Probability distribution

- The probability (distribution) $p(w)$ of a word w in a corpus with s distinct words is:

$$p(w) = \frac{\text{count}(w)}{\sum_{i=1}^{s} \text{count}(w_i)}$$

- This estimation is referred to as "maximum likelihood"
- Distribution which answers the question:
  - "If I select randomly a word from a text, what is the probability that this word is the word w?"

Joint Probability

- Joint probabilities:
  - Study of two random variables at the same time
  - Example:
    - the words w_1 and w_2 that appear one after the other (a bigram), we model this with the distribution $p(w_1, w_2)$
    - If the occurrence of two words in bigrams is independent, we can write:
      - $p(w_1, w_2) = p(w_1)p(w_2)$, this assumption is probably wrong!
- Estimating the joint probability of two variables:
  - the same way this is done for a single variable

$$p(w_1, w_2) = \frac{\text{count}(w_1, w_2)}{\sum_{w_1', w_2'} \text{count}(w_1', w_2')}$$

Towards Language Models

- But words appear in sequence!
- Given a sequence of words, can we predict the next word?

$$p(w_n|w_1, \ldots, w_{n-1})$$

- Conditional probability, written $p(a|b)$
  - answers the question: if the random variable $B = b$, what is the probability that the variable A takes the 'value' a
    - mathematically: $p(a|b) = \frac{p(b,a)}{p(b)}$ ($p(b,a)$ joint probability)
    - note: if a and b are independent then $p(a|b) = p(a)$

“Chain rule”

- We have
  - $p(w_1, w_2) = p(w_1)p(w_2|w_1)$
  - $p(w_1, w_2, w_3) = p(w_1)p(w_2|w_1)p(w_3|w_1, w_2)$
  - etc.

Bayes rule

- The rule:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

- Obtained from:

$$p(x,y) = p(y,x)$$

$$p(x|y)p(y) = p(y|x)p(x)$$

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

Language Models

- Can answer the question
  - What is the probability that this string of words is correct?
    - "The cat is dead" => very good (≈ 1.0)
    - "The cat is talkative" => quite poor ($\approx ?$)
    - "Is the dead cat" => very poor (≈ 0.0)
- Use
  - Automatic Speech Recognition
  - Machine translation
  - Language recognition
  - Optical Character Recognition
  - ....

Language Models

- Given a string of words $W = w_1 w_2 w_3 w_4 \ldots w_n$
  - *chain rule*

$$p(w_1, w_2, w_3, \ldots, w_n)$$
$$=$$
$$p(w_1)p(w_2|w_1)p(w_3|w_1, w_2) \ldots p(w_n|w_1, w_2, \ldots, w_{n-1})$$

- Markov assumption (use past history of limited length)
  - Only the k preceding words belong to the history
  - Model of order k
- Example: a model of order 1 (bigram)

$$p(w_1, w_2)$$
$$=$$
$$p(w_1)p(w_2|w_1)$$

Estimate the n-grams Probabilities

- Collect frequencies of words and word sequences in very large corpus
  - Several million words
- Using “chain rule”:

$$p(w_2|w_1) = \frac{\text{count}(w_1, w_2)}{\text{count}(w_1)}$$

Model size?

- For each n-gram, one must store a probability
- If we assume a vocabulary of 20,000 words

Model	Max number of parameters
0^{th} order (unigram)	$20{,}000$
1^{st} order (bigram)	$20{,}000^2 = 400$ million
2^{nd} order (trigram)	$20{,}000^3 = 8$ trillion
3^{rd} order (4-gram)	$20{,}000^4 = 160$ quadrillion

- Trigram LM are mostly used

Practical example

- From a corpus of 275 million words written in English
  - newspapers such as “Wall Street Journal”

Model	Number of n-grams
1-gram	716,706
2-gram	12,537,755
3-gram	22,174,483

Visualisation

- http://www.visualizing.org/stories/visualizing-ngrams
- https://books.google.com/ngrams

Quality of a language model

- Entropy of a sequence $w_1, w_2, \dots, w_n$

$$H(w_1, w_2, \dots, w_n) = -\sum_{w_1 \dots w_n \in \Sigma^n} p(w_1 \dots w_n) \log_2 p(w_1 \dots w_n)$$

- per word Entropy of a sequence $w_1, w_2, \dots, w_n$

$$\frac{1}{n} H(w_1, w_2, \dots, w_n) = -\frac{1}{n} \sum_{w_1 \dots w_n \in \Sigma^n} p(w_1 \dots w_n) \log_2 p(w_1 \dots w_n)$$

- Entropy of a language $L = \{w_1, w_2, \dots, w_n | 0 < n < \infty\}$

$$H(L) = -\lim_{n \to \infty} \frac{1}{n} H(w_1 \dots w_n)$$

- Perplexity

$$perplexity(L) = 2^{H(L)}$$

  - A language model m is better than m' if it assign lower perplexity to the test corpus $w_1 \dots w_n$

Example: 1-gram

- Training set [14 tokens ($^1/_{14} \cong 0.0714$)]
  - *there is a big house*
  - *i buy a house*
  - *they buy the new house*
- Model

$p(there) = 0.0714$	$p(is) = 0.0714$	$p(a) = 0.1429$
$p(big) = 0.0714$	$p(house) = 0.2143$	$p(i) = 0.0714$
$p(buy) = 0.1429$	$p(they) = 0.0714$	$p(the) = 0.0714$
$p(new) = 0.0714$		

- Test sentence S: *they buy a big house*
  - $p(S) = \underbrace{0.0714}_{they} \times \underbrace{0.1429}_{buy} \times \underbrace{0.0714}_{a} \times \underbrace{0.1429}_{big} \times \underbrace{0.2143}_{house} = 0.0000231$

Example: 2-gram

- Training set
  - *there is **a** big house*
  - ***i** buy **a** house*
  - *they buy **the** new house*
- Model

$p(big\|a) = 0.5$	$p(is\|there) = 1$	$p(buy\|they) = 1$
$p(house\|a) = 0.5$	$p(buy\|i) = 1$	$p(a\|buy) = 0.5$
$p(new\|the) = 1$	$p(house\|big) = 1$	$p(the\|buy) = 0.5$
$p(a\|is) = 1$	$p(house\|new) = 1$	$p(they\|< s >) = 0.333$
$p(there\|< s >) = 0.333$	$p(i\|< s >) = 0.333$	

- Test sentence $S1$: *they buy a big house*
  - $p(S) = \underbrace{0.333}_{they} \times \underbrace{1}_{buy} \times \underbrace{0.5}_{a} \times \underbrace{0.5}_{big} \times \underbrace{1}_{house} = 0.0833$

Problem of Unknown Events

- Training set
  - *there is **a** big house*
  - ***i** buy **a** house*
  - *they buy **the** new house*

- Let sentence $S2$
  - *they buy a new house*
- The bigram "*a new*" has never been seen
  - $p(S2) = 0$?!
  - But the sentence is correct

Problem of Unknown Events

- Two types of "zeroes"
  - Unknown words
    - Problem dealt with a label "$UNKNOWN$"
    - The probability $p(UNKNOWN)$ is estimated
      - Tend to over-estimate the probability
        - Smoothing mechanisms
  - Unknown N-grams
    - Smoothing by giving them a low probability (but not zero!)
    - Fall back (backoff) to a lower-order n-gram
  - Give a non-zero probability to un-observed events
    - This is not a maximum likelihood estimate

More on that topic with Laurent Besacier