioning," which means that a product tries to stake out an area for itself in the smoker's mind.

You can see the position a brand is trying to occupy by reading the copy and looking at the illustrations in the ads. A brand on the upswing stays with its "mood" message. A brand that's slipping keeps changing the message.

The longest-running campaign in the cigarette business is Marlboro's western motif, and it will continue as long as the brand continues to move up on the sales chart. Marlboro is on the verge of overtaking Winston as the top-selling brand in the country.

The Marlboro message is clear: "Come to where the flavor is." What it's really saying, beyond that, is: "This is not one of those sissy low-tar and low-nicotine brands. This is one you can taste."

Winston is also in this "full-flavor" category. It has a heavy slug of tar and nicotine, even more than Marlboro. Since it has begun to lose ground to Marlboro, Winston has changed the models who appear in its ads. They now tend to be young people dressed in blue jeans, and they're saying things like:

The Money Tree

by Milton Moskowitz

Viceroy, for example, is now coming at smokers with the line: "Why Viceroy? Because I'd never smoke a boring cigarette." Boring must mean those cigarette brands which are enjoying good sales.

So we have ads for True showing a smoker who says: "Why all the talk about smoking I decided I'd either quit or smoke True. I smoke True." Or the ads for Vantage, which promise the flavor of a full-flavor cigarette "without anywhere near the tar and nicotine." Or the ones for Doral, where smokers related how much tar and nicotine they lost on the "Doral Diet."

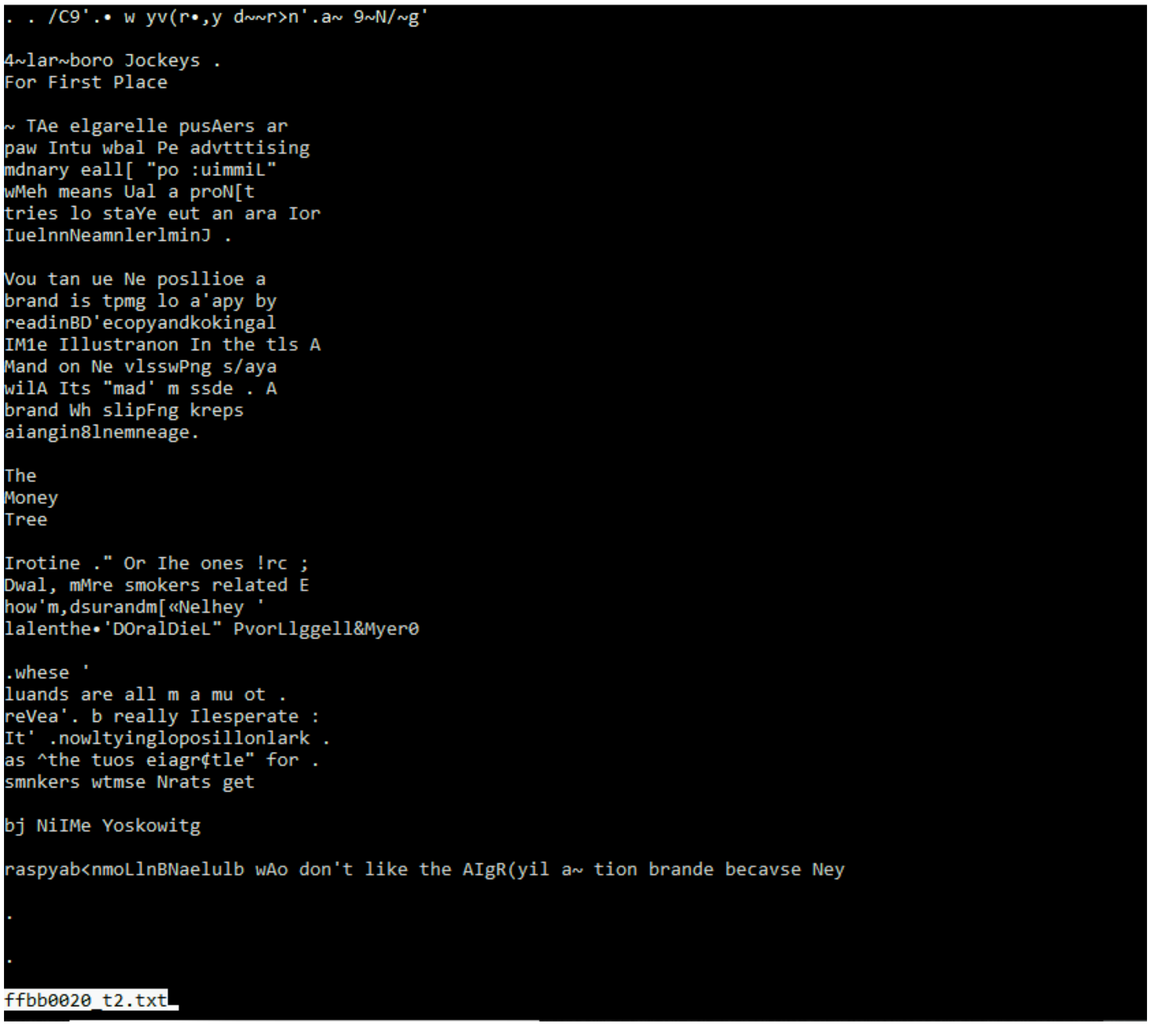
Poor Liggett & Myers, whose brands are all in a state of retreat, is really desperate. It's now trying to position Lark as "the third cigarette" for smokers whose throats get raspy after smoking those full-flavored cigarettes all day and who don't like the high-filtration brands because they "hardly taste like anything."

On its two other brands, L&M and Chesterfield, Liggett & Myers seems to have just about thrown in the towel. L&M, a brand on the decline for many years, is now being positioned as "the proud smoke," whatever that means. And Chesterfield, whose sales are in danger of disappearing, is using the slogan that was a winner for it 40 years ago, "They satisfy." It may be that the people who remember that slogan are the only ones still smoking the brand.

Figure 7.7: Original PDF file saved in UCSF website

It can be seen that the quality of scanned text cannot be regarded too highly in terms of readability; it is unlikely that the font would be easily recognized by machines. The quality of the print in the document is not consistent either. Some of the alphabetics are darker than the others throughout the article, which is probably due to the kind of ink used for printing in those days. In the current OCR-ed version, the body of the document is composed of words which are not necessarily part of the English dictionary and are generated based on the wrong interpretations of words present on the scanned document.

Applying method (1) with PyPDF2 yielded no result as the code failed to parse the text. Method (2) using PDFMiner successfully extracted a text editable file, but the content does not make much sense; it is similar to what is the case with the current version. Figure 7.8 shows the text file obtained.



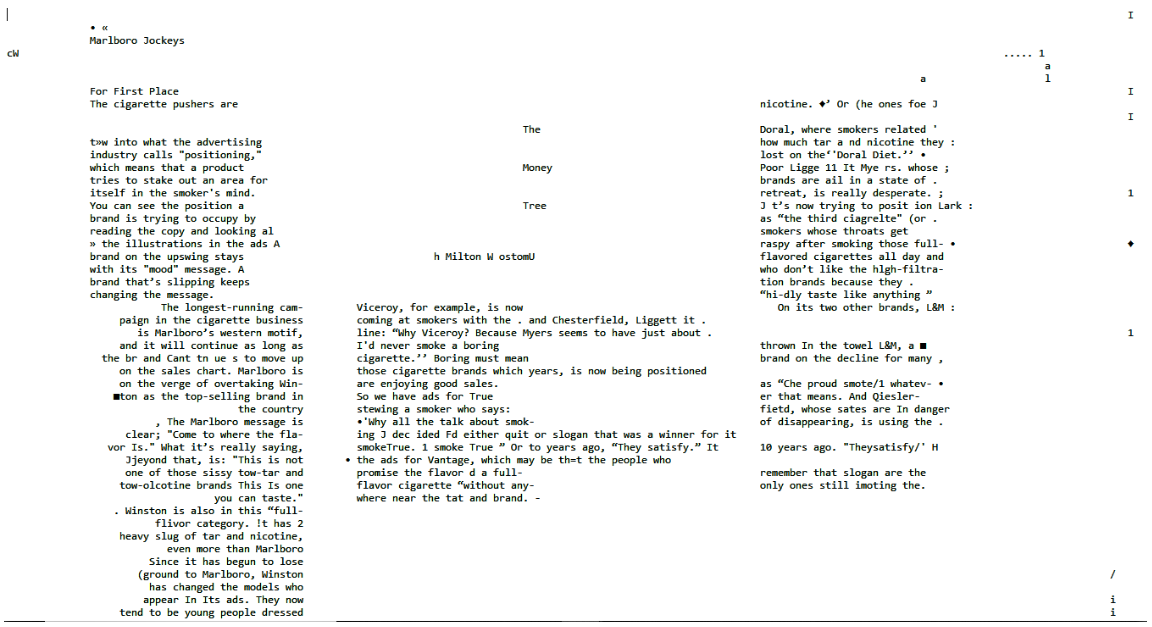
```
. . /C9'.• w yv(r•,y d~~r>n'.a~ 9~N/~g'

4~lar~boro Jockeys .
For First Place

~ TAe elgarelle pusAers ar
paw Intu wbal Pe advtttising
mdnary eall[ "po :uimmiL"
wMeh means Ual a proN[t
tries lo staYe eut an ara Ior
IuelnnNeamnlerlminJ .

Vou tan ue Ne posllioe a
brand is tpmg lo a'apy by
readinBD'ecopyandkokingal
IM1e Illustranon In the tls A
Mand on Ne vlsswPng s/aya
wilA Its "mad' m ssde . A
brand Wh slipFng kreps
aiangin8lnemneage.

The
Money
Tree

Irotine ." Or Ihe ones !rc ;
Dwal, mMre smokers related E
how'm,dsurandm[«Nelhey '
lalenthe•'DOralDieL" PvorLlggell&Myer0

.whese '
luands are all m a mu ot .
reVea'. b really Ilesperate :
It' .nowltyingloposillonlark .
as ^the tuos eiagr¢tle" for .
smnkers wtmse Nrats get

bj NiIMe Yoskowitg

raspyab<nmoLlnBNaelulb wAo don't like the AIgR(yil a~ tion brande becavse Ney

.


.

ffbb0020_t2.txt
```

Figure 7.8: Text file extracted using method (2)

The process failed to recognize the English words and distinguish between sentences in a paragraph. The accuracy of identifying meaningful text was below average. For example, the first sentence in the article which says **"The cigarette pushers are now**

into what the advertising industry calls "positioning". which means that a product tries to stake out an area for itself tn the smoker's mind." has been interpreted as **"TAe elgarelle pusAers ar paw Intu wbal Pe advtttising mdnary eall["po :uimmiL" wMeh means Ual a proN[t tries lo staYe eut an ara Ior IuelnnNeamnlerlminJ ."** As is evident, the resulting text does not pass for a meaningful sentence.

Finally, the method (3) using Abbyy Cloud OCR did manage to extract the text with much better accuracy in this test case. Even though we cannot vouch for the degree of success this method achieves with every other newspaper article, it is not a bleak situation that we look forward to. The English words were mostly well recognized as were the columns in the report. Having said that, due to limited spacing between the second and third column in the scanned document, the spacing between the last two columns is not very wide in the generated text document either. If one refers to Figure 7.9, one may notice that the two columns are pretty close to each other.

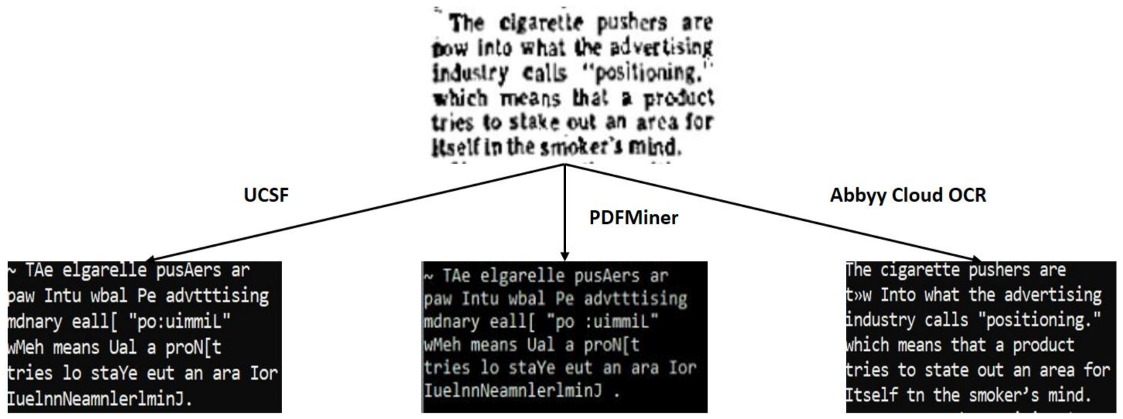


```
|                                                                                                                                                             I
              • «
              Marlboro Jockeys
cW                                                                                                                                                  ..... 1
                                                                                                                                                          a
                                                                                                                                         a                1
              For First Place                                                                                                                                 I
              The cigarette pushers are                                                                                   nicotine. ♦' Or (he ones foe J
                                                                                                                                                              I
                                                                          The                                             Doral, where smokers related '
              t»w into what the advertising                                                                               how much tar a nd nicotine they :
              industry calls "positioning,"                                                                               lost on the''Doral Diet.'' •
              which means that a product                                  Money                                           Poor Ligge 11 It Mye rs. whose ;
              tries to stake out an area for                                                                              brands are ail in a state of .
              itself in the smoker's mind.                                                                                retreat, is really desperate. ;                     1
              You can see the position a                                  Tree                                            J t's now trying to posit ion Lark :
              brand is trying to occupy by                                                                                as “the third ciagrelte" (or .
              reading the copy and looking al                                                                             smokers whose throats get
              » the illustrations in the ads A                                                                            raspy after smoking those full- •                   ♦
              brand on the upswing stays                           h Milton W ostomU                                      flavored cigarettes all day and
              with its "mood" message. A                                                                                  who don't like the hlgh-filtra-
              brand that's slipping keeps                                                                                 tion brands because they .
              changing the message.                                                                                       “hi-dly taste like anything ”
                          The longest-running cam-         Viceroy, for example, is now                                      On its two other brands, L&M :
                 paign in the cigarette business           coming at smokers with the . and Chesterfield, Liggett it .
                     is Marlboro's western motif,          line: “Why Viceroy? Because Myers seems to have just about .                                       1
                 and it will continue as long as           I'd never smoke a boring                                       thrown In the towel L&M, a ■
              the br and Cant tn ue s to move up           cigarette.'' Boring must mean                                  brand on the decline for many ,
                 on the sales chart. Marlboro is           those cigarette brands which years, is now being positioned
                 on the verge of overtaking Win-           are enjoying good sales.                                       as “Che proud smote/1 whatev- •
                ■ton as the top-selling brand in           So we have ads for True                                        er that means. And Qiesler-
                                     the country           stewing a smoker who says:                                     fietd, whose sates are In danger
                       , The Marlboro message is           •'Why all the talk about smok-                                 of disappearing, is using the .
                  clear; "Come to where the fla-           ing J dec ided Fd either quit or slogan that was a winner for it
              vor Is." What it's really saying,            smokeTrue. 1 smoke True ” Or to years ago, “They satisfy.” It  10 years ago. "Theysatisfy/' H
                  Jjeyond that, is: "This is not         • the ads for Vantage, which may be th=t the people who
                  one of those sissy tow-tar and           promise the flavor d a full-                                   remember that slogan are the
                 tow-olcotine brands This Is one           flavor cigarette “without any-                                 only ones still imoting the.
                                 you can taste."           where near the tat and brand. -
                . Winston is also in this “full-
                       flivor category. !t has 2
                 heavy slug of tar and nicotine,
                       even more than Marlboro
                     Since it has begun to lose
                   (ground to Marlboro, Winston
                     has changed the models who
                    appear In Its ads. They now
                 tend to be young people dressed                                                                                                                   /

                                                                                                                                                                   i
                                                                                                                                                                   i
```

Figure 7.9: Text file extracted using ABbyy Cloud OCR, method (3), with scaling

This may require the reader of the text file to be slightly more attentive to be able to separate them. This problem could, however, be overcome to some extent by increasing the scaling factor – a feature of Abbyy that magnifies the incoming PDF document by a multiple of the scaling factor – and reading the PDF.

Figure 7.10 show how each OCR technique performs on a sentence extracted from the newspaper article.



"The cigarette pushers are now into what the advertising industry calls "positioning," which means that a product tries to stake out an area for Itself in the smoker's mind.

```
~ TAe elgarelle pusAers ar
paw Intu wbal Pe advtttising
mdnary eall[ "po:uimmiL"
wMeh means Ual a proN[t
tries lo staYe eut an ara Ior
IuelnnNeamnlerlminJ.
```

```
~ TAe elgarelle pusAers ar
paw Intu wbal Pe advtttising
mdnary eall[ "po :uimmiL"
wMeh means Ual a proN[t
tries lo staYe eut an ara Ior
IuelnnNeamnlerlminJ .
```

```
The cigarette pushers are
t»w Into what the advertising
industry calls "positioning."
which means that a product
tries to state out an area for
Itself tn the smoker's mind.
```

Figure 7.10: Comparison between different OCR techniques on a sentence of the news article shown

Table 2.1 illustrates a comparative study among all the above discussed OCR techniques over a varied set of types of documents.

Table 7.9: Time and cost involved in implementing the different OCR techniques

	UCSF	PyPDF2	PDFMiner	Abbyy Cloud OCR
Time	Unknown	Good	Good	Average (depending upon server speed)
Cost	Unknown	None	None	Paid

Table 7.9 shows how time and cost may impact our choice of tool.

Hence, we may conclude that Abbyy Cloud OCR is an ideal choice if we have the resources. Otherwise, PDFMiner may be the optimal choice, given that it outperforms the UCSF's proprietary tool in scenarios involving two column PDFs. However, it should be noted, there are no other significant differences between the merits of those two methods.

7.7 File Storage System

We are using Ceph to store our JSON files, which is an open-source storage platform based on a single distributed computer cluster. All the recent data derived during the project are uploaded and stored there.

To access Ceph, ssh into tobacco.cs.vt.edu as user1 and run the following command in the terminal:

```
cd /mnt/ceph
```

You will find several folders under this directory. We, the CMT team, have used the folder ‘shared’ to store our JSON files and tobacco documents, under which there is a ‘tobacco’ folder. To view, use:

```
cd /mnt/ceph/shared/tobacco
```

Generally, to access a JSON or text file in the storage, make sure that an appropriate folder is opened. After that, run one of the following commands in order to open the file.

- **vim filename** (for OCR files without a file extension)
- **vim filename.json** (for JSON files)

To obtain a list of the current path’s objects, run **dir**. The storage is organized in the following hierarchy. There are 3 main directories:

1. **data**
2. **metadata_final**
3. **metadata_with_data_final**

Details are as follows.

1. **data:** This folder includes groups of OCR files used and/or produced during the project.
   - *1million_raw:* First 1 million OCR files obtained via sorting the whole set of documents by their IDs in database. Used for intermediate data preparation and testing of some of the algorithms.
   - *article_data:* All the OCR files of ’article’ type. Served as one of subsets of documents for cleaning

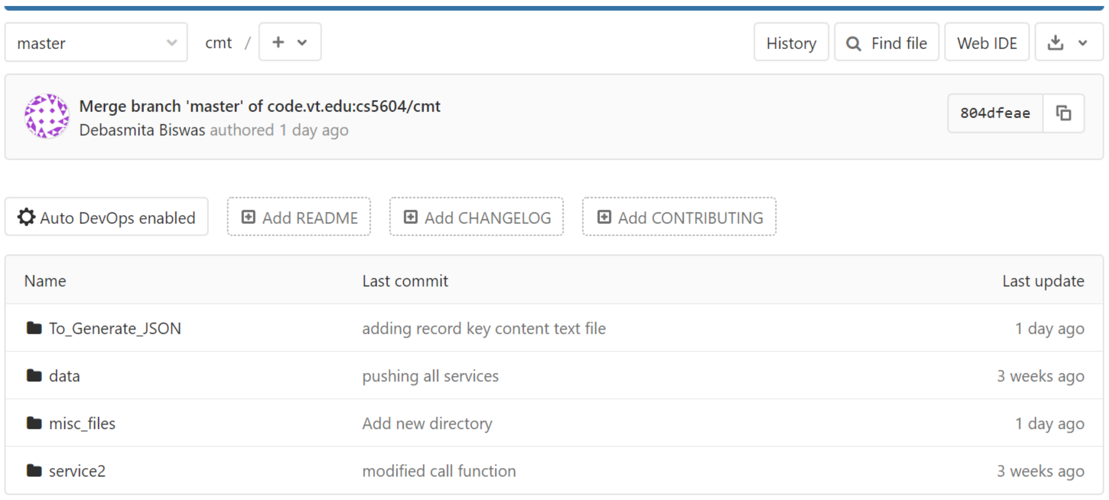
- *article_data_content:* Output of the cleaning algorithm derived after using a set of 'article' documents as an input
- *deposition_data:* An unprocessed set of all 'deposition' type documents. Used for intermediate file processing and examining their contents.

2. **metadata_final:** This directory comprises metadata JSON files for 10M documents of the tobacco dataset. There are 31 files overall; each file stores metadata for 500K documents sorted by corresponding document IDs in the database. See Section 5.2 for details.

3. **metadata_with_data_final:** Here, 3 JSON files embracing both metadata and text content of 100K documents are stored, each for 33K files (see Section 5.2 for details)

7.8 Code Repository

The final code used for generating the metadata and data has been put in a GitLab repository [5], under the project name 'CMT' [2]. The code is arranged in folders and should be easily navigable. Figure 7.11 shows a screenshot of the repository.



Merge branch 'master' of code.vt.edu:cs5604/cmt
Debasmita Biswas authored 1 day ago
804dfeae

Name	Last commit	Last update
To_Generate_JSON	adding record key content text file	1 day ago
data	pushing all services	3 weeks ago
misc_files	Add new directory	1 day ago
service2	modified call function	3 weeks ago

Figure 7.11: Screenshot of the CMT team's GitLab repository

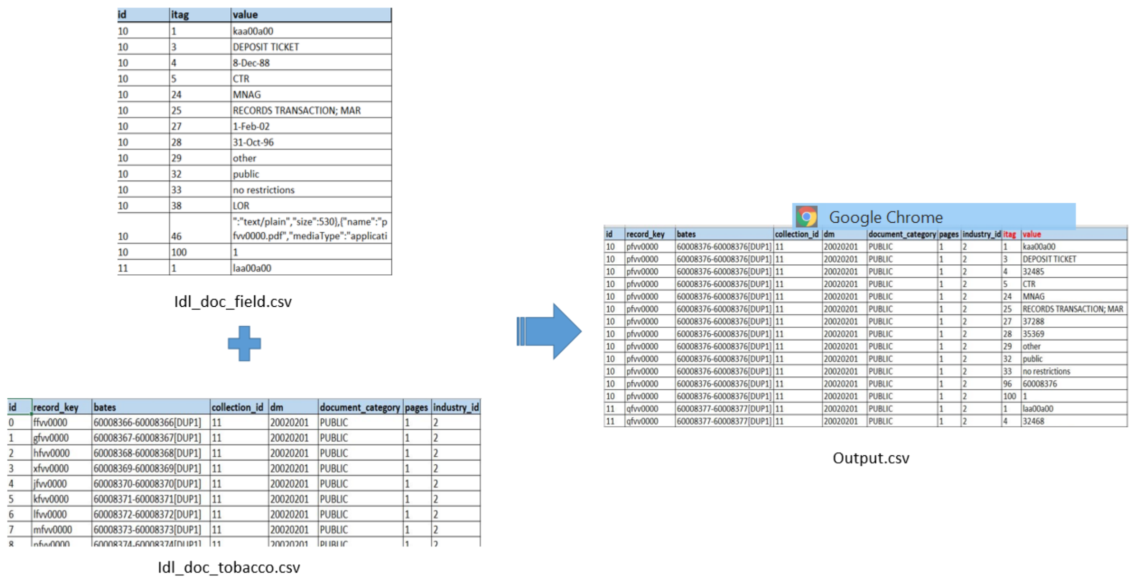
7.9 Ingestion of New Documents

In addition to the current dataset, it is anticipated that we may have to deal with a small number of files being added from time to time. It would have been fairly easy to write a script that would extract the new data from the database and parse it into JSON format. Finally, it could append the newly generated JSON file to our older version present in Ceph, which could in turn be accessed by the other teams on the project. However, the INT team would not have access to our Virtual Machine (tobacco.cs.vt.edu) and the database 'data'. The INT team also tried to set up Kafka, but could not do so within the available time. After weighing alternate options, it was proposed that, as a proof of concept, we create a solution to show that it is possible to create the JSON files even without a database.

1. Creation of a demo folder outside the VM containing instances of metadata files, PDFs, and text editable content of tobacco documents.
2. Write a Python script to pick metadata and data from these files and parse them into JSON format matching the one for tobacco files currently present on the VM.

We employed the following steps to achieve a solution.

1. To handle incoming PDFs of tobacco documents, we created a script that can download the PDF directly from the UCSF website and implement the OCR technique to generate a text editable file. We chose PDFMiner for this operation since it is free.
2. To generate the JSON of the metadata, we prepared a modified version of our Python scripts to:
    (a) Merge the CSV files of the tables idl_doc_tobacco and idl_doc_field based on common ID values; and export results into new CSV file 'output.csv'. Figure 7.12 shows a high level view.
    (b) Store itag descriptions in an array 'itag_array'
    (c) Pick data from the columns of 'output.csv' that contain the Itag and Value associated with a document ID; retrieve corresponding itag description based on itag_array.
    (d) Parse the data into desired JSON format and store into a new file 'output.json'.


Idl_doc_field.csv
Idl_doc_tobacco.csv
Google Chrome
Output.csv

id	itag	value
10	1	kaa00a00
10	3	DEPOSIT TICKET
10	4	8-Dec-88
10	5	CTR
10	24	MNAG
10	25	RECORDS TRANSACTION; MAR
10	27	1-Feb-02
10	28	31-Oct-96
10	29	other
10	32	public
10	33	no restrictions
10	38	LOR
10	46	":"text/plain","size":530},{"name":"p fvv0000.pdf","mediaType":"applicati
10	100	1
11	1	laa00a00

id	record_key	bates	collection_id	dm	document_category	pages	industry_id
0	ffvv0000	60008366-60008366[DUP1]	11	20020201	PUBLIC	1	2
1	gfvv0000	60008367-60008367[DUP1]	11	20020201	PUBLIC	1	2
2	hfvv0000	60008368-60008368[DUP1]	11	20020201	PUBLIC	1	2
3	xfvv0000	60008369-60008369[DUP1]	11	20020201	PUBLIC	1	2
4	jfvv0000	60008370-60008370[DUP1]	11	20020201	PUBLIC	1	2
5	kfvv0000	60008371-60008371[DUP1]	11	20020201	PUBLIC	1	2
6	lfvv0000	60008372-60008372[DUP1]	11	20020201	PUBLIC	1	2
7	mfvv0000	60008373-60008373[DUP1]	11	20020201	PUBLIC	1	2
8	nfvv0000	60008374-60008374[DUP1]	11	20020201	PUBLIC	1	2

id	record_key	bates	collection_id	dm	document_category	pages	industry_id	itag	value
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	1	kaa00a00
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	3	DEPOSIT TICKET
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	4	32485
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	5	CTR
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	24	MNAG
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	25	RECORDS TRANSACTION; MAR
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	27	37288
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	28	35369
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	29	other
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	32	public
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	33	no restrictions
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	96	60008376
10	pfvv0000	60008376-60008376[DUP1]	11	20020201	PUBLIC	1	2	100	1
11	qfvv0000	60008377-60008377[DUP1]	11	20020201	PUBLIC	1	2	1	laa00a00
11	qfvv0000	60008377-60008377[DUP1]	11	20020201	PUBLIC	1	2	4	32468

Figure 7.12: CSV files used to generate metadata.

The above solution has its own set of limitations viz. scalability issues, not implemented end-to-end since the Kafka integration by the INT team is incomplete and cannot be termed practical.

7.10 Using MySQL Workbench

Note: One may also use an interface to manipulate the MySQL database. Section 7.6 provides the instructions to connect to the MySQL Workbench.

The following instructions provide step-by-step guidance on how to connect the MySQL Workbench to the database containing all of the metadata stored at tobacco.cs.vt.edu. We suggest using the MySQL Workbench as one of the most convenient up-to-date MySQL database management tools.

1. Download MySQL Workbench installer by downloading the installation package from the following website. https://dev.mysql.com/downloads/workbench/ (make sure to choose an appropriate OS version)

2. Install MySQL Workbench and run the program.

3. Click on the “+” button right next to the "MySQL Connections".
4. You will see a new window with the "Setup New Connection" header. Input the following parameters.
    - Connection name: whatever name you want for the connection (e.g., tobacco).
    - Connection method: choose the "Standard TCP/IP over SSH" from the drop down menu.
    - Make sure that the "Parameters" tab is chosen.
    - SSH Hostname: tobacco.cs.vt.edu.
    - SSH Username: user1
    - SSH Password: click on "Store inVault", then input the password in an open window (contact Dr. Fox/Saurabh Chakrabarty for the password).
    - SSH Key File: leave blank
    - MySQL Hostname: leave the default value (127.0.0.1)
    - MySQL Server Port: leave the default value (3306)
    - Username: root
    - Password: leave unchanged
    - Default Schema: leave blank

    After all the parameters are set up, click on "Test connection". You will see a connection warning – just ignore it and click "OK". If the connection is set successfully, click "OK" in the previous window. After this step, the connection should be established.

Chapter 8

Future Work

This section discusses the future tasks with respect to processing the tobacco dataset.

8.1 Improve the Text Cleaning Process

We intend to improve the exisitng text cleaning process. The current process is good at identifying line numbers that are at the beginning of the line but fails to detect line numbers occurring in between sentences. We would like to improve this process to account for line numbers within the text as well as increase the precision of identifying garbage characters.

8.2 Finalize OCR Method

We have explored alternate OCR implementation techniques and have received preliminary results. The current OCR'ed documents have multiple non-decodable characters that require removal. We are removing such characters using a Python script in the preprocessing stage in order for data to be used for processing. Hence, we revisited the OCR process for alternative approaches that would result in better quality documents. A decision needs to be taken with respect to moving forward with an alternate OCR method, and this needs to be finalized and applied to documents.

8.3 Test Coverage

We intent to add more unit tests for the scripts written. This would help ensure that updates to the scripts do not break the existing functionality. This would also aid the initiative of building a CI/CD pipeline for the project.

8.4 Pipeline for Ingesting New Documents

We need to focus on automating the ingestion of new tobacco documents as and when they become available. This would involve coordinating with the INT team to set up a Kafka pipeline and prepare automation scripts to trigger the ingestion of new documents. We presently have a proof of concept that needs to be extended to build a full-fledged solution.

Chapter 9

Acknowledgements

This project has been completed during the course of CS 5604: Information Storage and Retrieval at Virginia Tech.

We would like to thank Dr. Edward A. Fox for his support and professional insights. He has been immensely helpful throughout the course of the project and has always helped unblock us whenever needed. We would also like to thank our Teaching Assistant, Ziqian Song, and our classmates for all the helpful suggestions we received. We appreciate the help received from Saurabh Chakravarty to get started on this project. We are grateful to Dr. Townsend for sharing his knowledge, thoughts, and experiences. We would also like to thank Chris Arnold and the Department of Computer Science for supporting the container cluster development environment. We further extend our appreciation to the technical support team at the UCSF Library for their assistance.

Bibliography

[1] ABBYY. Cloud OCR SDK, accessed on Nov. 18, 2019. https://www.ocrsdk.com/features/.

[2] ALON BENDELAC, DEBASMITA BISWAS, S. M. A. S. A. M. T. Y. Z. GitLab repository of CMT team, accessed on Dec. 20, 2019. https://code.vt.edu/cs5604/cmt.

[3] BITEXT. Difference between stemming and lemmatization, accessed on Sept. 07, 2019. https://blog.bitext.com/what-is-the-difference-between-stemming-and-lemmatization/.

[4] CHRISTOPHER D. MANNING, P. R., AND SCHÜTZE, H. *An Introduction to Information Retrieval.* Cambridge University Press, 2009.

[5] GITLAB. Setting up GitLab, accessed on Dec. 05, 2019. https://docs.gitlab.com/ee/gitlab-basics/start-using-git.html.

[6] JABEEN, H. Stemming and lemmatization in Python, accessed on Sept. 19, 2019. https://www.datacamp.com/community/tutorials/stemming-lemmatization-python.

[7] KHEMIRI, A. Pdf processing with Python, accessed on Oct. 22, 2019. https://towardsdatascience.com/pdf-preprocessing-with-python-19829752af9f.

[8] NICK ONOFRIO, NICK SORKIN, D. V. M. D. C. J., AND FOX, E. A. Tobacco Settlement Documents Report: Virginia Tech, accessed on Sept. 15, 2019. https://vtechworks.lib.vt.edu/handle/10919/91193.

[9] STEVEN BIRD, E. L. Natural Language Toolkit, accessed on Oct. 02, 2019. https://www.nltk.org/.

[10] Techopedia. Tokenization, accessed on Sept. 09, 2019, Oct. 2019. https://www.techopedia.com/definition/13698/tokenization.

[11] University of California, S. F. Tobacco Dataset and Tutorial, accessed on Oct. 10, 2019. https://ucsf.app.box.com/v/IDL-DataSets.

[12] University of California, S. F. Tobacco settlement dataset information: UCSF, accessed on Nov. 11, 2019. https://www.industrydocuments.ucsf.edu/tobacco/research-tools/field-names/.

[13] University of California, S. F. Truth Tobacco Industry Documents: UCSF, accessed on Oct. 17, 2019. https://www.industrydocuments.ucsf.edu/tobacco/.