

How to Find, Organize, and Manipulate It

Taming Text

Grant S. Ingersoll
Thomas S. Morton
Andrew L. Farris





**MEAP Edition
Manning Early Access Program
Taming Text version 8**

Copyright 2011 Manning Publications

For more information on this and other Manning titles go to
www.manning.com

Table of Contents

Chapter 1: Getting Started Taming Text

Chapter 2: Foundations of Taming Text

Chapter 3: Searching

Chapter 4: Fuzzy String Matching

Chapter 5: Identifying people, places, and things

Chapter 6: Clustering text

Chapter 7: Classification, categorization and tagging

Chapter 8: An example application: Question answering

Chapter 9: Untamed text: Exploring the next frontier

Appendix A: Example configuration files

Getting Started Taming Text



- Understand why processing text is important
- Learn what makes taming text hard
- Set the stage for leveraging open source libraries to tame text

If you are reading this book, chances are you are a programmer, or at least in the information technology field. You operate with relative ease when it comes to email, instant messaging, Google, YouTube, Facebook, Twitter, blogs and most of the other technologies that define our digital age. After you are done congratulating yourself on your technical prowess, take a moment to imagine your users. They feel imprisoned by the sheer volume of email they receive. They struggle to organize all the data that inundates their lives. And they probably don't know or even care about RSS or JSON, much less search engines, Bayesian classifiers or neural networks. They want to get answers to their questions without sifting through pages of results. They want email to be organized and prioritized, but spend very little time actually doing it themselves. Ultimately, your users want tools that enable them to focus on their lives and their work, not just their technology. In other words, they want to control, i.e. tame, the uncontrolled beast that is text. But what does it mean to tame text? We'll talk more about it later in this chapter, but for now taming text means three primary things:

- The ability to find relevant answers and supporting content given an information need
- The ability to organize (label, extract, summarize) and manipulate text with

little to no user intervention

- The ability to do both of these things with ever increasing amounts of input

This leads us to the primary goal of this book: to give you, the programmer, the tools and hands-on advice to build applications to help people better manage the tidal wave of communication that swamps their lives. The secondary goal of *Taming Text* is to show how to do this using existing, freely available, high quality, open source libraries and tools.

Before we get to those broader goals later in the book, let's take a step back and examine some of the factors involved in text processing, why it is hard, and also look at some use cases as motivation for the chapters to follow. Specifically, this chapter aims to provide some background on why processing text effectively is both important and challenging. We'll also lay some groundwork for the application we are building at the end of this book: a fact-based question answering system. With that, let's take a look at some of the motivation for taming text by scoping out the size and shape of the information world we live in.

1.1 Why Taming Text is Important

Just for fun, try to imagine going a whole day without reading a single word. That's right, one whole day without reading any news, signs, websites or even watching television. Think you could do it? Not likely, unless you sleep the whole day. Now spend a moment thinking about all the things that go into reading all that content: years of schooling and hands-on feedback from parents, teachers and peers; countless spelling tests, grammar lessons and book reports, not to mention the hundreds of thousands of dollars it takes to educate a person through college. Next, step back another level and think about how much content you do read in a day.

To get started, take a moment to consider the following questions:

- How many email messages did you get today (both work and personal, including spam)?
- How many of those did you read?
- How many did you respond to right away? Within the hour? Day? Week?
- How do you find old email?
- How many blogs did you read today?
- How many online news sites?
- Did you Instant Message (IM), Twitter or Facebook with friends or colleagues?
- How many searches did you do on Google, Yahoo or Bing?
- What documents on your computer did you read? What format were they in? (i.e. Word, PDF, text)
- How often do you search for something locally (either on your machine or your corporate

intranet?)

- How much content did you produce in the form of emails, reports, etc.?

Finally, the big question: how much time did you spend doing this?

If you are anything like the typical information worker, then you can most likely relate to IDC's findings from their 2009 study:

Email consumes an average of 13 hours per week per worker... But email is no longer the only communication vehicle. Social networks, instant messaging, Yammer, Twitter, Facebook, and LinkedIn have added new communication channels that can sap concentrated productivity time from the information worker's day. The time spent searching for information this year averaged 8.8 hours per week, for a cost of \$14,209 per worker per year. Analyzing information soaked up an additional 8.1 hours, costing the organization \$13,078 annually, making these two tasks relatively straightforward candidates for better automation. It makes sense that if workers are spending over a third of their time searching for information and another quarter analyzing it, this time must be as productive as possible.

Furthermore, this survey doesn't even account for how much time these same employees spend creating content during their personal time. In fact, JupiterResearch estimates that Internet users average 14 hours a week online (jupiterOnline-2006) and compares this to other leisure activities. Internet Users Time Spent Online details how the average Internet users spends their time. No doubt this number is low compared to 2011 terms.

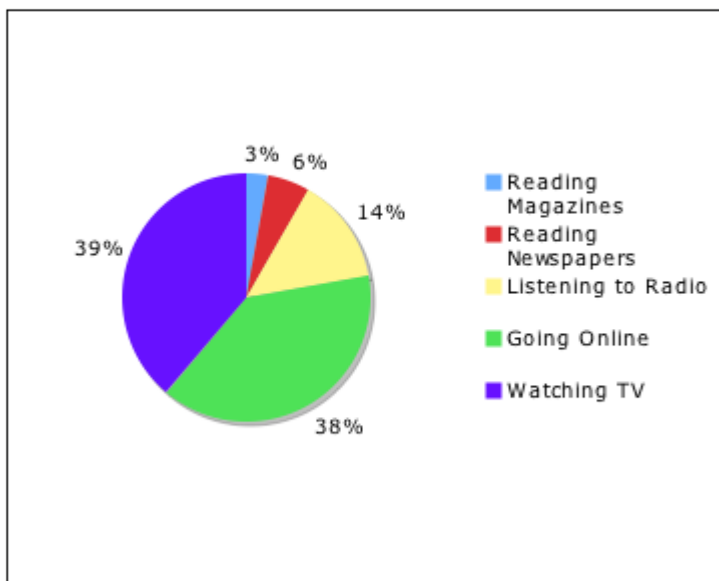


Figure 1.1 Internet Users Time Spent Online

Thus, whether it is reading email, searching Google, reading a book or logging into Facebook, the written word is everywhere in our lives.

We've seen the individual part of the content picture, but what about the big picture? According to International Data Corporation (IDC 2011), the world generated *1.8 zettabytes* of digital information in 2011 and "by 2020 the world will generate 50 times [that amount]". Even if most of the size of this data is due to signal data, images, audio and video, the current best approach to making all this data findable is to write analysis reports, add keyword tags and text descriptions or to transcribe the audio using speech recognition or a manual closed-captioning approach so that it can be treated as text. In other words, no matter how much structure we add, it still comes back to text for us to share and comprehend our content. As you can see, just the sheer volume of content can be daunting, never mind that text processing is also a hard problem on a small scale, as you'll see in a later section. In the meantime, it is worthwhile to think about what would the ideal applications or tools do to help stem the tide of text that is engulfing us. For many of us, the the answer lies in the ability to quickly and efficiently hone in on the answer to our questions, not just a list of possible answers that we need to then sift through. Moreover, we wouldn't need to jump through hoops to ask our questions, we'd just be able to use our own words or voice to express them with no needs for things like quotations, AND/OR operators or other things that make it easier on the machine but harder on the person. While we all know we don't live in an ideal world, one of the promising approaches for taming text, popularized by IBM's Jeopardy! playing Watson program and Apple's Siri application, is a Question Answering system that can process natural languages such as English and return back actual answers, not just page after page of possible answers. In *Taming Text*, we aim to lay some of the groundwork for building such a system, so let's take a look at what such a system might look like, before proceeding to delve a bit deeper into why building such a system as well as other language based systems are so hard. We'll then finish off with a look at how the chapters to follow in this book will lay the foundation for the building of a very simple, fact-based QA system.

1.1.1 Towards a Fact-Based Question Answering System

For the purposes of this book, a QA system should be capable of ingesting a collection of documents suspected to have answers to questions that users might ask. For instance, Wikipedia or a collection of research papers might be used as a source for finding answers. In other words, the QA system we are proposing to build is based on identifying and analyzing text that has a chance of providing the answer based on patterns it has seen in the past. It will not be capable of inferring an answer from a variety of sources. For instance, if the system is asked "Who is Bob's Uncle?" and there is a document in the collection with the sentences "Bob's father is Ola. Ola's brother is Paul." the system would not be able to infer that Bob's Uncle is Paul. However if there is a sentence that directly states "Bob's uncle is Paul." we would expect the system to be able to answer the question. This is not to say that the former example can't be attempted, it is just beyond the scope of this book.

A simple workflow for building the QA system described earlier is outlined in the figure titled A Simple QA Workflow.

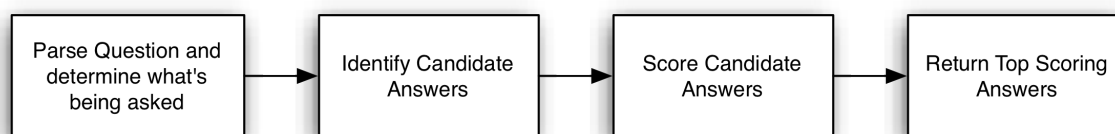


Figure 1.2 A Simple QA Workflow

Naturally, such a simple workflow hides a lot of details and it also doesn't cover the ingestion of the documents, but it does allow us to highlight some of the key components needed to process user's questions. First off, the ability to parse a user's question and determine what is being asked typically requires basic functionality like identifying words, as well as the ability to understand what kind of answer is appropriate for a question. For instance, the answer to "Who is Bob's Uncle?" should likely be a person, whereas the answer to "where is Buffalo?" probably requires a place to be returned. Second, the need to identify candidate answers typically involves the ability to quickly look up phrases, sentences or passages that contain potential answers without having to force the system to parse large quantities of text. Scoring implies many of the basic things again, such as parsing words, as well as a deeper understanding of whether a candidate actually contains the necessary components to answer a question, such as it mentions a

person. As easy as some of these basic things sound, they are not to be taken for granted and why some of the other parts can prove to be quite difficult. The next section will touch on some of these difficulties and should serve to motivate many of the approaches we take in the remainder of the book.

1.1.2 Understanding Text is Hard

Suppose Robin and Joe are talking, and Joe states "The bank on the left is solid, but the one on the right is crumbling." What are Robin and Joe talking about? Are they on Wall Street looking at the offices of two financial institutions or are they floating down the Mississippi River looking for a place to land their canoe? If you assume the former, the words "solid" and "crumbling" probably refer to the state of the banks finances, while the latter case is an assessment of the quality of the ground on the side of a river. Now, what if we had replaced the characters names with the names Huck and Tom from "The Adventures of Tom Sawyer"? You would likely feel pretty confident in stating it's a river bank and not a financial institution, right? As you can see, context is also important. In fact, it is often the case that only with more information from the surrounding context combined with our own experiences can we truly know what some piece of content is about. The ambiguity in Joe's statement only touches on the surface of the complexity involved in understanding text.

Given well-written, coherent sentences and paragraphs, knowledgeable people seamlessly look up the meanings of words and incorporate their experiences and knowledge of their surroundings to arrive at an understanding of content and conversations. Literate adults can (more or less) effortlessly dissect sentences, identify relationships and infer meaning nearly instantaneously. And, just as in the Robin and Joe example, people almost always are aware when something is significantly out of place or lacking from a sentence, paragraph or document as a whole. Human beings also feed off others in conversation, instantly adapting tone and emotions to convey thoughts on subjects ranging from the weather to politics to the role of the designated hitter. While we often take these skills for granted, we should remember that they have been fine-tuned through many years of conversation, education and feedback from others, not to mention all the knowledge passed down from our ancestors. At the same time, computers and the fields of Information Retrieval (IR) and Natural Language Processing (NLP) are still relatively young. Computers need to be capable of processing language on many different levels in order to come close to "understanding" content like people do. (For an in-depth discussion of the many factors that go into NLP, see Liddy

2001.) While full understanding is a tall order for a computer, even doing basic tasks can be overwhelming under the sheer volume of text available and the variety with which it occurs.

There is a reason the saying is "the numbers don't lie" and not "the text doesn't lie"; text comes in all shapes and meanings and trips up even the smartest of people on a regular basis. Writing applications to process text can mean facing a number of technical and non-technical challenges. Text Challenges outlines some of the challenges text applications face, each row increasing in difficulty from the previous.

Table 1.1 Text Challenges

Level	Challenges
Character	<ul style="list-style-type: none"> • Character encodings, such as ASCII, Shift-JIS, Big 5, UNICODE, UTF-8, UTF-16 • Case (upper and lower), punctuation, accents, numbers all require different treatment in different applications
Words and Morphemes ¹	<ul style="list-style-type: none"> • Word Segmentation -- Dividing text into words. Fairly easy for English and other languages that use whitespace. Much harder for languages like Chinese and Japanese. • Assigning Part of Speech • Identifying synonyms -- Synonyms are useful for searching. • Stemming -- The process of shortening a word to its base or root form. For example a simple stemming of "words" is "word". • Abbreviations, acronyms, and spelling also play important roles in understanding words.
Multi-word and Sentence	<ul style="list-style-type: none"> • Phrase detection -- "quick red fox", "hockey legend Bobby Orr" and "big, brown shoe" are all example phrases. • Parsing -- Breaking sentences down into Subject-Verb and other relationships often yields useful information about words and their relationships to each other. • Sentence boundary detection is a well understood problem in English, but is still not perfect. • Coreference resolution -- "Jason likes dogs, but he would never buy one." In this example, "he" is a coreference to Jason. The need for coreference resolution can also span sentences. • Words often have multiple meanings, using the context of a sentence or more may help choose the correct word. This process is called Word Sense Disambiguation and is difficult to do well. • Combining the definitions of words and their relationships to each other to determine the meaning of a sentence
Multi-sentence and Paragraph	<p>At this level, processing becomes more difficult in an effort to find deeper understanding of an author's intent. Algorithms for summarization often require being able to identify which sentences are more important than others.</p>

Document	Similar to the paragraph level, understanding the meaning of a document often requires knowledge that goes beyond what is contained in the actual document. Authors often expect readers to have a certain background or certain reading skills. For example, most of this book won't make much sense if you have never used a computer and done some programming, while most newspapers assume at least a sixth grade reading level.
Multi-Document and Corpus	At this level, people want to quickly find what items of interest as well as group related documents and read summaries of those documents. Applications that can aggregate and organize facts and opinions and find relationships are particularly useful.

Beyond these challenges, human factors also play a role in working with text. Different cultures, different languages and different interpretations of the same writing can leave even the best engineer wondering what to implement. Merely looking at some sample files and trying to extrapolate an approach for a whole collection of documents is often problematic. On the other side of the coin, manually analyzing and annotating large sets of documents can be expensive and time consuming. Rest assured, however, that help is available and text can be tamed.

1.1.3 Text, Tamed

Now that you've seen some of the challenges we are about to face, take heart knowing that many tools exist both commercially and in the open source community (see <http://www.opensource.org>) to tackle these topics and many more. One of the great things about the topic we are embarking on is the ever-changing and ever-improving nature of it. Problems that were intractable ten years ago due to resource limits are now yielding positive results thanks to better algorithms, faster CPUs, cheaper memory, cheaper disk space and tools for easily harnessing many computers into a single, virtual CPU. Now more than ever, quality open source tools exist that can form the foundation for new ideas and new applications.

This book is written to bring real world experience to these open source tools and introduce you to the field of Natural Language Processing (NLP) and Information Retrieval (IR). We can't possibly cover all aspects of NLP and IR nor are we going to discuss bleeding edge research, at least not until the end of the book; instead we will focus on areas that are likely to have the biggest impact in taming your text.

By focusing our chapters on topics like search, entity identification (finding people, places and things), grouping and labeling, clustering and summarization, we can build practical applications that help users find and understand the important parts of their text quickly and easily.

While we hate to be a buzzkill on all the excitement of taming text, it is important to note that there are no perfect approaches in working with text. There are many times when two people reviewing the same output will not agree on the correctness of the results, nor will it be obvious what to fix to satisfy them. Furthermore, fixing one problem may expose other problems. Testing and analysis are as important as ever to achieving quality results. Ultimately, the best systems take a human-in-the-loop approach and learn from user feedback where possible, just as smart people learn from their mistakes and from their peers. The user feedback need not be explicit, either. Capturing clicks, analyzing logs and other user behaviors can provide valuable feedback on how your users are utilizing your application. With that in mind, here are some general tips for improving your application and keeping your sanity:

- Get to know your users. Do they care about certain structures like tables and lists or is it enough to just collect all the words in a document? Are they willing to give you more information in return for better results or is simplicity the rule? Are they willing to wait longer for better results or do they need a best guess immediately?
- Get to know your content. What file formats (HTML, Microsoft Word, PDF, text) are used? What structures and features are important? Does the text contain a lot of jargon or abbreviations or different ways of saying the same thing? Is the content focused on a single area of interest or does it cover a number of topics?
- Test, test, and more test. Take the time (but not too much time) to measure the quality of your results and the cost of obtaining them. Become practiced in the art of arbitration. Every non-trivial, text-based application will need to make tradeoffs in regards to quality and scalability. By combining your knowledge of your users and your content, you can often find the "sweet spot" of quality and performance that satisfy most people most of the time.
- Sometimes, a best guess is as good as it gets. Look for ways to provide confidence levels to your users so they can make an informed decision about your response.
- All else being equal, favor the simpler approach. Moreover, you will be amazed at how good simple solutions can be at getting decent results.

Also, while working in non-native languages is an interesting problem on its own, we are going to stick to English for this book. Rest assured, though, that many of the approaches can be applied to other languages given the right resources.

It should also be pointed out that the kinds of problems one wishes to solve

range in difficulty from relatively straightforward to you might as well flip a coin. For instance, in English and other European languages, tokenization and part of speech tagging algorithms perform quite well, while tools like machine translation of foreign languages, sentiment analysis and reasoning from text are much more difficult and often do not perform well in unconstrained environments.

Finally, text processing is much like riding a roller coaster. There will be highs when your application can do no wrong and lows when your application can do no right. The fact is, none of the approaches discussed in this book or in the broader field of NLP are the final solution to the problem. Therein lies the ultimate opportunity for you to dig in and add your signature on the problem. So, let's get started and lay the foundation for the ideas to come in later chapters by setting the context that takes us beyond search into the wonderful world of natural language processing.

1.2 Text and the Intelligent App: Search and Beyond

For many years now, search has been king. Without the likes of Google and Yahoo!, there is no doubt that the Internet would not be anywhere near what it is today. Yet, with the rise of good open source search tools like Apache Solr and Apache Lucene, along with a myriad of crawlers and distributed processing techniques, search is a commodity, at least on the smaller scale of personal and corporate search where huge data centers are not required. At the same time, people's expectations of search engines are increasing. We want better results in less time all while entering only one or two keywords. We also want our own content easily searched and organized.

Furthermore, corporations are under huge pressures to constantly add value. Every time some big player like Google or Amazon makes a move to better access information, the bar is raised for the rest of us. Five, ten or fifteen years ago, it was enough to add search capabilities to be able to find data, now search is a prerequisite and the game-changing players use complex algorithms utilizing machine learning and deep statistical analysis to work with volumes of data that would take people years to understand. This is, in fact, the evolution of the intelligent application. More and more companies are adopting machine learning and deep text analysis in well-defined areas to bring more intelligence to their applications. Moreover, this adoption of machine learning and NLP techniques is grounded in the reality of practical applications dealing with large volumes of data, and not the grandiose, albeit worthwhile, notion of machines "understanding" people or somehow passing the Turing Test (see

http://en.wikipedia.org/wiki/Turing_Test.) Indeed, these companies are focused on finding and extracting important text features; aggregating information like user clicks, ratings and reviews; grouping and summarizing similar content; and, finally, displaying all of these features in ways that allow end users to better find and use the content, which should ultimately lead to more purchases or traffic or whatever it is they sell. After all, you can't buy something if you can't find it, right?

So, how do we get started doing all of these great things? We start by establishing the baseline with search (covered in Searching) and then we examine ways of automatically organizing content using concepts that we employ in our daily lives. Instead of doing it manually, however, we let the machine do it for us (with a little help when needed.) With that in mind, the next few sections break down the ideas of search and organizing content into four distinct areas, which will be explored more completely in the ensuing chapters.

SEARCHING AND MATCHING

Search provides the starting point for most of your text taming activities, including our proposed QA system, where we will rely on it both for indexing the input data as well as for identifying candidate passages that match a user's question. Even when you need to apply techniques that go beyond search, you will likely use search to find the sub-set of text or documents on which to apply more advanced techniques. In Chapter 3, Searching, we will explore how to make documents available for searching, indexing, and how to retrieve documents based on a query. We will also explore how documents are ranked by a search engine and use this information to improve the returned results. Finally, we will examine faceted search which allows searches to be refined by limiting results to a pre-defined category. The coverage of these topics will be grounded in examples using Apache Solr and Apache Lucene.

Once familiar with the techniques of search, you will quickly realize that search is only as good as the content backing that search. If the word and phrases that your users are looking for are not in your index, then you won't be able to return a relevant result. In Chapter 4, Fuzzy String Matching, we will look at techniques for enabling query recommendations based on the content that is available via query spell checking as well as how these same techniques can be applied to database or record linking tasks that go beyond simple database joins. These techniques are often used not only as part of search, but also for more complex things like

identifying whether two user profiles are indeed the same person, such as what might happen when two companies merge and their customer lists must be combined.

EXTRACTING INFORMATION

While search will help you find documents that contain the information you need, often one needs to be able to identify smaller units of information. For instance, the ability to identify proper names in a large collection of text can be immensely helpful in tracking down criminal activity or finding relationships between people that might not otherwise meet. To do this we explore techniques for identifying and classifying small selections of text, typically just a few words in length. In Chapter 2, Foundations of Taming Text, we introduce techniques for identify words which form a linguistic unit such as noun-phrases which can be used to identify word in a document or query which can be grouped together. In Chapter 5, Identifying People, Places and Things, we look at how to identify proper names and numeric phrases and put them into semantic categories such as person, location, and date irrespective of their linguistic usage. This ability will be fundamental to our ability to build out a QA system in Chapter 8. For both of these tasks we'll use the capabilities of OpenNLP and explore how to use its existing models as well as build new models that better fit the data. Unlike the problem of searching and matching, these models will be built from examining manually annotated content and then using statistical machine learning approaches to produce a model.

GROUPING INFORMATION

The flip side to extracting information from text is adding supplemental information to your text by grouping it together or adding labels. For example, think about how much easier it would be to process your email if it were automatically tagged and prioritized and that you could also find all emails that are similar to one another? This way, you could focus in on just those emails that require your immediate attention as well as find supporting content for emails you are sending. One common approach to this is to group your text into categories and, it turns out, that the techniques used for extracting information can also be applied to grouping text or documents into categories. These groups can often then be used as facets in your search index, supplemental keywords, or simply as an alternate way for users to navigate information. Even in cases where your users are providing the categories via tagging, these techniques can recommend tags that have been used in the past. Chapter 6, , will show how to build models to classify documents and how to apply these models to new documents to improve user experience with text.

Once you have tamed your text and are able to find what you're looking for, and extracted the information needed, you may find you have too much of a good thing. In Chapter 7, Clustering Text, we will look at how to group similar information. These techniques can be used to identify redundant information and, if necessary, suppress it. It can also be used to group similar documents so that a user can peruse entire topics at a time and access the relevancy of multiple documents at once without having to read each document.

AN INTELLIGENT APPLICATION

In our penultimate chapter, *An Example Application: Question Answering*, we will bring, as discussed earlier, a number of the approaches described in the early chapters together to build an intelligent application. Specifically, we will build a fact-based question answering system designed to find answers to trivia-like questions in text. For instance, given the right content, we should be able to answer questions like "Who is the President of the United States?". This system uses the techniques of Chapter 3, *Searching*, to identify text which might contain the answer to our question. The approaches presented in Chapter 5, *Identifying People, Places and Things*, will be used to find these pieces of text that are often the answers to fact-based questions. The material in Chapter 2, *Foundations of Taming Text*, and Chapter 6, *Ranking*, will be used to analyze the question being asked, and determine what type of information the question is looking for. Finally, we will apply the techniques for document ranking described in Chapter 3, *Searching*, to rank our answers.

1.3 Summary

Taming text is a large and sometimes overwhelming task, further complicated by different languages, different dialects and different interpretations. Text can appear as elegant prose written by great authors or the ugliest of essays written without style or substance. Whatever its form, text is everywhere and it must be dealt with by people and programs. Luckily, many tools exist both commercially and in open source to help try to make sense of it all. It won't be perfect, but it is getting better all the time. So far, we've taken a look at some of the reasons why text is so important as well as hard to process. We've also looked at what role text plays in the intelligent web, introduced the topics we are going to cover and gave a brief overview of some of the things needed to build a simple question answering system. In the next chapter, we will kick things off by laying down the foundations of text analysis along with some basics on extracting raw text from the many file formats found in the wild today.

Bibliography