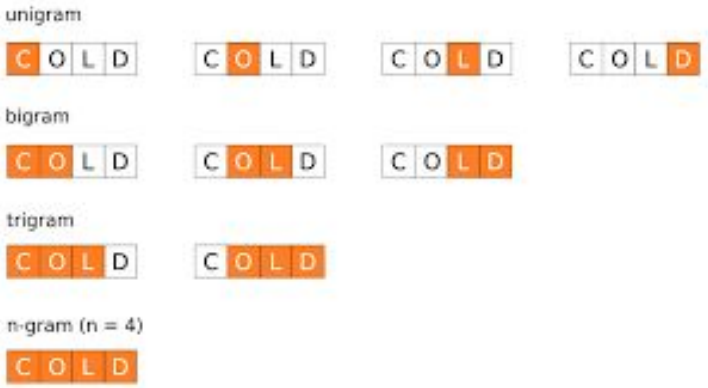
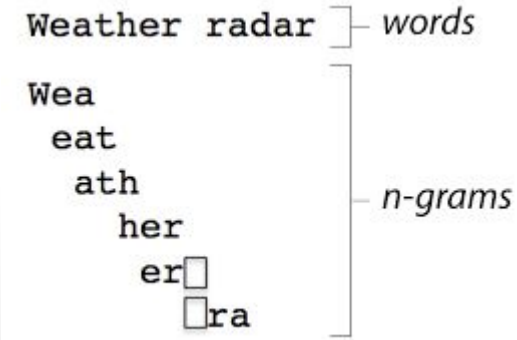
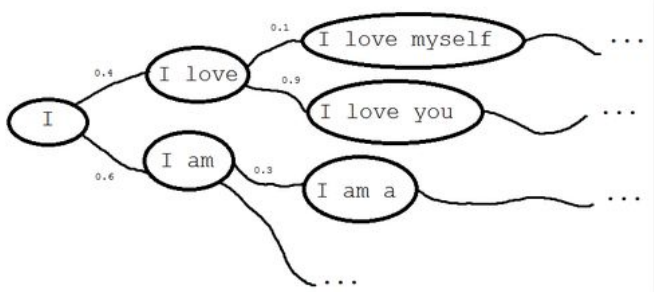
Topic 5 (Pt 2) :
Information Theory & Language Model Evaluation







Skip-gram Modeling

- Skip-gram is a technique based on n-gram modelling whereby n-grams are formed (bigrams, tri-grams, etc.) but additionally allows for tokens to be "skipped".
- A *k-skip-n-grams* for a sentence $w_1 \ldots w_n$ can be defined as :

$$\{w_{i_1}, w_{i_2}, \ldots w_{i_n} \mid \sum_{j=1}^{n} i_j - i_{j-1} < k\}$$

- Skip-grams for a certain *skip distance k* allow a total of *k* or less skips to construct the n-gram. Thus, "4-skip-n-gram" results include 4 skips, 3 skips, 2 skips, 1 skip, and 0.

Skip-gram Modeling Example

- Example (bigram & 2-skip-bigrams):

"Insurgents killed in ongoing fighting"

  - bigrams = {insurgents killed, killed in, in ongoing, ongoing fighting}.

  - 2-skip-bigrams = {insurgents killed, insurgents in, insurgents ongoing, killed in, killed ongoing, killed fighting, in ongoing, in fighting, ongoing fighting}

Skip-gram Modeling Example

- Example:

  "Insurgents killed in ongoing fighting"

  - Tri-grams = {insurgents killed in, killed in ongoing, in ongoing fighting}.
  - 2-skip-tri-grams = {insurgents killed in, insurgents killed ongoing, insurgents killed fighting, insurgents in ongoing, insurgents in fighting, insurgents ongoing fighting, killed in ongoing, killed in fighting, killed ongoing fighting, in ongoing fighting}.

- Skips are never allowed to cross sentence boundaries.

Example Size of Skip-grams Extracted

- Source : Guthrie, D. et al. “A closer look at skip-gram modeling, LREC”
- A lot of extra contextual information are extracted and be beneficial provided that these skip-grams truly expand the representation of context.

Bi-grams

Sentence Length	Bi-grams	1- skip	2-skip	3-skip	4-skip
5	4	7	9	10	10
10	9	17	24	30	35
15	14	29	30	50	60
20	19	37	54	70	85

Tri-grams

Sentence Length	Tri-grams	1- skip	2-skip	3-skip	4-skip
5	3	7	10	10	10
10	8	22	40	60	80
15	13	37	70	110	155
20	18	53	100	160	230

Table 1: Number of n-grams vs. number of k-skip n-grams produced

Information Theory

- A usable measure of the information we get from observing the occurrence of an event having probability p
- Ignore any particular features of the event, and only observe whether or not it happened.
- Think of an event as the observance of a symbol whose probability of occurring is p
- Define information in terms of the probability p

Maximum Likelihood Estimation MLE)

- Calculate word strings in corpus, take fraction

$$P(w_i|w_1 \ldots w_{i-1}) = \frac{c(w_1 \ldots w_i)}{c(w_1 \ldots w_{i-1})}$$

- Trigram model:

<s> i live in cheras . </s>

<s> i am an undergraduate student . </s>

<s> my college is in gombak . </s>

- P(live | <s> i) = c(<s> i live)/c(<s> i) = 1/2 = 0.5
- P(am | <s> i) = c(<s> i am)/c(<s> i) = 1/2 = 0.5

Likelihood

- Likelihood is the probability of some observed data (the *test set* W_{test}), given the model M (*training set*)

$$P(W_{test}|M) = \prod_{w \in W_{test}} P(w|M)$$

i live in nara
i am a student
my classes are hard

P(w="i live in nara"|M) = $2.52*10^{-21}$

X

P(w="i am a student"|M) = $3.48*10^{-19}$

X

P(w="my classes are hard"|M) = $2.15*10^{-34}$

=

$1.89*10^{-73}$

Log Likelihood

- Likelihood uses very small numbers = underflow
- Taking the log resolves this problem ($\log_{10}$ or $\log_e$)

$$\log P(W_{test}|M) = \sum_{w \in W_{test}} \log P(w|M)$$

i live in nara
i am a student
my classes are hard

log P(w="i live in nara"|M) = -20.58
+
log P(w="i am a student"|M) = -18.45
+
log P(w="my classes are hard"|M) = -33.67
=
-72.60

Problem with Unseen Sequences

- Suppose we want to evaluate bigram models on our *test data* containing this sentence:

  *"I could not submit my assignment because my **cat ate** it"*

- Further suppose that our training data did not have the sequence ***"cat ate"*** .

1. What is the probability of $p(ate \mid cat)$ according to our ***bigram model***?
2. What is the ***probability of the above sentence*** based on bigram approximation?
3. Higher N-gram models suffer more from previously unseen word sequences (why???).

Data Sparsity

- Data sparsity is a serious problem in language modeling
- Describe the phenomenon of not observing enough data in a corpus (i.e., training data is small) to model language accurately.
- True observations about the distribution and pattern of language cannot be made because there is not enough data to see the true distribution.

Data Sparsity (cnt…)

Test data:

I could not submit my assignment because my cat ate it

I did not submit my assignment because the virus ate it

Training data:

<s> I could not** finish **my** homework **because** there is a **virus** **in my** hard drive **</s>

<s> I could** sleep all night since **it** is so quiet and peaceful **</s>

<s> I** have **not** eaten since **my cat** hide **the** plate **</s>

<s>** We don't like to do our **assignment** **</s>

- P(I|<s>) = 3/4 = 0.75 P(ate|cat) = 0/1 = 0
- P(not|could) = 1/2 = 0.5 P(ate|virus) = 0/1 = 0

Smoothing

- Need better estimators because MLEs give us a lot of zeros
- Zeros are bad for any statistical estimator
- A distribution without zeros is "smoother"
- The concept of *zakah*: Take from the rich (seen n-grams) and distribute to the poor (unseen n-grams)
- Distribute the probability mass from the seen events to the unseen events

Laplace Smoothing

- Simplest and oldest smoothing technique
- Just **add 1** to **all n-gram counts** including the unseen ones
- Also known as "**add one smoothing**"

Laplace Smoothing (cnt…)

- Unigrams

$$P_{LAP}(w_i) = \frac{C(w_i) + 1}{N + V}$$

Tokens + Types

- Bigrams

$$P_{LAP}(w_j|w_i) = \frac{P_{LAP}(w_i, w_j)}{P_{LAP}(w_i)} = \frac{C(w_i, w_j) + 1}{C(w_i) + V}$$

Evaluating Language Models

- **Entropy** and **perplexity** are used to evaluate the quality of language models
- In language modeling, low entropy and low perplexity **are better**
- Low perplexity and low entropy means that **a language model is good at knowing what comes next in a sentence/text**
- **LESS SURPRISES ARE GOOD!**

Entropy

- A metric to measure how much information there is in a particular information
  - how well a given grammar matches a given language?
  - how predictive a given n-gram grammar is (what the next word could be)?
- The average(expected) amount of the information from the event
- A lower bound on the number of bits it would take to encode a certain decision or piece of information in the optimal coding scheme

Entropy (cnt…)

- Formula :

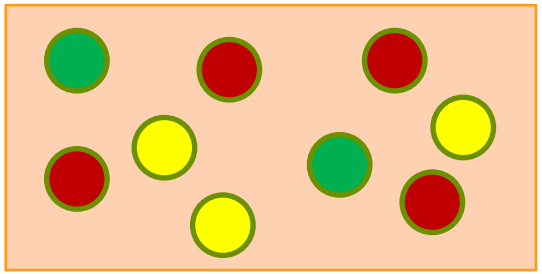
$$H(X) = -\sum_{x \in \chi} p(x) \log_2 p(x)$$

- where X is a random variable, $p(x)$ is the probability function of an event and χ (chi) is the set of sentences, words, letters or parts of speech that we are predicting
- ** Note : log base 2 means our information are measured in binary bits.

Entropy (cnt…)

- Given two grammars/models/speech recognition systems and a corpus, we can use entropy to tell us which grammar/model/system better matches the corpus.

Example (1)

- $\chi =$ 

- Three possible selection outcomes: red, yellow or green ball ($V = 3$, V types)

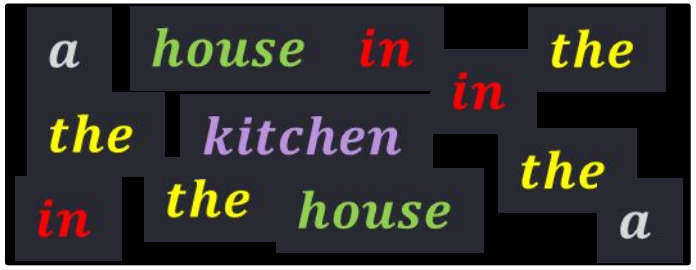
$$H(X) = -\sum_{x \in \chi} p(x)\, log_2\, p(x)$$

$$H(X) = -[p(red) * log_2\, p(red) + p(yellow) * log_2\, p(yellow) + p(green) * log_2\, p(green)]$$

$$H(X) = -\left[\frac{4}{9} * log_2\, \frac{4}{9} + \frac{3}{9} * log_2\, \frac{3}{9} + \frac{2}{9} * log_2\, \frac{2}{9}\right]$$

$$= -[-0.51997 - 0.4822 - 0.5283] = 1.53047 \text{ (bits)}$$

Example (2)

- $\chi =$ 

- Five possible selection outcomes: **in**, **the**, **a**, **kitchen** or **house** (V = 5, N= 12) → V=#word types

$$H(X) = -[p(in) * log_2\, p(in) + p(the) * log_2\, p(the) + p(a) * log_2\, p(a) + p(kitchen) * log_2\, p(kitchen) + p(house)\ log_2\, p(house)]$$

$$= -\left[\frac{3}{12} * log_2\, \frac{3}{12} + \frac{4}{12} * log_2\, \frac{4}{12} + \frac{2}{12} * log_2\, \frac{2}{12} + \frac{1}{12} * log_2\, \frac{1}{12} + \frac{2}{12} * log_2\, \frac{2}{12}\right]$$

$$= -[-0.5 - 0.528321 - 0.430827 - 0.29875 - 0.430827]$$

$$= -[-2.188725] = 2.188725\ (\text{bits})$$

Per word Entropy

- Average negative $\log_2$ likelihood per word

$$H(X) = -\frac{1}{N}\sum_{x \in \chi} p(x)\, log_2\, p(x)$$

- where N is the number of items in the set

Example Per word Entropy

$$H(W_{test}|M) = \frac{1}{|W_{test}|} \sum_{w \in W_{test}} -\log_2 P(w|M)$$

i live in nara
i am a student
my classes are hard

$\log_2$ P(w="i live in nara"|M)= 68.43

$\log_2$ P(w="i am a student"|M)= 61.32

$\log_2$ P(w="my classes are hard"|M)= 111.84

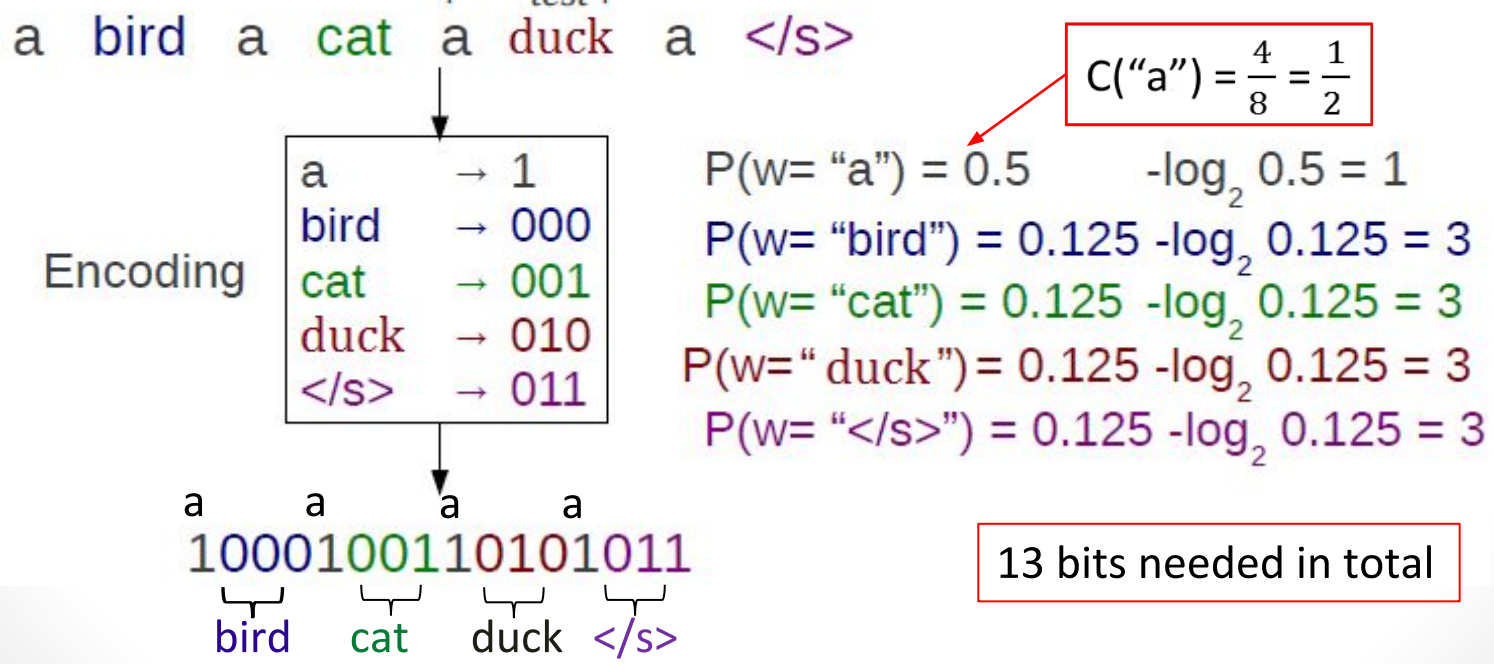
(68.43 + 61.32 + 111.84) / # of words= 12 = 20.13

- **Note: We usually include counts of <s> and/or </s> in the corpus, thus # of words in this case will be 15 (if only </s> is considered) or 18 (include both <s> & </s>)

Entropy and Compression

- *Entropy H* is also the **average number of bits needed to encode information** (Shannon's information theory)

$$H = \frac{1}{|W_{test}|} \sum_{w \in W_{wtest}} -\log_2 P(w|M)$$



Perplexity

- The weighted average number of choices a random variable has to make
- The probability of the entire test set normalized by the number of words
- Given test set *W* with words $w_1, \ldots, w_N$
- Treat entire test set as one word sequence

$$PP(T) = P(w_1, \ldots, w_N)^{-1/N}$$

- Using the probability chain rule (e.g., bigram model), we can write this as

$$PP(T) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}}$$

Perplexity

- How surprised are you on average by what comes next ?
- **Related to entropy** in that **perplexity is equal to two to the power of per-word entropy**

$$PP = 2^{H(X)}$$

- (Mainly because it makes more impressive numbers)
- For uniform distributions, equal to the size of vocabulary

Example Perplexity

- From previous example (H(X)):

i live in nara
i am a student
my classes are hard

$\log_2$ P(w="i live in nara"|M)= 68.43
$\log_2$ P(w="i am a student"|M)= 61.32
$\log_2$ P(w="my classes are hard"|M)= 111.84
\# of words= 12
20.13

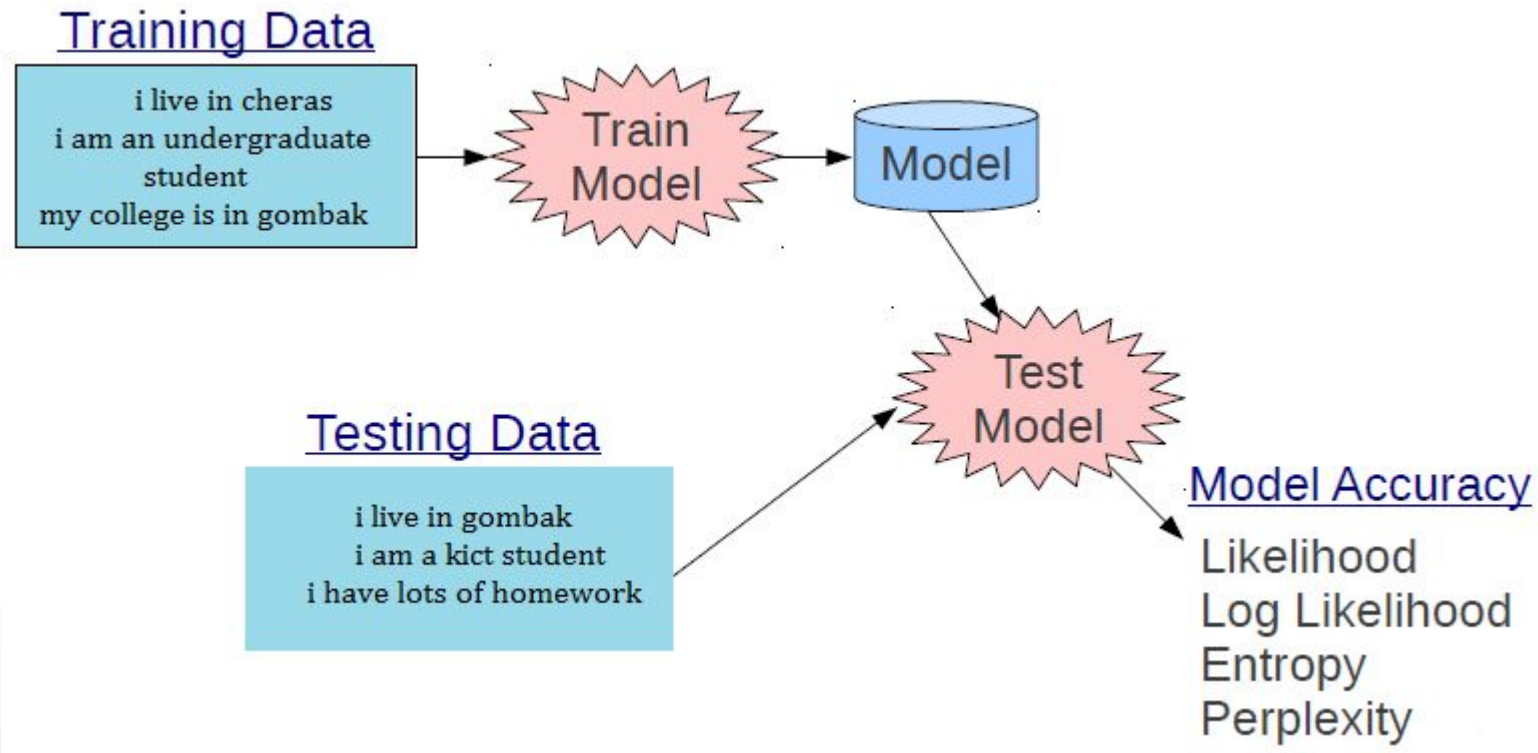
$$PP = 2^{H(X)}$$
$$= 2^{20.13}$$
$$= 1147450.11209$$
$$= 1.1475 \times 10^4$$

Training a Language Model

- A language model must be trained on a large corpus of text to estimate good parameter values
- Models can be evaluated based on its ability to predict a high probability for a disjoint (held-out) test corpus (testing on the training corpus will give an optimistically biased estimate)
- Ideally, the training & test corpus should be representative of the actual application data

Experimental Setup

- Use training and test set



Exercise (Entropy & Perplexity)

- Using a bigram model, find the per word entropy for the following test sentence
- Apply Laplace smoothing on all words to distribute some probabilities to words with zero counts
- Find the perplexity of the test sentence
- Submit your solution in the next class

Training set

- *<s> I could not finish my homework because there is a virus in my hard drive </s>*
- *<s> I could sleep all night since it is so quiet and peaceful </s>*
- *<s> I have not eaten since my cat hide the plate </s>*
- *<s> We don't like to do our assignment </s>*

Test sentence

<s> I could not submit my assignment because my cat ate it </s>