Final Project : Documentation

MOOC Lecture Video and Content Integration with Associated Textbook

1. Team Information

NetID : jaecho2@illinois.edu   Team Member Count : 1
Captain : jaecho2@illinois.edu

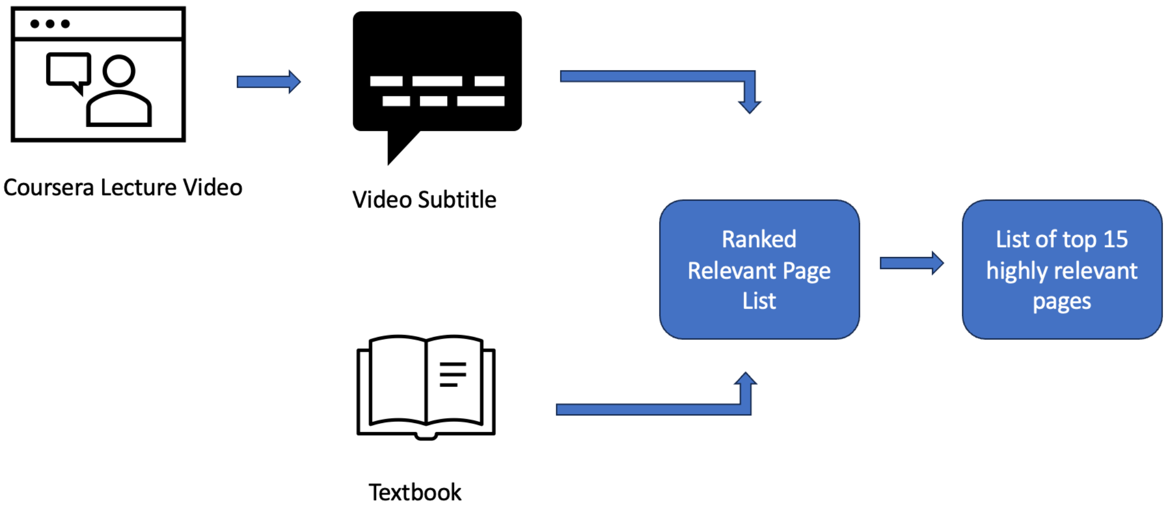
2. Theme of the Project

As students we all dream of a world where we can read the associated textbook cover to cover (or perhaps just me) so that each point mentioned in the lectures can be corelated with a specific section within the text. This has the benefit of reinforcing the learning experience.

The problem is that we rarely have the luxury of time to read a textbook cover to cover, which may lead us to overlook valuable information. This project is an attempt to assist students with a tool that will build the association between a textbook and MOOC lecture, so that the full power of a textbook can be utilized to enhance and expand the learning experience.

3. Architecture and Design

The project consists of the following input and output.
Input : Video Subtitle File selected by the user as the query list.
Input : Textbook for the course used as the document list.
Output : List of Top Ranked pages considered to be relevant to the current lecture video that the student is learning from.


Coursera Lecture Video
Video Subtitle
Ranked Relevant Page List
List of top 15 highly relevant pages
Textbook

4. Use Case

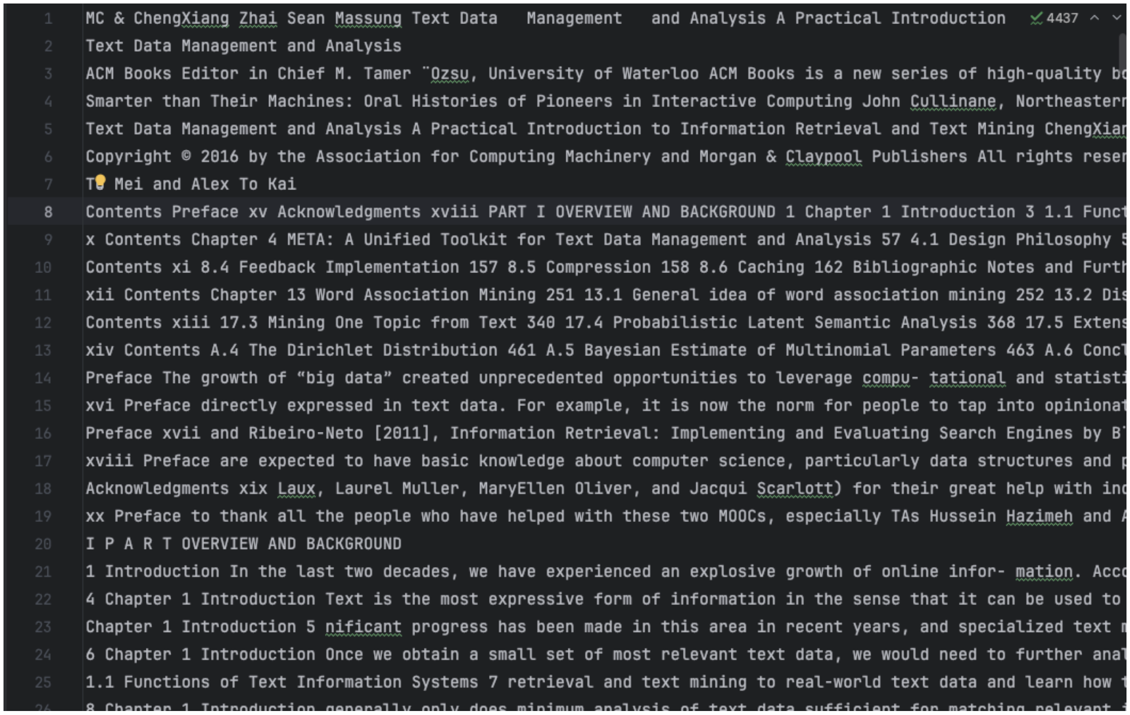
[STUDENT] Student watches a Lecture video that is provided in Coursera.
After completing the Video student want to also review relevant section in the associated textbook to gain additional understanding and clarity.
[STUDENT] Scans the textbook to determine where the relevant section is that covers all the terms and concepts that the professor mentioned in class.
[STUDENT] Student can scan and pickup a couple of relevant sections but is afraid that he/she might have missed something that is relevant.
[STUDENT]Student submits the textbook and subtitles for the lecture that was just completed.
[SYSTEM]System takes this information and creates a ranked list of most probable pages associated with the video.

5. Code Documentation

```
def extract_and_save_text(pdf_path, output_dir):
```

This function takes a path to a textbook as an input and stores the information in a file called ‘text_data.dat’. Each row in ‘text_data.dat’ pertains to a single page in the textbook.

Below is a sample screenshot of the content of ‘text_data.dat’.

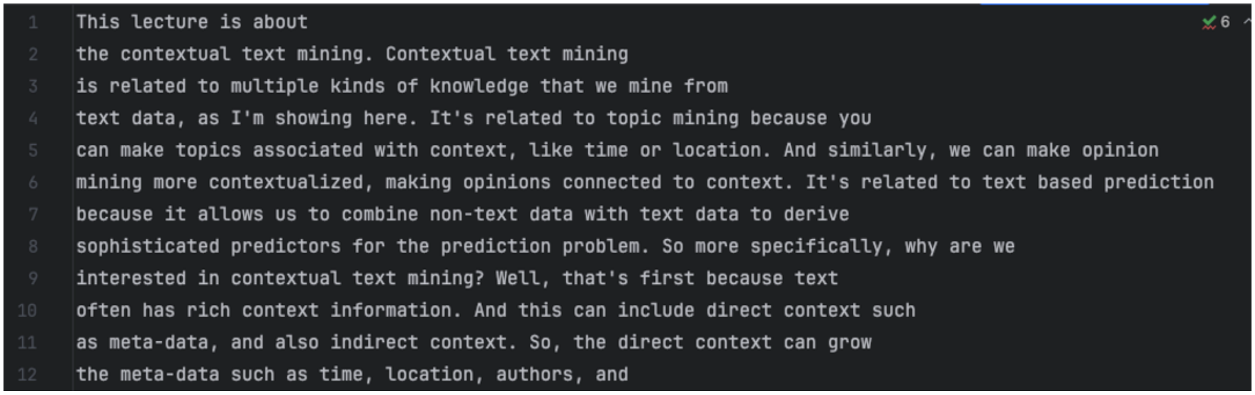


```
MC & ChengXiang Zhai Sean Massung Text Data   Management   and Analysis A Practical Introduction
Text Data Management and Analysis
ACM Books Editor in Chief M. Tamer ¨Ozsu, University of Waterloo ACM Books is a new series of high-quality bo
Smarter than Their Machines: Oral Histories of Pioneers in Interactive Computing John Cullinane, Northeastern
Text Data Management and Analysis A Practical Introduction to Information Retrieval and Text Mining ChengXian
Copyright © 2016 by the Association for Computing Machinery and Morgan & Claypool Publishers All rights reser
T Mei and Alex To Kai
Contents Preface xv Acknowledgments xviii PART I OVERVIEW AND BACKGROUND 1 Chapter 1 Introduction 3 1.1 Funct
x Contents Chapter 4 META: A Unified Toolkit for Text Data Management and Analysis 57 4.1 Design Philosophy 5
Contents xi 8.4 Feedback Implementation 157 8.5 Compression 158 8.6 Caching 162 Bibliographic Notes and Furth
xii Contents Chapter 13 Word Association Mining 251 13.1 General idea of word association mining 252 13.2 Dis
Contents xiii 17.3 Mining One Topic from Text 340 17.4 Probabilistic Latent Semantic Analysis 368 17.5 Extens
xiv Contents A.4 The Dirichlet Distribution 461 A.5 Bayesian Estimate of Multinomial Parameters 463 A.6 Concl
Preface The growth of “big data” created unprecedented opportunities to leverage compu- tational and statisti
xvi Preface directly expressed in text data. For example, it is now the norm for people to tap into opinionat
Preface xvii and Ribeiro-Neto [2011], Information Retrieval: Implementing and Evaluating Search Engines by B
xviii Preface are expected to have basic knowledge about computer science, particularly data structures and p
Acknowledgments xix Laux, Laurel Muller, MaryEllen Oliver, and Jacqui Scarlott) for their great help with inc
xx Preface to thank all the people who have helped with these two MOOCs, especially TAs Hussein Hazimeh and A
I P A R T OVERVIEW AND BACKGROUND
1 Introduction In the last two decades, we have experienced an explosive growth of online infor- mation. Acco
4 Chapter 1 Introduction Text is the most expressive form of information in the sense that it can be used to
Chapter 1 Introduction 5 nificant progress has been made in this area in recent years, and specialized text m
6 Chapter 1 Introduction Once we obtain a small set of most relevant text data, we would need to further anal
1.1 Functions of Text Information Systems 7 retrieval and text mining to real-world text data and learn how t
8 Chapter 1 Introduction generally only does minimum analysis of text data sufficient for matching relevant
```

```
def load_subtitle():
```

This function takes a path to a subtitle file as an input and stores it as a list of queries in a file called `subtitle-queries.txt`. The subtitle file must be in a txt file format.

Below is a sample screenshot of the content of ‘`subtitle-queries.txt`.



```
This lecture is about
the contextual text mining. Contextual text mining
is related to multiple kinds of knowledge that we mine from
text data, as I'm showing here. It's related to topic mining because you
can make topics associated with context, like time or location. And similarly, we can make opinion
mining more contextualized, making opinions connected to context. It's related to text based prediction
because it allows us to combine non-text data with text data to derive
sophisticated predictors for the prediction problem. So more specifically, why are we
interested in contextual text mining? Well, that's first because text
often has rich context information. And this can include direct context such
as meta-data, and also indirect context. So, the direct context can grow
the meta-data such as time, location, authors, and
```

```
def perform_search():
```

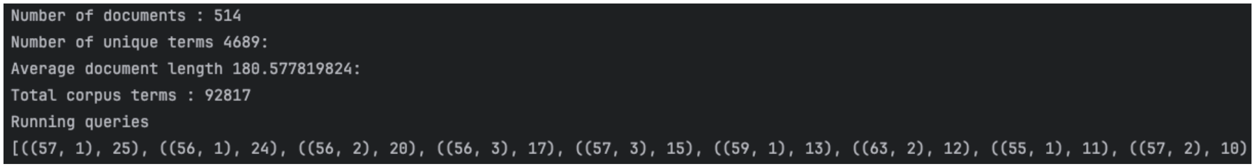
This function does the heavy lifting by indexing the textbook file ‘text_data.dat’ and ranking them based on the queries stored in ‘subtitle-queries.txt’

From the list of rankers available in MeTA, Okapi BM25 was used.

The logic to decide relevant document is as below.

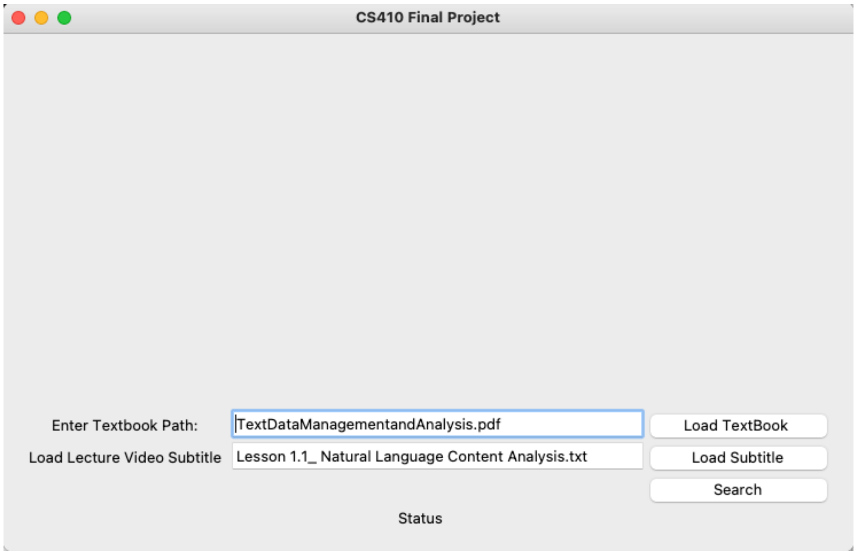
- Go through each query topic in subtitle-queries.txt.
- Pic the top 5 ranked documents from this list
- This will be stored in a list called *ranked_doc_ids*
- Pick the 15 most common documents in *ranked_doc_ids*
- Present the top 15 documents as the relevant documents that students should review for the video lecture.

A sample output is below



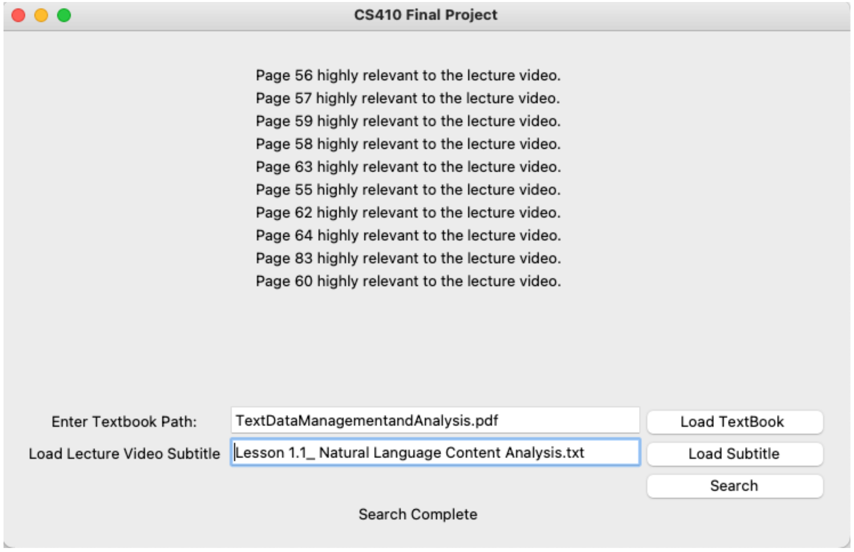
```
Number of documents : 514
Number of unique terms 4689:
Average document length 180.577819824:
Total corpus terms : 92817
Running queries
[((57, 1), 25), ((56, 1), 24), ((56, 2), 20), ((56, 3), 17), ((57, 3), 15), ((59, 1), 13), ((63, 2), 12), ((55, 1), 11), ((57, 2), 10)
```

Also added a nice GUI so that user can easily change the input files.



Enter Textbook Path: TextDataManagementandAnalysis.pdf

Load Lecture Video Subtitle Lesson 1.1_ Natural Language Content Analysis.txt



```
Page 56 highly relevant to the lecture video.
Page 57 highly relevant to the lecture video.
Page 59 highly relevant to the lecture video.
Page 58 highly relevant to the lecture video.
Page 63 highly relevant to the lecture video.
Page 55 highly relevant to the lecture video.
Page 62 highly relevant to the lecture video.
Page 64 highly relevant to the lecture video.
Page 83 highly relevant to the lecture video.
Page 60 highly relevant to the lecture video.
```

Enter Textbook Path: TextDataManagementandAnalysis.pdf

Load Lecture Video Subtitle Lesson 1.1_ Natural Language Content Analysis.txt

Search Complete

6. Challenges

4.1 Package unable to parse PDF properly.

The biggest initial challenge was in using the correct library that can parse the textbook which is in pdf format.

First attempt was to use *PyPDF2,* but faced problems because each word was not parsed properly with space in between. Some of the words were but majority of the words were all appended together without any spaces.

Second attempt was to try a different library called *pdfPlumber*, this provided marginally better results. Even after trying to parse the text with regular expression did not work well.

Third attempt was to experiment with a library called FITZ. Based on a number of testing this worked well.

4.2 Combining all documents into a single .dat file.

Initially I tried to create a separate .dat file for each page in the textbook. This would mean I had a total of 514 pages. So each page would be a document within the data source. I had a lot of trouble working with the ranking function and trying to combine the results based on each individual document. To overcome this I combined all of the documents into a single text_data.dat file and had each row represent a page in the textbook.

4.3 Using legacy Python version 2.7 for compatibility issues with MeTA

Using anaconda, I created 2 environment one based on Python 3.11 and another based on 2.7. I wanted to create nice bar charts to present the user with the ranking results. I found matplotlib does not work well in Python 2.7 and MeTA does not work at all in Python 3.11. The only option I had was to stick with Python 2.7 and use text based results.

7. Validation

For validation of the rank results and relevancy, human intervention was required. Through the 3 test scenarios that were run I rated the system on Low, Medium, and High.

- Low : 20% accuracy of the listed pages on relevancy to the lecture video.
- Medium : 20% ~ 80% accuracy of the listed pages on relevancy to the lecture video.
- High : over 80% accuracy of the listed pages on relevancy to the lecture video.

Test Case 1

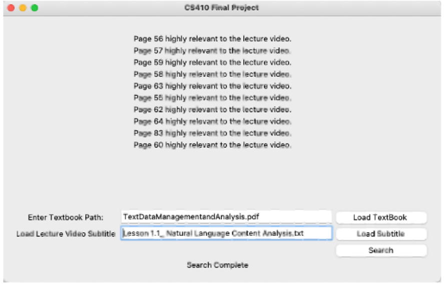
Input :

Document : Text Data Management and Analysis.pdf (The Textbook)

Query : Lesson 1.1_ Natural Language Content Analysis.txt (The subtitles for the video)

Expected Output : The first lecture video and subtitles were chosen as the input. Naturally since this is the first lecture, student would expect that the relevant textbook content should be in the beginning chapters.

Output : Result is High : The outcome show that most of the contents on the listed pages are indeed about the content presented in the video.

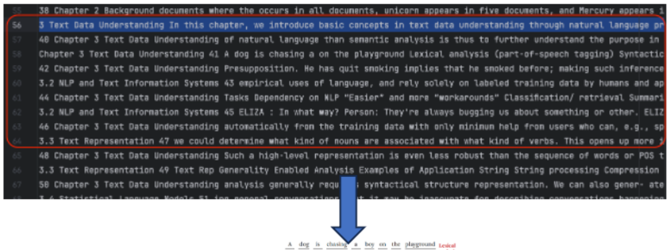


```
Page 56 highly relevant to the lecture video.
Page 57 highly relevant to the lecture video.
Page 59 highly relevant to the lecture video.
Page 58 highly relevant to the lecture video.
Page 63 highly relevant to the lecture video.
Page 55 highly relevant to the lecture video.
Page 62 highly relevant to the lecture video.
Page 64 highly relevant to the lecture video.
Page 83 highly relevant to the lecture video.
Page 60 highly relevant to the lecture video.
```

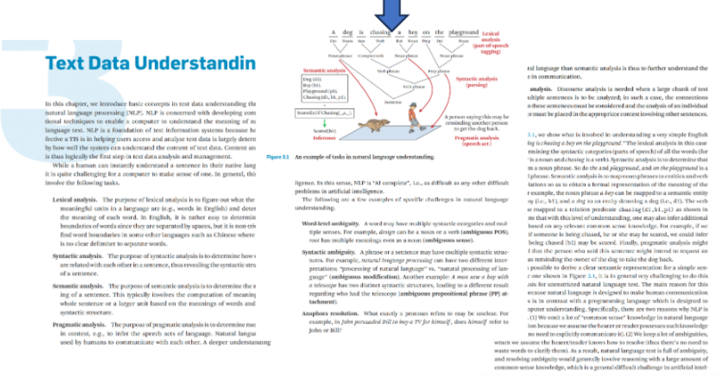
Enter Textbook Path: TextDataManagementandAnalysis.pdf

Load Lecture Video Subtitle: Lesson 1.1_ Natural Language Content Analysis.txt

Search Complete



```
55  38 Chapter 2 Background documents where the occurs in all documents, unicorn appears in five documents, and Mercury appears i
56  3 Text Data Understanding In this chapter, we introduce basic concepts in text data understanding through natural language pr
57  40 Chapter 3 Text Data Understanding of natural language than semantic analysis is thus to further understand the purpose in
58  Chapter 3 Text Data Understanding 41 A dog is chasing a on the playground Lexical analysis (part-of-speech tagging) Syntactic
59  42 Chapter 3 Text Data Understanding Presupposition. He has quit smoking implies that he smoked before; making such inference
60  3.2 NLP and Text Information Systems 43 empirical uses of language, and rely solely on labeled training data by humans and ap
61  44 Chapter 3 Text Data Understanding Tasks Dependency on NLP "Easier" and more "workarounds" Classification/ retrieval Summari
62  3.2 NLP and Text Information Systems 45 ELIZA : In what way? Person: They're always bugging us about something or other. ELIZ
63  46 Chapter 3 Text Data Understanding automatically from the training data with only minimum help from users who can, e.g., sp
64  3.3 Text Representation 47 we could determine what kind of nouns are associated with what kind of verbs. This opens up more i
65  48 Chapter 3 Text Data Understanding Such a high-level representation is even less robust than the sequence of words or POS t
66  3.3 Text Representation 49 Text Rep Generality Enabled Analysis Examples of Application String String processing Compression
67  50 Chapter 3 Text Data Understanding analysis generally requ  s syntactical structure representation. We can also gener- ate
```



Text Data Understandin

Figure 3.1 An example of tasks in natural language understanding.

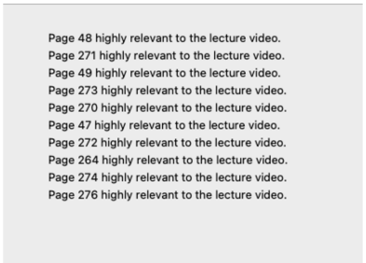
Test Case 2

Input :

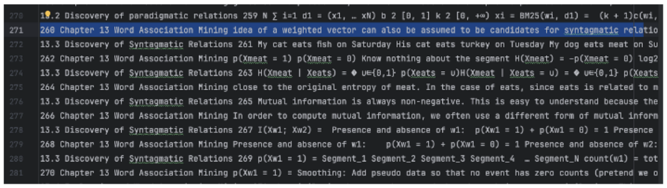
Document : Text Data Management and Analysis.pdf (The Textbook)

Query : Lesson 8.1 Syntagmatic Relation Discovery_ Entropy.txt (The subtitles for the video)

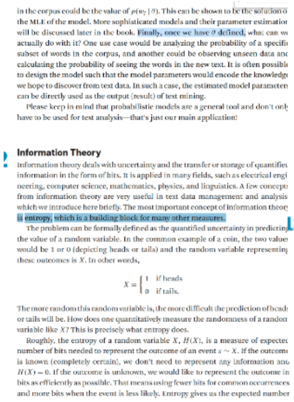
Output : Result is High : The outcome show that most of the contents on the listed pages are indeed about the content presented in the video.



Page 48 highly relevant to the lecture video.
Page 271 highly relevant to the lecture video.
Page 49 highly relevant to the lecture video.
Page 273 highly relevant to the lecture video.
Page 270 highly relevant to the lecture video.
Page 47 highly relevant to the lecture video.
Page 272 highly relevant to the lecture video.
Page 264 highly relevant to the lecture video.
Page 274 highly relevant to the lecture video.
Page 276 highly relevant to the lecture video.



```
13.2 Discovery of paradigmatic relations 259 N ∑ i=1 d1 = (x1, … xN) b 2 [0, 1] k 2 [0, +∞) xi = BM25(wi, d1) =  (k + 1)c(wi,
260 Chapter 13 Word Association Mining idea of a weighted vector can also be assumed to be candidates for syntagmatic relatio
13.3 Discovery of Syntagmatic Relations 261 My cat eats fish on Saturday His cat eats turkey on Tuesday My dog eats meat on Su
262 Chapter 13 Word Association Mining p(Xmeat = 1) p(Xmeat = 0) Know nothing about the segment H(Xmeat) = −p(Xmeat = 0) log2
13.3 Discovery of Syntagmatic Relations 263 H(Xmeat | Xeats) = ◆ u∈{0,1} p(Xeats = u)H(Xmeat | Xeats = u) = ◆ u∈{0,1} p(Xeats
264 Chapter 13 Word Association Mining close to the original entropy of meat. In the case of eats, since eats is related to m
13.3 Discovery of Syntagmatic Relations 265 Mutual information is always non-negative. This is easy to understand because the
266 Chapter 13 Word Association Mining In order to compute mutual information, we often use a different form of mutual inform
13.3 Discovery of Syntagmatic Relations 267 I(Xw1; Xw2) =  Presence and absence of w1:  p(Xw1 = 1) + p(Xw1 = 0) = 1 Presence
268 Chapter 13 Word Association Mining Presence and absence of w1:    p(Xw1 = 1) + p(Xw1 = 0) = 1 Presence and absence of w2:
13.3 Discovery of Syntagmatic Relations 269 p(Xw1 = 1) = Segment_1 Segment_2 Segment_3 Segment_4  … Segment_N count(w1) = tot
270 Chapter 13 Word Association Mining p(Xw1 = 1) = Smoothing: Add pseudo data so that no event has zero counts (pretend we o
```



in the corpus could be the value of $p(w_i \mid \theta)$. This can be shown to be the solution o the MLE of the model. More sophisticated models and their parameter estimatior will be discussed later in the book. Finally, once we have θ defined, what can w actually do with it? One use case would be analyzing the probability of a specifi subset of words in the corpus, and another could be observing unseen data anc calculating the probability of seeing the words in the new text. It is often possibl to design the model such that the model parameters would encode the knowledg we hope to discover from text data. In such a case, the estimated model parameter can be directly used as the output (result) of text mining.

Please keep in mind that probabilistic models are a general tool and don't onl have to be used for text analysis—that's just our main application!

Information Theory

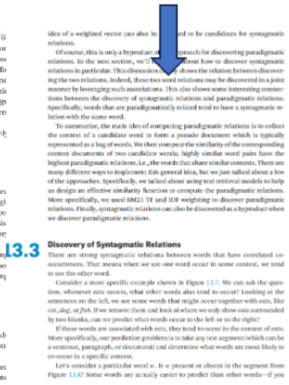
Information theory deals with uncertainty and the transfer or storage of quantifie information in the form of bits. It is applied in many fields, such as electrical engi neering, computer science, mathematics, physics, and linguistics. A few concept from information theory are very useful in text data management and analysis which we introduce here briefly. The most important concept of information theor is entropy, which is a building block for many other measures.

The problem can be formally defined as the quantified uncertainty in predictin the value of a random variable. In the common example of a coin, the two value would be 1 or 0 (depicting heads or tails) and the random variable representin these outcomes is X. In other words,

$$X = \begin{cases} 1 & \text{if heads} \\ 0 & \text{if tails.} \end{cases}$$

The more random this random variable is, the more difficult the prediction of head or tails will be. How does one quantitatively measure the randomness of a randon variable like X? This is precisely what entropy does.

Roughly, the entropy of a random variable X, $H(X)$, is a measure of expecte number of bits needed to represent the outcome of an event $x \sim X$. If the outcom is known (completely certain), we don't need to represent any information an $H(X) = 0$. If the outcome is unknown, we would like to represent the outcome in bits as efficiently as possible. That means using fewer bits for common occurrences and more bits when the event is less likely. Entropy gives us the expected number



idea of a weighted vector can also be [illegible]ed to be candidates for syntagmatic relations.

Of course, this is only a byproduct o[illegible]pproach for discovering paradigmatic relations. In the next section, we'll [illegible] about how to discover syntagmatic relations in particular. This discussion c[illegible]y shows the relation between discovering the two relations. Indeed, these two word relations may be discovered in a joint manner by leveraging such associations. This also shows some interesting connections between the discovery of syntagmatic relations and paradigmatic relations. Specifically, words that are paradigmatically related tend to have a syntagmatic relation with the same word.

To summarize, the main idea of computing paradigmatic relations is to collect the context of a candidate word to form a pseudo document which is typically represented as a bag of words. We then compute the similarity of the corresponding context documents of two candidate words; highly similar word pairs have the highest paradigmatic relations, i.e., the words that share similar contexts. There are many different ways to implement this general idea, but we just talked about a few of the approaches. Specifically, we talked about using text retrieval models to help us design an effective similarity function to compute the paradigmatic relations. More specifically, we used BM25 TF and IDF weighting to discover paradigmatic relations. Finally, syntagmatic relations can also be discovered as a byproduct when we discover paradigmatic relations.

13.3 Discovery of Syntagmatic Relations

There are strong syntagmatic relations between words that have correlated co-occurrences. That means when we see one word occur in some context, we tend to see the other word.

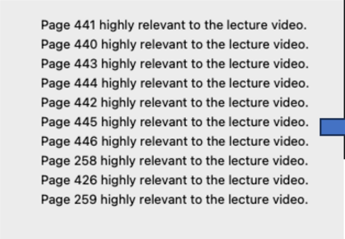
Consider a more specific example shown in Figure 13.7. We can ask the question, whenever *eats* occurs, what other words also tend to occur? Looking at the sentences on the left, we see some words that might occur together with *eats*, like *cat*, *dog*, or *fish*. If we remove them and look at where we only show *eats* surrounded by two blanks, can we predict what words occur to the left or to the right?

If these words are associated with *eats*, they tend to occur in the context of *eats*. More specifically, our prediction problem is to take any text segment (which can be a sentence, paragraph, or document) and determine what words are most likely to co-occur in a specific context.

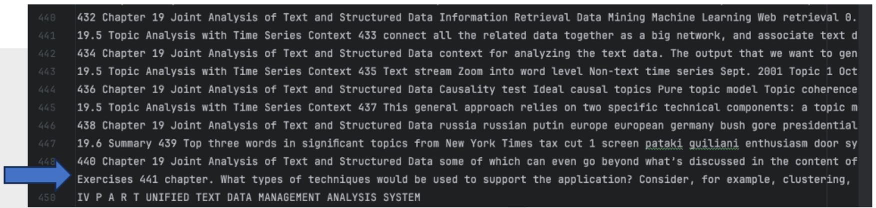
Let's consider a particular word w. Is w present or absent in the segment from Figure 13.8? Some words are actually easier to predict than other words—if you

Test Case 3
Input :
Document : Text Data Management and Analysis.pdf (The Textbook)
Query : Lesson 12.7 Contextual Text Mining_ Mining Causal Topics with Time Series Supervision.txt (The subtitles for the video)

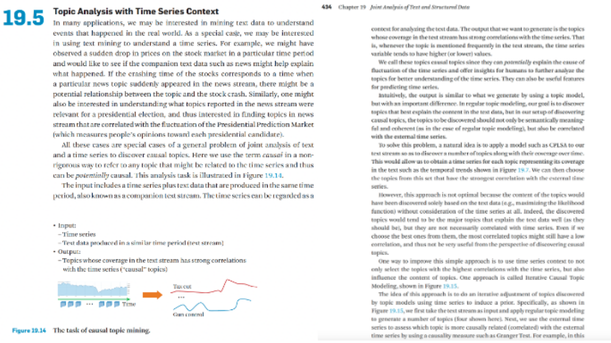
Output : Result is High : The outcome show that most of the contents on the listed pages are indeed about the content presented in the video.



```
Page 441 highly relevant to the lecture video.
Page 440 highly relevant to the lecture video.
Page 443 highly relevant to the lecture video.
Page 444 highly relevant to the lecture video.
Page 442 highly relevant to the lecture video.
Page 445 highly relevant to the lecture video.
Page 446 highly relevant to the lecture video.
Page 258 highly relevant to the lecture video.
Page 426 highly relevant to the lecture video.
Page 259 highly relevant to the lecture video.
```



```
432 Chapter 19 Joint Analysis of Text and Structured Data Information Retrieval Data Mining Machine Learning Web retrieval 0.
19.5 Topic Analysis with Time Series Context 433 connect all the related data together as a big network, and associate text d
434 Chapter 19 Joint Analysis of Text and Structured Data context for analyzing the text data. The output that we want to gen
19.5 Topic Analysis with Time Series Context 435 Text stream Zoom into word level Non-text time series Sept. 2001 Topic 1 Oct
436 Chapter 19 Joint Analysis of Text and Structured Data Causality test Ideal causal topics Pure topic model Topic coherence
19.5 Topic Analysis with Time Series Context 437 This general approach relies on two specific technical components: a topic m
438 Chapter 19 Joint Analysis of Text and Structured Data russia russian putin europe european germany bush gore presidential
19.6 Summary 439 Top three words in significant topics from New York Times tax cut 1 screen pataki guiliani enthusiasm door sy
440 Chapter 19 Joint Analysis of Text and Structured Data some of which can even go beyond what's discussed in the content of
Exercises 441 chapter. What types of techniques would be used to support the application? Consider, for example, clustering,
IV P A R T UNIFIED TEXT DATA MANAGEMENT ANALYSIS SYSTEM
```



19.5 Topic Analysis with Time Series Context

In many applications, we may be interested in mining text data to understand events that happened in the real world. As a special case, we may be interested in using text mining to understand a time series. For example, we might have observed a sudden drop in prices on the stock market in a particular time period and would like to see if the companion text data such as news might help explain what happened. If the crashing time of the stocks corresponds to a time when a particular news topic suddenly appeared in the news stream, there might be a potential relationship between the topic and the stock crash. Similarly, one might also be interested in understanding what topics reported in the news stream were relevant for a presidential election, and thus interested in finding topics in news stream that are correlated with the fluctuation of the Presidential Prediction Market (which measures people's opinions toward each presidential candidate).

All these cases are special cases of a general problem of joint analysis of text and a time series to discover causal topics. Here we use the term *causal* in a non-rigorous way to refer to any topic that might be related to the time series and thus can be *potentially* causal. This analysis task is illustrated in Figure 19.14.

The input includes a time series plus text data that are produced in the same time period, also known as a companion text stream. The time series can be regarded as a

- Input:
  - Time series
  - Text data produced in a similar time period (text stream)
- Output:
  - Topics whose coverage in the text stream has strong correlations with the time series ("causal" topics)

Figure 19.14 The task of causal topic mining.

434 Chapter 19 *Joint Analysis of Text and Structured Data*

context for analyzing the text data. The output that we want to generate is the topics whose coverage in the text stream has strong correlations with the time series. That is, whenever the topic is mentioned frequently in the text stream, the time series variable tends to have higher (or lower) values.

We call these topics causal topics since they *can potentially* explain the cause of fluctuation of the time series and offer insights for humans to further analyze the topics for better understanding of the time series. They can also be useful features for predicting time series.

Intuitively, the output is similar to what we generate by using a topic model, but with an important difference. In regular topic modeling, our goal is to discover topics that best explain the content in the text data, but in our setup of discovering causal topics, the topics to be discovered should not only be semantically meaningful and coherent (as in the case of regular topic modeling), but also be correlated with the external time series.

To solve this problem, a natural idea is to apply a model such as CPLSA to our text stream so as to discover a number of topics along with their coverage over time. This would allow us to obtain a time series for each topic representing its coverage in the text such as the temporal trends shown in Figure 19.7. We can then choose the topics from this set that have the strongest correlation with the external time series.

However, this approach is not optimal because the content of the topics would have been discovered solely based on the text data (e.g., maximizing the likelihood function) without consideration of the time series at all. Indeed, the discovered topics would tend to be the major topics that explain the text data well (as they should be), but they are not necessarily correlated with time series. Even if we choose the best ones from them, the most correlated topics might still have a low correlation, and thus not be very useful from the perspective of discovering causal topics.

One way to improve this simple approach is to use time series context to not only select the topics with the highest correlations with the time series, but also influence the content of topics. One approach is called Iterative Causal Topic Modeling, shown in Figure 19.15.

The idea of this approach is to do an iterative adjustment of topics discovered by topic models using time series to induce a prior. Specifically, as shown in Figure 19.15, we first take the text stream as input and apply regular topic modeling to generate a number of topics (four shown here). Next, we use the external time series to assess which topic is more causally related (correlated) with the external time series by using a causality measure such as Granger Test. For example, in this

8. Future Improvements

The logic behind the ranking system seems to work very well based on validation results. What remains is mostly GUI improvements to show a nice bar chart on the ranked documents.

Also need to figure out a better mapping between the page index in text_data.dat vs the actual PDF file.

Another future improvement would also involve taking into consideration queries that user would enter. This would be considered as an additional query with a higher weight than what is derived from the subtitle list.

9. Timeline

Overall, this project much more time than the initial estimated 20 hours per student. This was due to facing a lot of hurdles and research required during the initial implementation.

Timeline
W8 - Proposal writeup, initial design
W9 - Research into algorithms and approach enhancements
W10 – Version 0.1 : basic features
W11 – Associated Technology Review
W12 – Version 0.2 : work on UI
W13 – Version 0.3 : start work on information retrieval algorithm, progress report
W14 – Function Test
W15,16 – Additional user test and submit

10. References

PyPDF2 : https://pypi.org/project/PyPDF2/
Pdfplumber : https://pypi.org/project/pdfplumber/0.1.2/
FITZ : https://pymupdf.readthedocs.io/en/latest/
Matplotlib : https://matplotlib.org/stable/
Metapy : https://github.com/meta-toolkit/metapy