Digging Deeper Reaching Further

Libraries Empowering Users to Mine the HathiTrust Digital Library Resources

Module 3: Working with Textual Data
Instructor Guide

Further reading: go.illinois.edu/ddrf-resources

Narrative

The following is a suggested narrative to accompany the slides. Please feel free to modify content as needed when delivering workshops. Additional information and examples that are provided to aid the instructor's understanding of the materials are in *Italic text* – whether to present them to workshop audiences is entirely up to the instructor.

Slide M3-1

In order to do text analysis, a researcher needs some proficiency in wrangling and cleaning textual data. This module address the skills needed to prepare text for analysis after it has been acquired.

Slide M3-2

Here is an overview of what we'll be covering in this module. We will think about what happens when text is data and consider common steps to prepare data for text analysis. In our hands-on activity, we will practice data cleaning on the speeches we scraped. Finally, in our case study, we will learn how Sam prepared his Creativity Corpus for analysis.

Slide M3-3

By the end of this module, you will learn how to run a Python script. The script you will run can change the text we scraped from WikiSource that is currently in HTML format to cleaner text ready to be analyzed.

Slide M3-4

In a humanities research setting, "Data" can be defined as material generated or collected while conducting research. *This definition is adapted from the National Endowment for Humanities' definition of "data" in their document Data Management Plans for NEH Office of Digital*

Humanities Proposals and Awards. Examples of humanities data are also provided in this document.

Humanities data may include citations, software code, algorithms, databases, geospatial coordinates (for example, from archaeological digs). We can see that humanities data can be very diverse and broad in scope.

Can you think of other examples?

(Answers can be: digital tools, documentation, reports, articles, etc.)

Slide M3-5

We're not always used to thinking about text as data, but text can be viewed as data and analyzed in certain ways in digital scholarship.

When approaching text as data, here are some things to keep in mind:

- First, having textual data of sufficient quality is important. Textual data quality is determined by how it's created. Hand-keyed text is often of the best quality, while text obtained by OCR, Optical Character Recognition, can vary in quality. Raw, uncorrected OCR text is dirty, and it can only become clean until it is corrected. Please note that HathiTrust OCR is dirty and uncorrected.
- When viewing text as data, we usually analyze them by corpus or corpora. As we mentioned in previous modules, a "corpus" of text can refer to both a digital collection and an individual's research text dataset. Text corpora are bodies of text.
- When preparing text, one can think in terms of what Geoffrey Rockwell has called text decomposition or re-composition. The text will be split, combined, or represented in ways that distinguish it from human readable text. It may involve discarding some text, and requires the researcher to shift their understanding of the text from human-legible object to data. What stays, what goes, and how things are manipulated is a researcher's choice. While there are emerging best practices, there isn't a step-by-step guide to follow.

Slide M3-6

After gathering the data needed for research and before conducting the actual analysis, data often requires preparation. Preparing data, which has been referred to as "janitorial work," takes a lot of time and effort.

Examples of what may be necessary to do before the data is in a workable state can include:

- Correcting OCR errors.
- Removing title and header information.
- Removing html or xml tags.
- Splitting or combining files.
- Removing certain words or punctuation marks.
- Making text into lowercase.

This is where scripting is helpful – a person isn't going to (or shouldn't) open every file one-by-one and do all this clean-up work manually.

Slide M3-7

Next, we will introduce some key concepts that researchers are likely to encounter while preparing text for analysis. As mentioned, preparing text often involves splitting and combining files. In text analysis, splitting files is commonly referred to as chunking text. It means splitting text into smaller pieces before analysis. The text may be divided by paragraph, chapter, or a chosen number of words (e.g. 1000 word chunks). Let's say that we have a whole text that consist of speeches of Abraham Lincoln. Before conducting analysis, the researcher may need to split the text into individual speeches. This process can be called chunking text.

Slide M3-8

An opposite process that needs to be done just as often is combining text into larger pieces before analysis, which can be referred to as grouping text. Let's look at political speeches as an example. Say that this time we have individual texts of various speeches made by Abraham Lincoln as well as George Washington. Before conducting our analysis, we may need to group the texts by combining all speeches by Lincoln into one group and all speeches by Washington into another group.

Both chunking (from the previous slide) and grouping are ways of modifying the unit of analysis for the researcher, and it's wholly dependent on what the researcher wants to study. Maybe someone wants to compare all of Abraham Lincoln to all of George Washington, then they could create two large "buckets" of data via chunking. Or someone only wants to compare the chapters in John F. Kennedy's "Profiles in Courage" to see how descriptions of the figures it profiled are similar or different, then a researcher might split a single work out by chapter. Those are simplistic examples, but they highlight the kinds of splitting and combining that may happen.

Slide M3-9

An additional step in preparation is called tokenization. Tokenization is simply the process of breaking text into pieces called tokens. Often certain characters, such as punctuation marks, are discarded in the process.

- Here's a tokenized version of the beginning of *The Gettysburg Address* on the slide. The original text, which is in a human-readable form, has been translated into tokens.
- While the tokens can still be parsed by a human, it isn't in a form we regularly read. It can now, however, be read and processed by a computer.

Slide M3-10

It is important to note that different choices in text preparation will affect the results of the analysis. Depending on the amount of text and size of chunks, which stop words are removed and which characters are included, and whether to lowercase and normalize words, the eventual text that is ready for analysis can be very different. Additionally, preparation for analysis takes a lot of time and effort. This is where scripting becomes useful!

Additional information about how text preparation impacts results:
https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2849145

Slide M3-11

Let's do an activity to learn more about data preparation techniques for text analysis. On your handout, under "Working with Text Data", you'll find a table with key approaches and their definitions. In small groups, divide the terms amongst yourselves, read the definitions and when finished, explain your terms to your partners.

These definitions come from an interesting paper titled "Text Preprocessing For Unsupervised Learning: Why It Matters, When It Misleads, And What To Do About It" (https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2849145)

(Instructor organizes the activity. If time runs short, can ask participants to take about 3 minutes to familiarize themselves with the concepts on their own instead of doing the 5-10-minute group discussion.)

Term	Definition
Punctuation	"The first choice a researcher must make when deciding how to preprocess a corpus is what classes of characters and markup to consider as valid text. The most inclusive approach is simply to choose to preprocess all text, including numbers, any markup (html) or tags, punctuation, special characters ($, %, &, etc), and extra white-space characters. These non-letter

	characters and markup may be important in some analyses (e.g. hashtags that occur in Twitter data), but are considered uninformative in many applications. It is therefore standard practice to remove them. The most common of these character classes to remove is punctuation."
Numbers	"While punctuation is often considered uninformative, there are certain domains where numbers may carry important information. For example, references to particular sections in the U.S. Code ('Section 423', etc.) in a corpus of Congressional bills may be substantively meaningful regarding the content legislation. However, there are other applications where the inclusion of numbers may be less informative."
Lowercasing	"Another preprocessing step taken in most applications is the lowercasing of all letters in all words. The rationale for doing so is that that whether or not the first letter of a word is uppercase (such as when that words starts a sentence) most often does not affect its meaning. For example, 'Elephant' and 'elephant' both refer to the same creature, so it would seem odd to count them as two separate word types for the sake of corpus analysis. However, there are some instances where a word with the same spelling may have two different meanings that are distinguished via capitalization, such as 'rose' (the flower), and 'Rose' the proper name."
Stemming	"The next choice a researcher is faced with in a standard text preprocessing pipeline is whether or not to stem words. Stemming refers to the process of reducing a word to its most basic form (Porter, 1980). For example the words 'party', 'partying', and 'parties' all share a common stem 'parti'. Stemming is often employed as a vocabulary reduction technique, as it combines different forms of a word together. However, stemming can sometimes combine together words with substantively different meanings ('college students partying', and 'political parties'), which might be misleading in practice."
Stopword Removal	"…some words, often referred to as "stop words", are unlikely to convey much information. These consist of function words such as 'the', 'it', 'and', and 'she', and may also include some domain-specific examples such as 'congress' in a corpus of U.S. legislative texts. There is no single gold-standard list of English stopwords, but most lists range between 100 and 1,000 terms."
n-gram Inclusion	"While it is most common to treat individual words as the unit of analysis, some words have a highly ambiguous meaning when taken out of context. For example the word 'national' has substantially different interpretations when used in the multi-word expressions: "national defense", and "national debt". This has led to a common practice of including n-grams from documents where an n-gram is a contiguous sequence of tokens of length n (Manning and Schutze, 1999). For example, the multi-word expression 'a common practice' from the previous sentence would be referred to as a 3-gram or tri-gram."
Infrequently Used Terms	"In addition to removing common stopwords, researchers often remove terms that appear very infrequently as part of corpus preprocessing. The rationale

	for this choice is often two-fold; (1) theoretically, if the researcher is interested in patterns of term usage across documents, very infrequently used terms will not contribute much information about document similarity. And (2) practically, this choice to discard infrequently used terms may greatly reduce the size of the vocabulary, which can dramatically speed up many corpus analysis tasks."

Slide M3-12

Next, we will be introducing Python in this module. Python is a commonly-used programming language, and it's very useful for working with data. It is also an interpreted language, which basically means it follows step-by-step directions. *You do not need to worry about compiling the program or making sure that the proper libraries are linked and loaded, like when writing with C or C++.*

Additionally, Python is generally easy to learn with its relatively simple syntax. For example, one of its learner-friendly features is it avoids excess punctuation.

Slide M3-13

We can use Python to do interactive programming in an interactive mode. This is done in the Python interpreter. Each step is input and run via the console, and immediate feedback is given for each statement.

Slide M3-14

We can also use Python to write and run scripts. As we mentioned very briefly in module 2.2, scripts are basically text files containing programing statements – they are a set of directions. Once you have written something in Python and saved it as a script, you can execute it as many times as you like without having to retype it each time.

Python scripts are saved as files ending with the extension ".py". After you have saved a script, you can run it using the command line.

We will be running Python scripts for the activities in our workshop.

Slide M3-15

When you run a Python script from the command line, you follow a standard syntax, as shown on the screen.

- First, you write “python” to tell the computer to get ready for Python. It lets the computer know what language you are speaking.
- Then you write the script filename, ending in .py, so that the computer knows which set of directions to follow.
- Finally, some scripts will call for “arguments” or additional information to be supplied on the command line to run.
- An example is “python myscript.py newfile.txt” – we can imagine this script will run myscript.py and create some output called newfile.txt based on the directions in the script.

Slide M3-16

In the next section, we will return to our sample reference question. In Module 2, we have already scraped one presidential speech from WikiSource. However, this text still needs to be prepared before we can start using it for analysis.

Slide M3-17

In this hands-on activity, we’re going to practice data clean up. We will use a Python script to work with the previously scraped *Fourth State of the Union Address* and prepare it for analysis. The script we’re going to run will take out the HTML tags from the file you scraped from WikiSource earlier.

(Instructor should demo this segment live if possible and use the screenshots on the slides as backup)

Slide M3-18

For this activity, we will be using PythonAnywhere, the txt text file from our web scraping activity, and the remove_tag.py Python script.

Slide M3-19

First, log in to PythonAnywhere. On your Dashboard, click on the “Browse files” button in the Files column, or click on the “Files” option in the upper right corner of the page. This will bring you to your files page.

Slide M3-20

Click on the name of the file, remove_tag.py, so we can see what it looks like.

- You don’t need to be able to read the Python code in order to do the activity. A few key points of the code are highlighted on the slide.

- First, we see the script calls for “sys.argv[1]” and that means we need to pass the script an argument, in this case something called “file_name” which is the file we want to clean.
- Then we see that we are going to write to a new file called “tagless_file.txt”

Instructors may choose to explain why throughout the workshop we will be providing specific file names for some activities and just directories for other activities. This is because for Python scripts, the arguments that are required are dictated by the script. Some of the scripts that we will be using have specific file names written in the code, so we can only use or generate file names as the script dictates. For other scripts or commands, there is more flexibility and we won’t need to adhere to specific file names.

Slide M3-21

Next, open up a Bash shell. We will now run the Python script. Enter the command “`python remove_tag.py washington_4.txt`” and press the return key. If no additional warnings appear, you have run the script successfully.

Slide M3-22

Click on the PythonAnywhere logo in the top left to return to your dashboard. Now we can locate the new cleaned-up version of the scraped text file. Click on the “Files” tab to go to your file list. The cleaned-up file will appear as “tagless_file.txt”. Click on the file to review the results.

Slide M3-23

As we scroll through the text (if your text is in one line, scroll from left to right), we can see that the HTML tags have been successfully removed. If it is too difficult to read the text in one line, you can go to your Bash console and use “less tagless_file.txt” to view the file content in your console instead. Press “q” to quit viewing.

Slide M3-24

What happened? This script looked for HTML open and close tags (the carrots), and wrote the text that was not the tags and not within the tags to a file.

- This script gets the job done for this example, but it’s not the most elegant or robust solution!
- **IMPORTANT:** In the “real world” there are better tools out there for cleaning HTML, but sometimes you need to go for the simplest thing and not the most sophisticated thing in text analysis – so take this as an example of that!

Slide M3-25

Hands-on activity:

Can you run a second pre-processing script to remove stopwords?

The remove_tag.py script we just ran required only a single argument, which was for the file to remove the tags from (washington_4.txt). The remove_stopwords.py script requires three arguments:

- The input file (your tagless file)
- The list of stop words (stopwords.txt)
- The output file name (you make this up!)

Remember to put a space between each argument.

Then, can you edit the stopwords.txt file to customize your list? Are there any stop words you think would be important to remove from this text?

Instructors may conclude the exercise by asking people how they have done it and then show how to run it on the big screen.

Slide M3-26

Now let's see how Sam handled the pre-processing of his Creativity Corpus.

- In preparation for analysis, he discarded (deleted) all of the pages in the corpus that did not include a form of creativ*
    - He was only interested in knowing what concepts were talked about around creative, so he wanted to retain only text in close proximity in the volume
- He also discarded (deleted) certain tokens, such as pronouns and conjunctions to keep only the most "meaningful" terms.
- Sam did not do additional OCR cleaning to his corpus. That kind of pre-processing is at the discretion of the researcher, and depends on how much noise or messiness he was willing to tolerate and often relates to the size of the corpus and the quality at which the data started. One might be willing to tolerate more noise in a larger corpus, or be more interested in cleaning OCR that is below a certain quality. It need not always be "clean" before analysis.

Slide M3-27

Katie Rawson and Trevor Munoz published a piece called “Against Cleaning” that theorized data preparation in the humanities, and presents their ideas for best practice in standardizing humanities data. They suggest a strategy for dealing with humanities data that takes into account the kind of non-standard menu data they dealt with during the “What’s on the Menu” Project, a crowdsourced-transcription project. Some of those suggestions are on the screen.

Slide M3-28

Let’s read an excerpt from the piece, which can be found on your handout or the screen.

When humanities scholars recoil at data-driven research, they are often responding to the reductiveness inherent in this form of scholarship. This reductiveness can feel intellectually impoverishing to scholars who have spent their careers working through particular kinds of historical and cultural complexity… From within this worldview, data cleaning is then maligned because it is understood as a step that inscribes a normative order by wiping away what is different. The term “cleaning” implies that a data set is “messy.” “Messy” suggests an underlying order. It supposes things already have a rightful place, but they’re not in it—like socks on the bedroom floor rather than in the wardrobe or the laundry hamper.
-- Katie Rawson and Trevor Muñoz, “Against Cleaning,”
http://curatingmenus.org/articles/against-cleaning/

Slide M3-29

We have time now to reflect in small groups about what it means to “clean” data in the humanities, and how you might broach the topic with scholars on your campus.

- What does this excerpt suggest about the nuances of data cleaning?
- What does “clean” imply?
- How might you talk to researchers on your campus who would be uncomfortable with the idea of clean v. messy data?

(This is should be a small group discussion for 5-10 minutes)

Slide M3-30

That’s all we have for this lesson, and we will be happy to take any questions from you.

Slide M3-31
(Show references so attendees know where to find them later.)