Admin

Assignment 4a
- Solutions posted
- If you’re still unsure about questions 3 and 4, come talk to me.

Assignment 4b

Grading

Quiz #2 next Thursday covering material through 3/6

Admin

Office hours between:
- M: 3 - 3:50pm
- T: 11am - 12
- Th: 11am - 12:30
- F: 10 - 11am

Course feedback

Thanks!
Course feedback

Assignments can be tough but overall are doable. Toughness is due to debugging at times.

The pace is too slow. For example, we took too much time to cover n-grams.

If the homeworks had more regular due-date (eg. every Friday), it’d be easier to plan an NLP schedule.

Course feedback

It’s difficult to tell exactly what parts of the material we cover we’ll be expected to remember.

I know it would be a lot of work, but if possible I’d appreciate Python starter code.

Grade homeworks more quickly

Due to there being only one instructor, sometimes it is hard to get advice/answers to questions.

Course feedback

This is the only stem class I’ve taken where I haven’t collaborated in person with any other students. I’m not suggesting mandatory group assignments, but I more so miss the environment mentor sessions provide. Has not been an issue though, just a consideration.

Organize mentor-less mentor sessions (some teachers in the math department do this when there are no TAs available). Students work together and answer each other’s questions.
A common question in NLP is how similar are texts?

\[ \text{sim}(\text{document}_1, \text{document}_2) = ? \]

How could these be useful? Applications?

---

Text classification

- sports
- politics
- business

These “documents” could be actual documents, for example using k-means or pseudo-documents, like a class centroid/average.

---

Text clustering

---

Information retrieval (search)

- Query
- Data set (e.g., web)
Text similarity: applications

Automatic evaluation

Word similarity

sim( banana, apple ) = ?

Word-sense disambiguation

I went to the bank to get some money.

Financial bank  River bank

Text similarity: applications

Automatic grader

Question: what is a variable?
Answer: a location in memory that can store a value

How good are:
- a variable is a location in memory where a value can be stored
- a named object that can hold a numerical or letter value
- it is a location in the computer’s memory where it can be stored for use by a program
- a variable is the memory address for a specific type of stored data or from a mathematical perspective a symbol representing a fixed definition with changing values
- a location in memory where data can be stored and retrieved

Text similarity

There are many different notions of similarity depending on the domain and the application

Today, we’ll look at some different tools

There is no one single tool that works in all domains
Text similarity approaches

sim(\text{A}, \text{B}) = ?

A: When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

B: When the defendant walked into the courthouse with his attorney, the crowd turned their backs on him.

How can we do this?

The basics: text overlap

Texts that have overlapping words are more similar

A: When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

B: When the defendant walked into the courthouse with his attorney, the crowd turned their backs on him.

Word overlap: a numerical score

Idea 1: number of overlapping words

A: When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

B: When the defendant walked into the courthouse with his attorney, the crowd turned their backs on him.

\( sim(T_1, T_2) = 11 \) problems?

Word overlap problems

- Doesn’t take into account word order
- Related: doesn’t reward longer overlapping sequences

A: defendant his the When lawyer into walked backs him the court, of supporters and some the victim turned their backs him to.

B: When the defendant walked into the courthouse with his attorney, the crowd turned their backs on him.

\( sim(T_1, T_2) = 11 \)
Word overlap problems

Doesn't take into account length

A: When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

B: When the defendant walked into the courthouse with his attorney, the crowd truned their backs on him. I ate a large banana at work today and thought it was great!

\[ \text{sim}(T_1, T_2) = 11 \]

Word overlap problems

Doesn't take into account synonyms

A: When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

B: When the defendant walked into the courthouse with his attorney, the crowd truned their backs on him.

\[ \text{sim}(T_1, T_2) = 11 \]

Word overlap problems

Doesn't take into account spelling mistakes

A: When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

B: When the defendant walked into the courthouse with his attorney, the crowd turned their backs on him.

\[ \text{sim}(T_1, T_2) = 11 \]

Word overlap problems

Treats all words the same

A: When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

B: When the defendant walked into the courthouse with his attorney, the crowd turned their backs on him.
Word overlap problems

May not handle frequency properly

A: When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him. I ate a banana and then another banana and it was good!

B: When the defendant walked into the courthouse with his attorney, the crowd turned their backs on him. I ate a large banana at work today and thought it was great!

Word overlap: sets

What is the overlap, using set notation?
- $|A \cap B|$ the size of the intersection

How can we incorporate length/size into this measure?

- Jaccard index (Jaccard similarity coefficient)
  \[ J(A,B) = \frac{|A \cap B|}{|A \cup B|} \]
- Dice's coefficient
  \[ Dice(A,B) = \frac{2|A \cap B|}{|A| + |B|} \]
Word overlap: sets

\[ J(A,B) = \frac{|A \cap B|}{|A \cup B|} \quad \text{and} \quad \text{Dice}(A,B) = \frac{2|A \cap B|}{|A|+|B|} \]

How are these related?

\[ = \frac{|A \cap B|}{|A - B| + |B - A| + |A \cap B|} \]

\[ = \frac{2|A \cap B|}{|A - B| + |B - A| + 2|A \cap B|} \]

Dice’s coefficient gives twice the weight to overlapping words

Set overlap

Our problems:
- word order
- length
- synonym
- spelling mistakes
- word importance
- word frequency

Bag of words representation

When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

What information do we lose?
Bag of words representation

For now, let's ignore word order:

Obama said banana repeatedly last week on tv, "banana, banana, banana"

(4, 1, 1, 0, 0, 0, 0, 0, 0, ...)

"Bag of words representation": multi-dimensional vector, one dimension per word in our vocabulary.

Frequency of word occurrence

Vector based word

A  
<table>
<thead>
<tr>
<th>term</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1: when</td>
<td>1</td>
</tr>
<tr>
<td>a2: the</td>
<td>2</td>
</tr>
<tr>
<td>a3: defendant</td>
<td>3</td>
</tr>
<tr>
<td>a4: and</td>
<td>1</td>
</tr>
<tr>
<td>a5: courthouse</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

B  
<table>
<thead>
<tr>
<th>term</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>b1: when</td>
<td>1</td>
</tr>
<tr>
<td>b2: the</td>
<td>2</td>
</tr>
<tr>
<td>b3: defendant</td>
<td>1</td>
</tr>
<tr>
<td>b4: and</td>
<td>0</td>
</tr>
<tr>
<td>b5: courthouse</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Multi-dimensional vectors, one dimension per word in our vocabulary.

How do we calculate the similarity based on these vectors?

Vector based similarity

We have a $|V|$-dimensional vector space.

Terms are axes of the space.

Documents are points or vectors in this space.

Very high-dimensional.

This is a very sparse vector - most entries are zero.

What question are we asking in this space for similarity?
Vector based similarity

Similarity relates to distance

We’d like to measure the similarity of documents in the $|V|$ dimensional space

What are some distance measures?

Distance measures

Euclidean (L2)

$$\text{dist}(A,B) = \sqrt{\sum (a_i - b_i)^2}$$

Manhattan (L1)

$$\text{dist}(A,B) = \sum |a_i - b_i|$$

What do these mean for our bag of word vectors?

Distance can be problematic

Which $d$ is closest to $q$ using one of the previous distance measures?

Which do you think should be closer?

The Euclidean (or L1) distance between $q$ and $d_2$ is large even though the distribution of words is similar
Use angle instead of distance

Thought experiment:
- take a document \( d \)
- make a new document \( d' \) by concatenating two copies of \( d \)
- "Semantically" \( d \) and \( d' \) have the same content

What is the Euclidean distance between \( d \) and \( d' \)?
What is the angle between them?
- The Euclidean distance can be large
- The angle between the two documents is 0

From angles to cosines

Cosine is a monotonically decreasing function for the interval \([0^\circ, 180^\circ]\)

![Graph showing cosine function](https://www.youtube.com/watch?v=iZhEcRvMA-M)

How do we calculate the cosine between two vectors?
Cosine of two vectors

\[ A \cdot B = \|A\| \|B\| \cos \theta \]

\[ \cos \theta = \frac{A \cdot B}{\|A\| \|B\|} = \frac{A}{\|A\|} \cdot \frac{B}{\|B\|} \]

Dot product between unit length vectors

Cosine as a similarity

\[ \text{sim}_{\text{cos}}(A, B) = A \cdot B = \sum_{i=1}^{n} a_i b_i \]

Just another distance measure, like the others:

\[ \text{dist}_{L_2}(A, B) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2} \]

\[ \text{dist}_{L_1}(A, B) = \sum_{i=1}^{n} |a_i - b_i| \]

For bag of word vectors, what does this do?

Cosine as a similarity

\[ \text{sim}_{\text{cos}}(A, B) = A \cdot B = \sum_{i=1}^{n} a_i b_i \]

Only words that occur in both documents count towards similarity

Words that occur more frequently in both receive more weight
Length normalization

A vector can be length-normalized by dividing each of its components by its length.

Often, we’ll use $L_2$ norm (could also normalize by other norms):

$$\|v\|_2 = \sqrt{\sum x_i^2}$$

Dividing a vector by its $L_2$ norm makes it a unit (length) vector.

Unit length vectors

In many situations, normalization improves similarity, but not in all situations.

Normalized distance measures

**Cosine**

$$\text{sim}_{\cos}(A, B) = \frac{A \cdot B}{\|A\|_2 \cdot \|B\|_2} = \frac{\sum a_i b_i}{\sqrt{\sum a_i^2} \sqrt{\sum b_i^2}}$$

**$L_2$**

$$\text{dist}_{L_2}(A, B) = \sqrt{\sum (a'_i - b'_i)^2}$$

**$L_1$**

$$\text{dist}_{L_1}(A, B) = \sum |a'_i - b'_i|$$

$A'$ and $B'$ are length normalized versions of the vectors.

Distance measures

**Cosine**

$$\text{sim}_{\cos}(A, B) = A \cdot B = \sum a_i b_i$$

**$L_2$**

$$\text{dist}_{L_2}(A, B) = \sqrt{\sum (a_i - b_i)^2}$$

**$L_1$**

$$\text{dist}_{L_1}(A, B) = \sum |a_i - b_i|$$

Cosine is the most common measure. Why do you think?
Distance measures

<table>
<thead>
<tr>
<th>Distance Measure</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cosine</td>
<td>( \text{sim}<em>{\text{cos}}(A, B) = A \cdot B = \sum</em>{i=1}^{n} a_i b_i )</td>
</tr>
<tr>
<td>L2</td>
<td>( \text{dist}<em>{L2}(A, B) = \sqrt{\sum</em>{i=1}^{n} (a_i - b_i)^2} )</td>
</tr>
<tr>
<td>L1</td>
<td>( \text{dist}<em>{L1}(A, B) = \sum</em>{i=1}^{n}</td>
</tr>
</tbody>
</table>

- L1 and L2 penalize sentences for not having words, i.e. if \( a \) has it but \( b \) doesn’t.
- Cosine can be significantly faster since it only calculates over the intersection.

Our problems

Which of these have we addressed?
- word order
- length
- synonym
- spelling mistakes
- word importance
- word frequency

Word overlap problems

Treats all words the same

**A:** When the defendant and his lawyer walked into the court, some of the victim supporters turned their backs to him.

**B:** When the defendant walked into the courthouse with his attorney, the crowd turned their backs on him.

Ideas?
Word importance

Include a weight for each word/feature

A
- When 1 \( w_1 \)
- the 2 \( w_2 \)
- defendant 1 \( w_3 \)
- and 1 \( w_4 \)
- courthouse 0 \( w_5 \)
- ...

B
- When 1 \( w_1 \)
- the 2 \( w_2 \)
- defendant 1 \( w_3 \)
- and 0 \( w_4 \)
- courthouse 1 \( w_5 \)
- ...

Distance + weights

We can incorporate the weights into the distances

Think of it as either (both work out the same):
- preprocessing the vectors by multiplying each dimension by the weight
- incorporating it directly into the similarity measure

\[
sim_{\text{cos}}(A, B) = \frac{A \cdot B}{\sqrt{\sum_{i} (w_ia_i)^2} \sqrt{\sum_{i} (wibia_i)^2}}
\]

Idea: use corpus statistics

- the
- defendant

What would be a quantitative measure of word importance?

Document frequency

document frequency (DF) is one measure of word importance

Terms that occur in many documents are weighted less, since overlapping with these terms is very likely
- In the extreme case, take a word like the that occurs in almost EVERY document

Terms that occur in only a few documents are weighted more
**Document vs. overall frequency**

The overall frequency of a word is the number of occurrences in a dataset, counting multiple occurrences.

**Example:**

<table>
<thead>
<tr>
<th>Word</th>
<th>Overall frequency</th>
<th>Document frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>insurance</td>
<td>10440</td>
<td>3997</td>
</tr>
<tr>
<td>try</td>
<td>10422</td>
<td>8760</td>
</tr>
</tbody>
</table>

Which word is a more informative (and should get a higher weight)?

**Document frequency**

<table>
<thead>
<tr>
<th>Word</th>
<th>Collection frequency</th>
<th>Document frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>insurance</td>
<td>10440</td>
<td>3997</td>
</tr>
<tr>
<td>try</td>
<td>10422</td>
<td>8760</td>
</tr>
</tbody>
</table>

Document frequency is often related to word importance, but we want an actual weight. Problems?

\[
sim_{cos}(A,B) = \frac{\sum_i (A_i \cdot B_i)}{\sqrt{\sum_i (A_i)^2} \cdot \sqrt{\sum_i (B_i)^2}}
\]

**From document frequency to weight**

<table>
<thead>
<tr>
<th>Word</th>
<th>Collection frequency</th>
<th>Document frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>insurance</td>
<td>10440</td>
<td>3997</td>
</tr>
<tr>
<td>try</td>
<td>10422</td>
<td>8760</td>
</tr>
</tbody>
</table>

Weight and document frequency are inversely related:
- Higher document frequency should have lower weight and vice versa.
- Document frequency is unbounded.
- Document frequency will change depending on the size of the data set (i.e. the number of documents).

**Inverse document frequency**

\[
idf_w = \log \frac{N}{df_w}
\]

- IDF is inversely correlated with DF.
- Higher DF results in lower IDF.
- N incorporates a dataset dependent normalizer.
- Log dampens the overall weight.
### IDF example, suppose \( N = 1 \) million

<table>
<thead>
<tr>
<th>term</th>
<th>df(t)</th>
<th>idf(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>calpurnia</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>animal</td>
<td>100</td>
<td>0.01</td>
</tr>
<tr>
<td>sunday</td>
<td>1,000</td>
<td>0.001</td>
</tr>
<tr>
<td>fly</td>
<td>10,000</td>
<td>0.0001</td>
</tr>
<tr>
<td>under</td>
<td>100,000</td>
<td>0.00001</td>
</tr>
<tr>
<td>the</td>
<td>1,000,000</td>
<td>0.0000001</td>
</tr>
</tbody>
</table>

What are the IDFs assuming log base 10?

<table>
<thead>
<tr>
<th>term</th>
<th>df(t)</th>
<th>idf(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>calpurnia</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>animal</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>sunday</td>
<td>1,000</td>
<td>3</td>
</tr>
<tr>
<td>fly</td>
<td>10,000</td>
<td>2</td>
</tr>
<tr>
<td>under</td>
<td>100,000</td>
<td>1</td>
</tr>
<tr>
<td>the</td>
<td>1,000,000</td>
<td>0</td>
</tr>
</tbody>
</table>

There is one idf value/weight for each word.

### IDF example, suppose \( N = 1 \) million

<table>
<thead>
<tr>
<th>term</th>
<th>df(t)</th>
<th>idf(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>calpurnia</td>
<td>1</td>
<td>1,000,000</td>
</tr>
<tr>
<td>animal</td>
<td>100</td>
<td>10,000</td>
</tr>
<tr>
<td>sunday</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>fly</td>
<td>10,000</td>
<td>100</td>
</tr>
<tr>
<td>under</td>
<td>100,000</td>
<td>10</td>
</tr>
<tr>
<td>the</td>
<td>1,000,000</td>
<td>1</td>
</tr>
</tbody>
</table>

What if we didn't use the log to dampen the weighting?
TF-IDF

One of the most common weighting schemes

\[ \text{TF} = \text{term frequency} \]

\[ \text{IDF} = \text{inverse document frequency} \]

\[ a'_i = a_i \times \log \frac{N}{df_i} \]

TF-IDF (word importance weight)

We can then use this with any of our similarity measures!

Stoplists: extreme weighting

Some words like ‘a’ and ‘the’ will occur in almost every document

- IDF will be 0 for any word that occurs in all documents
- For words that occur in almost all of the documents, they will be nearly 0

A stoplist is a list of words that should not be considered (in this case, similarity calculations)

- Sometimes this is the most frequent words
- Often, it’s a list of a few hundred words manually created

Stoplist

Two main benefits

- More fine grained control: some words may not be frequent, but may not have any content value (alas, teh, gosh)
- Often does contain many frequent words, which can drastically reduce our storage and computation

Any downsides to using a stoplist?

- For some applications, some stop words may be important
Text similarity so far...

Set based – easy and efficient to calculate
- word overlap
- Jaccard
- Dice

Vector based
- create a feature vector based on word occurrences (or other features)
- Can use any distance measures
  - L1 (Manhattan)
  - L2 (Euclidean)
  - Cosine (most common)
- Normalize the length
- Feature/dimension weighting
  - inverse document frequency (IDF)