Data-Intensive Distributed Computing
CS 431/631 451/651 (Winter 2021)

Part 4: Analyzing Text (1/2)

Ali Abedi

These slides are available at https://www.student.cs.uwaterloo.ca/~cs451

This work is licensed under a Creative Commons Attribution-Noncommercial-Share Alike 3.0 United States
See http://creativecommons.org/licenses/by-nc-sa/3.0/us/ for details
Structure of the Course

“Core” framework features and algorithm design
Structure of the Course

“Core” framework features and algorithm design

- Analyzing Text
- Analyzing Graphs
- Analyzing Relational Data
- Data Mining
Natural Language Processing

PayPal 08:46 PM
Hi! I'm PayPal's virtual agent. To get started, simply ask me a question.
I am still learning, so if I can't help you I'll direct you to additional resources.

BP Brady [redacted] 08:47 PM
I got scammed

PayPal 08:47 PM
Great!
Pairs. Stripes.
Seems pretty trivial...

More than a “toy problem”?  
Answer: language models
Language Models

\[ P(w_1, w_2, \ldots, w_T) \quad \text{Assigning a probability to a sentence} \]

Why?

- **Machine translation**
  - \[ P(\text{High winds tonight}) > P(\text{Large winds tonight}) \]
- **Spell Correction**
  - \[ P(\text{Waterloo is a great city}) > P(\text{Waterloo is a grate city}) \]
- **Speech recognition**
  - \[ P(\text{I saw a van}) > P(\text{eyes awe of an}) \]

Sentence with T words - assign a probability to it

Slide: from Dan Jurafsky
Language Models

\[ P(w_1, w_2, \ldots, w_T) \]
\[ = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \ldots P(w_T|w_1, \ldots, w_{T-1}) \]  
[chain rule]

\[ P(“Waterloo is a great city”) = \]
\[ P(Waterloo) \times P(is \mid Waterloo) \times P(a \mid Waterloo is) \]
\[ \times P(great \mid Waterloo is a) \]
\[ \times P(city \mid Waterloo is a great) \]

Is this tractable?

Sentence with T words - assign a probability to it

\[ P(A,B) = P(B) \times P(A|B) \]
Approximating Probabilities: \( N \)-Grams

Basic idea: limit history to fixed number of \((N - 1)\) words

(Markov Assumption)

\[
P(w_k|w_1, \ldots, w_{k-1}) \approx P(w_k|w_{k-N+1}, \ldots, w_{k-1})
\]

\(N=1\): Unigram Language Model

\[
P(w_k|w_1, \ldots, w_{k-1}) \approx P(w_k)
\]

\[
\Rightarrow P(w_1, w_2, \ldots, w_T) \approx P(w_1)P(w_2)\ldots P(w_T)
\]
Approximating Probabilities: \(N\)-Grams

Basic idea: limit history to fixed number of \((N-1)\) words
(Markov Assumption)

\[
P(w_k|w_1, \ldots, w_{k-1}) \approx P(w_k|w_{k-N+1}, \ldots, w_{k-1})
\]

\(N=2\): Bigram Language Model

\[
P(w_k|w_1, \ldots, w_{k-1}) \approx P(w_k|w_{k-1})
\]

\[
\Rightarrow P(w_1, w_2, \ldots, w_T) \approx P(w_1|s) P(w_2|w_1) \ldots P(w_T|w_{T-1})
\]

Since we also want to include the first word in the bigram model, we need a dummy beginning of sentence marker \(<s>\). We usually also have an end of sentence marker but for the sake of brevity, I don’t show that here.
Approximating Probabilities: $N$-Grams

Basic idea: limit history to fixed number of $(N - 1)$ words
(Markov Assumption)

$$P(w_k | w_1, \ldots, w_{k-1}) \approx P(w_k | w_{k-N+1}, \ldots, w_{k-1})$$

$N=3$: Trigram Language Model

$$P(w_k | w_1, \ldots, w_{k-1}) \approx P(w_k | w_{k-2}, w_{k-1})$$

$$\Rightarrow P(w_1, w_2, \ldots, w_T) \approx P(w_1 | <S>|S>) \cdots P(w_T | w_{T-2}w_{T-1})$$
Building N-Gram Language Models

Compute maximum likelihood estimates (MLE) for individual n-gram probabilities

Unigram

\[ P(w_i) = \frac{C(w_i)}{N} \]

Bigram

\[ P(w_i, w_j) = \frac{C(w_i, w_j)}{C(w_i)} \]

\[ P(w_j|w_i) = \frac{P(w_i, w_j)}{P(w_i)} = \frac{C(w_i, w_j)}{\sum_w C(w_i, w)} \times \frac{C(w_i)}{C(w_i)} \]

Generalizes to higher-order n-grams
State of the art models use ~5-grams

We already know how to do this in MapReduce!
SOMETIMES I’LL START A SENTENCE AND I DON’T EVEN KNOW WHERE IT’S GOING.
Estimating Probability Distribution
Sparsity problem
Example: Bigram Language Model

<s> I am Sam </s>
<s> Sam I am </s>
<s> I do not like green eggs and ham </s>

Training Corpus

\[
P(I \mid <s>) = \frac{2}{3} = 0.67 \quad P(\text{Sam} \mid <s>) = \frac{1}{3} = 0.33
\]
\[
P(\text{am} \mid I) = \frac{2}{3} = 0.67 \quad P(\text{do} \mid I) = \frac{1}{3} = 0.33
\]
\[
P(\langle s \rangle \mid \text{Sam}) = \frac{1}{2} = 0.50 \quad P(\text{Sam} \mid \text{am}) = \frac{1}{2} = 0.50
\]

... 

Bigram Probability Estimates

Note: We don’t ever cross sentence boundaries
Data Sparsity

\[
P(I \mid <s>) = \frac{2}{3} = 0.67 \quad P(\text{Sam} \mid <s>) = \frac{1}{3} = 0.33 \\
P(\text{am} \mid I) = \frac{2}{3} = 0.67 \quad P(\text{do} \mid I) = \frac{1}{3} = 0.33 \\
P(<s> \mid \text{Sam}) = \frac{1}{2} = 0.50 \quad P(\text{Sam} \mid \text{am}) = \frac{1}{2} = 0.50 \\
\]

\[\ldots\]

Bigram Probability Estimates

\[P(I \text{ like ham})\]

\[= P(I \mid <s>) \cdot P(\text{like} \mid I) \cdot P(\text{ham} \mid \text{like}) \cdot P(<s> \mid \text{ham})\]

\[= 0\]

Why is this bad?

Issue: Sparsity!

Why is the 0 bad?
WORK HOURS ON A LANGUAGE MODEL, TIME TO TEST IT
AAAND IT SUCKS
Solution: Smoothing

Zeros are bad for any statistical estimator
Need better estimators because MLEs give us a lot of zeros
A distribution without zeros is “smoother”

The Robin Hood Philosophy: Take from the rich (seen n-grams) and give to the poor (unseen n-grams)
Need better estimators because MLEs give us a lot of zeros
A distribution without zeros is “smoother”

Lots of techniques:
Laplace, Good-Turing, Katz backoff, Jelinek-Mercer
Kneser-Ney represents best practice
Laplace Smoothing

Simplest and oldest smoothing technique
Just add 1 to all \( n \)-gram counts including the unseen ones
So, what do the revised estimates look like?
Laplace Smoothing

Unigrams

\[ P_{MLE}(w_i) = \frac{C(w_i)}{N} \quad \longrightarrow \quad P_{LAP}(w_i) = \frac{C(w_i) + 1}{N + V} \]

Bigrams

\[ P_{MLE}(w_i, w_j) = \frac{C(w_i, w_j)}{N} \quad \longrightarrow \quad P_{LAP}(w_i, w_j) = \frac{C(w_i, w_j) + 1}{N + V^2} \]

What if we don’t know \( V \)?

You have to make sure that the joint is well-formed and understand how the conditional probability formula is derived.
The Central Problem in Search

Why is IR hard? Because language is hard!

Do these represent the same concepts?

“tragic love story”

“fateful star-crossed romance”
Abstract IR Architecture

- Query
  - Representation Function
    - Query Representation
  - Comparison Function
  - Hits

- Documents
  - Representation Function
    - Document Representation
  - Index
  - Document acquisition (e.g., web crawling)
How do we represent text?
Remember: computers don’t “understand” anything!

“Bag of words”
Treat all the words in a document as index terms
Assign a “weight” to each term based on “importance”
(or, in simplest case, presence/absence of word)
Disregard order, structure, meaning, etc. of the words
Simple, yet effective!

Assumptions
Term occurrence is independent
Document relevance is independent
“Words” are well-defined
What’s a word?

天主教教宗若望保祿二世因感冒再度住院。

这是他今年第二度因同样的病因住院。

马克·里杰夫 - 作为以色列外交部的发言人 - 表示，沙龙将按照原定计划，在首次访问突尼斯后，突尼斯在突尼斯作为临时总部的长期存在之后，于1982年从黎巴嫩撤军。

在莫斯科的梅什贾诺法院审讯时，尤科斯前总经理表示，他没有犯任何违法行为，这是俄罗斯总检察官指控他的。

印度政府在2005-2006年经济普查中预计，2005年-2006年的七年内，印度将实现7%的经济增长，并且在改善税收方面将有所改善。

美日联合声明中国台湾问题…阿明提戈前副长官提言

乔浩明记者于25日向市长朴三民市长提出了有关“行政中心综合都市”建设的建议，市长表示，即使有军事支持，也要采取行动来阻止这项计划。
McDonald’s slims down spuds

Fast-food chain to reduce certain types of fat in its french fries with new cooking oil.

NEW YORK (CNN/Money) - McDonald’s Corp. is cutting the amount of “bad” fat in its french fries nearly in half, the fast-food chain said Tuesday as it moves to make all its fried menu items healthier.

But does that mean the popular shoestring fries won’t taste the same? The company says no. “It’s a win-win for our customers because they are getting the same great french-fry taste along with an even healthier nutrition profile,” said Mike Roberts, president of McDonald’s USA.

But others are not so sure. McDonald’s will not specifically discuss the kind of oil it plans to use, but at least one nutrition expert says playing with the formula could mean a different taste.

Shares of Oak Brook, Ill.-based McDonald’s (MCD: down $0.54 to $23.22, Research, Estimates) were lower Tuesday afternoon. It was unclear Tuesday whether competitors Burger King and Wendy’s International (WEN: down $0.80 to $34.91, Research, Estimates) would follow suit. Neither company could immediately be reached for comment.

…

“Bag of Words”

14 × McDonalds
12 × fat
11 × fries
8 × new
7 × french
6 × company, said, nutrition
5 × food, oil, percent, reduce, taste, Tuesday

…
Counting Words...

Documents

Bag of Words

case folding, tokenization, stopword removal, stemming

Inverted Index

syntax, semantics, word knowledge, etc.
Count.

Source: http://www.flickr.com/photos/guvnah/7861419602/
<table>
<thead>
<tr>
<th></th>
<th>Doc 1 one fish, two fish</th>
<th>Doc 2 red fish, blue fish</th>
<th>Doc 3 cat in the hat</th>
<th>Doc 4 green eggs and ham</th>
</tr>
</thead>
<tbody>
<tr>
<td>blue</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cat</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>egg</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fish</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>green</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ham</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hat</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>one</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>red</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>two</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

What goes in each cell?
- boolean
- count
- positions
Abstract IR Architecture

Query

Representation Function
Query Representation

Comparison Function

Documents

Representation Function
Document Representation

Index

Indexing

Retrieval

online offline

Hits
Indexing: building this structure
Retrieval: manipulating this structure
<table>
<thead>
<tr>
<th>Doc 1</th>
<th>Doc 2</th>
<th>Doc 3</th>
<th>Doc 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>one</td>
<td>red</td>
<td>cat</td>
<td>blue, red, one</td>
</tr>
<tr>
<td>fish</td>
<td>fish</td>
<td>fish</td>
<td>blue, fish, red</td>
</tr>
<tr>
<td>blue</td>
<td>green</td>
<td>blue</td>
<td>blue, green</td>
</tr>
<tr>
<td>cat</td>
<td>egg</td>
<td>cat</td>
<td>cat, blue, red</td>
</tr>
<tr>
<td>egg</td>
<td>fish</td>
<td>egg</td>
<td>red, cat, blue</td>
</tr>
<tr>
<td>green</td>
<td>fish</td>
<td>green</td>
<td>green, blue</td>
</tr>
<tr>
<td>fish</td>
<td>egg</td>
<td>fish</td>
<td>fish, green</td>
</tr>
<tr>
<td>blue</td>
<td>cat</td>
<td>blue</td>
<td>blue, cat</td>
</tr>
<tr>
<td>blue</td>
<td>cat</td>
<td>blue</td>
<td>blue, cat</td>
</tr>
<tr>
<td>red</td>
<td>two</td>
<td>red</td>
<td>red, two</td>
</tr>
<tr>
<td>two</td>
<td>red</td>
<td>two</td>
<td>two, red</td>
</tr>
</tbody>
</table>

**postings lists**
Indexing: Performance Analysis

Fundamentally, a large sorting problem
Terms usually fit in memory
Postings usually don’t

How is it done on a single machine?
How can it be done with MapReduce?

First, let’s characterize the problem size:
Size of vocabulary
Size of postings
Vocabulary Size: Heaps’ Law

\[ M = kT^b \]

- \( M \) is vocabulary size
- \( T \) is collection size (number of documents)
- \( k \) and \( b \) are constants

Typically, \( k \) is between 30 and 100, \( b \) is between 0.4 and 0.6

Heaps’ Law: linear in log-log space

Surprise: Vocabulary size grows unbounded!
Heaps’ Law for RCV1

First 1,000,020 terms:
- Predicted = 38,323
- Actual = 38,365


Manning, Raghavan, Schütze, Introduction to Information Retrieval (2008)
Postings Size: Zipf’s Law

\[ f(k; s, N) = \frac{1/k^s}{\sum_{n=1}^{N} (1/n^s)} \]

- \( N \): number of elements
- \( k \): rank
- \( s \): characteristic exponent

Zipf’s Law: (also) linear in log-log space

Specific case of Power Law distributions

In other words:
- A few elements occur very frequently
- Many elements occur very infrequently
Fit isn’t that good... but good enough!

Zipf’s Law for RCV1


Manning, Raghavan, Schütze, Introduction to Information Retrieval (2008)
Zipf’s Law for Wikipedia

Rank versus frequency for the first 10m words in 30 Wikipedias (dumps from October 2015)
Power Laws are everywhere!

MapReduce: Index Construction

Map over all documents
Emit term as key, \((docid, tf)\) as value
Emit other information as necessary (e.g., term position)

Sort/shuffle: group postings by term

Reduce
Gather and sort the postings (typically by \(docid\))
Write postings to disk

MapReduce does all the heavy lifting!
Inverted Indexing with MapReduce

**Map**

<table>
<thead>
<tr>
<th></th>
<th>Doc 1 one fish, two fish</th>
<th>Doc 2 red fish, blue fish</th>
<th>Doc 3 cat in the hat</th>
</tr>
</thead>
<tbody>
<tr>
<td>one</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>two</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>fish</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

**Reduce**

**Shuffle and Sort: aggregate values by keys**

<table>
<thead>
<tr>
<th></th>
<th>cat</th>
<th>fish</th>
<th>one</th>
<th>red</th>
<th>blue</th>
<th>hat</th>
<th>two</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

*Inverted Indexing with MapReduce*
Inverted Indexing: Pseudo-Code

class Mapper {
  def map(docid: Long, doc: String) = {
    val counts = new Map()
    for (term <- tokenize(doc)) {
      counts(term) += 1
    }
    for ((term, tf) <- counts) {
      emit(term, (docid, tf))
    }
  }
}

class Reducer {
  def reduce(term: String, postings: Iterable[(docid, tf)]) = {
    val p = new List()
    for ((docid, tf) <- postings) {
      p.append((docid, tf))
    }
    p.sort()
    emit(term, p)
  }
}

What’s the problem?

Stay tuned...