Admin

How did assignment 1 finish up?

Assignment 2 out soon (two part assignment)
  □ First part due Friday (work through calculations by hand)

Videos!

Independence

Two variables are independent if they do not affect each other

For two independent variables, knowing the value of one does not change the probability distribution of the other variable
  □ the result of the toss of a coin is independent of a roll of a dice
  □ price of tea in England is independent of the whether or not you get an A in NLP

Independent or Dependent?

You catching a cold and a butterfly flapping its wings in Africa

Miles per gallon and driving habits

Height and longevity of life
Independent variables

How does independence affect our probability equations/properties?

If A and B are independent, written $A \perp B$

- $P(A|B) = P(A)$
- $P(B|A) = P(B)$

What does that mean about $P(A,B)$?

Conditional Independence

Dependent events can become independent given certain other events

Examples,
- height and length of life
- "correlation" studies
- size of your lawn and length of life

I USED TO THINK CORRELATION IMPLIED CAUSATION.
THEN I TOOK A STATISTICS CLASS. NOW I DON'T.
SOUNDS LIKE THE CLASS HELPED. WELL, MORE.

http://xkcd.com/552/

If A, B are conditionally independent given C, $A \perp B | C$

- $P(A,B | C) = P(A | C) P(B | C)$
- $P(A | B, C) = P(A | C)$
- $P(B | A, C) = P(B | C)$
- but $P(A,B) \neq P(A)P(B)$
Assume independence

Sometimes we will assume two variables are independent (or conditionally independent) even though they're not

Why?
- Creates a simpler model
  - $p(X,Y)$ many more variables than just $p(X)$ and $p(Y)$
  - May not be able to estimate the more complicated model

Language modeling

What does natural language look like?

More specifically in NLP, probabilistic model

$p(\text{sentence})$
- $p(\text{"I like to eat pizza"})$
- $p(\text{"pizza like I eat"})$

Often is posed as: $p(\text{word} \mid \text{previous words})$ – or some other notion of context
- $p(\text{"pizza"} \mid \text{"I like to eat"})$
- $p(\text{"garbage"} \mid \text{"I like to eat"})$
- $p(\text{"run"} \mid \text{"I like to eat"})$

Language modeling

How might these models be useful?
- Language generation tasks
  - machine translation
  - summarization
  - simplification
  - speech recognition
  - …
- Text correction
  - spelling correction
  - grammar correction

Ideas?

$p(\text{"I like to eat pizza"})$
$p(\text{"pizza like I eat"})$
$p(\text{"pizza"} \mid \text{"I like to eat"})$
$p(\text{"garbage"} \mid \text{"I like to eat"})$
$p(\text{"run"} \mid \text{"I like to eat"})$
Look at a corpus

Language modeling

I think today is a good day to be me

Language modeling is about dealing with data sparsity!

Probabilistic Language modeling

A probabilistic explanation of how the sentence was generated

Key idea:
- break the generation process into smaller steps
- estimate the probabilities of these smaller steps
- the overall probability is the product of the steps

Many approaches:
- n-gram language modeling
  - Start at the beginning of the sentence
  - Generate one word at a time based on the previous words
- syntax-based language modeling
  - Construct the syntactic tree from the top down
  - e.g. context free grammar
  - eventually at the leaves, generate the words
- Neural language models
  - Predict the likelihood of the word based on the context
  - Often allows for generalization beyond the lexical strings
n-gram language modeling

I think today is a good day to be me

Our friend the chain rule

Step 1: decompose the probability

\[ P(\text{I think today is a good day to be me}) = \]

\[ P(\text{I } | <\text{start}> ) \times \]

\[ P(\text{think } | <\text{start}> ) \times \]

\[ P(\text{today } | \text{think}) \times \]

\[ P(\text{is } | \text{today}) \times \]

\[ P(\text{a } | \text{today is}) \times \]

\[ P(\text{good } | \text{today is a}) \times \]

... How can we simplify these?

The n-gram approximation

Assume each word depends only on the previous n-1 words (e.g. trigram: three words total)

\[ P(\text{is } | \text{I think today}) \approx P(\text{is } | \text{think today}) \]

\[ P(\text{a } | \text{I think today is}) \approx P(\text{a } | \text{today is}) \]

\[ P(\text{good } | \text{I think today is a}) \approx P(\text{good } | \text{is a}) \]

Estimating probabilities

How do we find probabilities? \( P(\text{is } | \text{think today}) \)

Get real text, and start counting (MLE)!

\[ P(\text{is } | \text{think today}) = \frac{\text{count}(\text{think today is})}{\text{count}(\text{think today})} \]
Estimating from a corpus

Corpus of sentences
(e.g. gigaword corpus)

\[ \text{n-gram language model} \]

\[ \text{?} \]

I am a happy Pomona College student.

count all of the trigrams

\[ \text{<start> <start> I} \]
\[ \text{<start> I am} \]
\[ \text{I am a} \]
\[ \text{a happy} \]
\[ \text{a happy Pomona} \]
\[ \text{happy Pomona College} \]
\[ \text{Pomona College student} \]
\[ \text{College student .} \]
\[ \text{student . <end>} \]
\[ \text{. <end> <end>} \]

why do we need
\[ \text{<start> and <end>}? \]

Do we need to count anything else?

\[ p(c | a b) = \frac{\text{count(a b c)}}{\text{count(a b)}} \]
Estimating from a corpus

1. Go through all sentences and count trigrams and bigrams
   - usually you store these in some kind of data structure

2. Now, go through all of the trigrams and use the count and the bigram count to calculate MLE probabilities
   - do we need to worry about divide by zero?

\[ p(c|a,b) = \frac{\text{count}(a,b,c)}{\text{count}(a,b)} \]

Applying a model

Given a new sentence, we can apply the model

\[ p(\text{Pomona College students are the best} . ) = \] 

\[ p(\text{Pomona} | <\text{start}> <\text{start}> ) \times 
\[ p(\text{College} | <\text{start}> \text{Pomona} ) \times 
\[ p(\text{students} | \text{Pomona College} ) \times 
\[ \vdots 
\[ p( <\text{end}> | . <\text{end}> ) \times \]

Generating examples

We can also use a trained model to generate a random sentence

Ideas?

\[ <\text{start}> <\text{start}> \]

\[ p( A | <\text{start}> <\text{start}> ) \]
\[ p(\text{Apples} | <\text{start}> <\text{start}> ) \]
\[ p( I | <\text{start}> <\text{start}> ) \]
\[ p(\text{The} | <\text{start}> <\text{start}> ) \]
\[ \vdots \]
\[ p(\text{Zebras} | <\text{start}> <\text{start}> ) \]

Repeat!

\[ p( \text{are} | <\text{start}> \text{Zebras} ) \]
\[ p(\text{eat} | <\text{start}> \text{Zebras} ) \]
\[ p(\text{think} | <\text{start}> \text{Zebras} ) \]
\[ p(\text{and} | <\text{start}> \text{Zebras} ) \]
\[ \vdots \]
\[ p(\text{mostly} | <\text{start}> \text{Zebras} ) \]

Generating examples
Generation examples

Unigram

are were that ères mammal naturally built describes jazz territory heteromyids film tenor prime live founding must an vas fees negro legal gate in on beside . provincial sam ; stephensian simply spaces stretched performance double-entry grove replacing station across to burma . repairing ères capital about double reached omnibus el time believed what hotels parameter jurisprudence words syndrome to ères proficiency is administrators ères offices hilarious institutionalized remains writer royalty demis , ères tyson , and objective , instructions seem timekeeper has ères valley ères magnitudes for love on ères from alladakt , , one central enlightened . to , ères is belongs fame they the corrected . on in pressure %NUMBER% her flavored ères derogatory is won metcard indirectly of crop duty lean northbound ères ères dancing similarity ères named ères berkeley . off scale overtime . each manifield stripes dânu traffic assets and at alpha popularity town .

Bigrams

the wikipedia county , mexico . maurice ravel . it is require that is sparta , where functions . most widely admired . halogens chimalli cast jason against test site .

Trigrams

is widespread in north africa in june %NUMBER% %NUMBER% units were built by with .

dual videos spiritual are considered irid , this season was an extratropical cyclone .

the british railways ’ s strong and a spot .

Evaluation

We can train a language model on some data

How can we tell how well we ’ re doing?

- for example
  - bigrams vs. trigrams
  - 100K sentence corpus vs. 100M
  - …
A very good option: extrinsic evaluation

If you're going to be using it for machine translation
- build a system with each language model
- compare the two based on their approach for machine translation

Sometimes we don't know the application

Can be time consuming

Granularity of results

Common NLP/machine learning/AI approach

A good model should do a good job of predicting actual sentences
**Evaluation**

Pros: Fine for comparing two models
Cons: Doesn’t give us a sense of how well any model is doing

---

**The problem**

Which of these sentences will have a higher probability based on a language model?

- I like to eat banana peels.
- I like to eat banana peels with peanut butter.

Since probabilities are multiplicative (and between 0 and 1), they get smaller for longer sentences.

---

**The solution: perplexity**

\[
prob(w_{1:n}) = \prod_{i=2}^{n} p(w_i|w_{1:i-1})
\]

average the probabilities geometric mean

\[
PP(w_{1:n}) = \sqrt[n]{\prod_{i=1}^{n} p(w_i|w_{1:i-1})}
\]
Calculating perplexity in practice

$log \left( \frac{1}{\prod_{i=1}^{n} p(w_i|w_{i-1})} \right) = log \left( \frac{1}{\prod_{i=1}^{n} p(w_i|w_{i-1})} \right)^{1/n} = log \left( \frac{1}{\prod_{i=1}^{n} p(w_i|w_{i-1})} \right) = -\frac{\sum_{i=1}^{n} \log p(w_i|w_{i-1})}{n} = -\frac{\sum_{i=1}^{n} \log p(w_i|w_{i-1})}{n} \times \frac{1}{n}

Average log prob per word!

Calculating perplexity

$PP(w_{1:n}) = \sqrt[n]{\prod_{i=1}^{n} p(w_i|w_{i-1})} = 10^{\frac{-\sum_{i=1}^{n} \log p(w_i|w_{i-1})}{n}}$

• This is often how it’s calculated (and how we’ll calculate it)

• Avoid underflow from multiplying too many small probabilities together

Another view of perplexity

Weighted average branching factor

- number of possible next words that can follow a word or phrase

- measure of the complexity/uncertainty of text (as viewed from the language model’s perspective)
Smoothing

What if our test set contains the following sentence, but one of the trigrams never occurred in our training data?

\[
P(I \text{ think today is a good day to be me}) = \\
P(I | \text{ <start> } \text{ <start> } \text{ <start>}) \times \\
P(\text{ think } | \text{ I } \text{ <start>}) \times \\
P(\text{ today } | \text{ I think}) \times \\
P(\text{ is } | \text{ think today}) \times \\
P(\text{ a } | \text{ today is}) \times \\
P(\text{ good } | \text{ is a}) \times \\
\ldots
\]

If any of these has never been seen before, prob = 0!

A better approach

\[
p(z | x \ y) = ?
\]

Suppose our training data includes

\[
\ldots x \ y \ a \ldots \\
\ldots x \ y \ d \ldots \\
\ldots x \ y \ \ldots \\
\]

but never: x y z

We would conclude

\[
p(a | x \ y) = 1/3r \\
p(d | x \ y) = 2/3r \\
p(z | x \ y) = 0/3r
\]

Is this ok?

Intuitively, how should we fix these?

Smoothing the estimates

Basic idea:

\[
p(a | x \ y) = 1/3r \quad \text{reduce} \\
p(d | x \ y) = 2/3r \quad \text{reduce} \\
p(z | x \ y) = 0/3r \quad \text{increase}
\]

Discount the positive counts somewhat

Reallocate that probability to the zeroes

Remember, it needs to stay a probability distribution

Other situations

\[
p(z | x \ y) = ?
\]

Suppose our training data includes

\[
\ldots x \ y \ a \ldots \ (100 \ \text{times}) \\
\ldots x \ y \ d \ldots \ (100 \ \text{times}) \\
\ldots x \ y \ \ldots \ (300 \ \text{times})
\]

but never: x y z

Suppose our training data includes

\[
\ldots x \ y \ a \ldots \\
\ldots x \ y \ d \ldots \\
\ldots x \ y \ \ldots \\
\]

\text{Is this the same situation as before?}
### Smoothing the estimates

Should we conclude

- $p(a | xy) = 1/3$? **reduce**  
  $p(c | a b) = \frac{\text{count}(a b c)}{\text{count}(a b)}$

- $p(d | xy) = 2/3$? **reduce**

- $p(z | xy) = 0/3$? **increase**

Readjusting the estimate is particularly important if:

- the denominator is small …
  - $1/3$ probably too high, $100/300$ probably about right

- numerator is small …
  - $1/300$ is probably too high, $100/300$ probably about right

### Add-one (Laplacian) smoothing

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>xya</td>
<td>1</td>
<td>1/3</td>
</tr>
<tr>
<td>xyb</td>
<td>0</td>
<td>0/3</td>
</tr>
<tr>
<td>xyc</td>
<td>0</td>
<td>0/3</td>
</tr>
<tr>
<td>xyd</td>
<td>2</td>
<td>2/3</td>
</tr>
<tr>
<td>xye</td>
<td>0</td>
<td>0/3</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>xyz</td>
<td>0</td>
<td>0/3</td>
</tr>
<tr>
<td><strong>Total xy</strong></td>
<td><strong>3</strong></td>
<td><strong>3/3</strong></td>
</tr>
</tbody>
</table>

### Add-one (Laplacian) smoothing

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>xya</td>
<td>$100$</td>
<td>$100/300$</td>
</tr>
<tr>
<td>xyb</td>
<td>0</td>
<td>$0/300$</td>
</tr>
<tr>
<td>xyc</td>
<td>0</td>
<td>$0/300$</td>
</tr>
<tr>
<td>xyd</td>
<td>$200$</td>
<td>$200/300$</td>
</tr>
<tr>
<td>xye</td>
<td>0</td>
<td>$0/300$</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>xyz</td>
<td>0</td>
<td>$0/300$</td>
</tr>
<tr>
<td><strong>Total xy</strong></td>
<td><strong>300</strong></td>
<td><strong>300/300</strong></td>
</tr>
</tbody>
</table>
What happens if we're now considering a vocabulary of 20,000 words?

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{ya}$</td>
<td>1</td>
<td>1/3</td>
<td>2</td>
</tr>
<tr>
<td>$x_{yb}$</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
</tr>
<tr>
<td>$x_{yc}$</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
</tr>
<tr>
<td>$x_{yd}$</td>
<td>2</td>
<td>2/3</td>
<td>3</td>
</tr>
<tr>
<td>$x_{ye}$</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
</tr>
<tr>
<td>$x_{xz}$</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
</tr>
</tbody>
</table>

Total $xy$ | 3   | 3/3 | 29  | 29/29 |

Any problem with this?

The general smoothing problem

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>see the abacus</td>
<td>1</td>
<td>1/3</td>
<td>2</td>
</tr>
<tr>
<td>see the abbot</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
</tr>
<tr>
<td>see the abduct</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
</tr>
<tr>
<td>see the above</td>
<td>2</td>
<td>2/3</td>
<td>3</td>
</tr>
<tr>
<td>see the Abram</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
</tr>
<tr>
<td>see the zygote</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
</tr>
</tbody>
</table>

Total $xyz$ | 3   | 3/3 | 20003 | 20003/20003 |

Any problem with this?
Add-lambda smoothing

A large dictionary makes novel events too probable.

Instead of adding 1 to all counts, add $\lambda = 0.01$.

- This gives much less probability to novel events.

<table>
<thead>
<tr>
<th>Term</th>
<th>Count</th>
<th>Count + $\lambda$</th>
<th>Count + $\lambda$ + Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>see the abacus</td>
<td>1</td>
<td>1/3</td>
<td>1.01</td>
</tr>
<tr>
<td>see the abbot</td>
<td>0</td>
<td>0/3</td>
<td>0.01</td>
</tr>
<tr>
<td>see the abduct</td>
<td>0</td>
<td>0/3</td>
<td>0.01</td>
</tr>
<tr>
<td>see the above</td>
<td>2</td>
<td>2/3</td>
<td>2.01</td>
</tr>
<tr>
<td>see the Abram</td>
<td>0</td>
<td>0/3</td>
<td>0.01</td>
</tr>
<tr>
<td>see the zygote</td>
<td>0</td>
<td>0/3</td>
<td>0.01</td>
</tr>
<tr>
<td>Total</td>
<td>3</td>
<td>3/3</td>
<td>203</td>
</tr>
</tbody>
</table>