Spoken and Natural Language Understanding Practical Assignment 1

Submitted by Jiahui Dai

March 28, 2025

Accompanying material: Assignment_1.ipynb

1 Zipf's Law

List of unique words sorted to their frequency in descending order

Shown in accompanying Jupyter Notebook.

Discussion on findings

Zipf's Law states that in a large collection of words, the frequency of any word is inversely proportional to its rank:

$$f \propto \frac{1}{r}$$

where

- f: frequency of word
- r: rank of the word

This means that the most frequent word occurs twice as often as the second most frequent, three times as often as the third most frequent, and so on.

With reference to the jungle book dataset, words that are short in length occur in higher frequencies compared to words with longer lengths. For example, "the", "and", "of" occur in higher frequency compared to "produce", "subscribe", "newsletter" of lower frequency.

To verify Zipf's law on a textual corpus, using the jungle book dataset, chi-square goodness of fit test is performed [**web:chi_sq_test**].

- H_0: The observed frequencies are equal to the expected frequencies.
- H_1: The observed frequencies are not equal to the expected frequencies.

$$\begin{aligned}
\chi^2_{\text{statistics}} &= 19760.11496822835 \\
\chi^2_{\text{critical}} &= 5107.674219300448 \\
\chi^2_{\text{statistics}} &> \chi^2_{\text{critical}} && \text{(Reject } H_0\text{)}
\end{aligned}$$

As the H_0 is rejected, this suggests that observed frequencies are not equal to the expected frequencies, and that Zipf's Law does not hold.

With reference to Figure 1, there is a general downward trend of frequency compared to rank. The statistical analysis shows that the observed frequency is not equal to the expected frequency, suggests that the jungle book dataset contains words that does not follow Zipf's Law significantly. If Zipf's Law holds, the observed line (blue) should form a straight line with a slope close to -1, as depicted by the expected line (red) in Log-Log curve. However, the observed line does not follow a slope close to -1 in Log-Log curve.

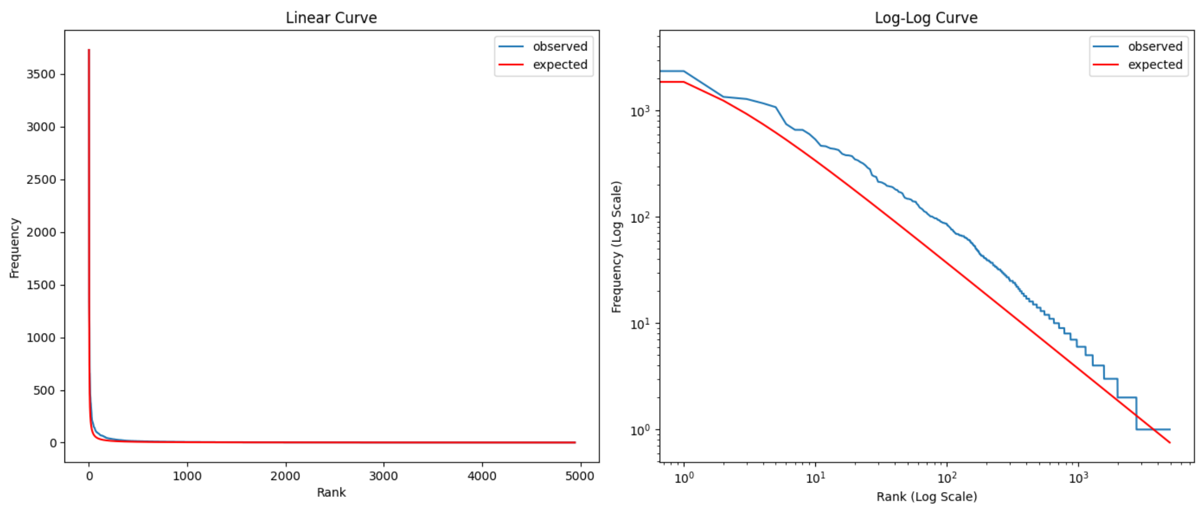

Figure 1: Linear and Log-Log Curves.
Observed line (blue) follows the frequency of words observed in the jungle book dataset. Expected line (red) follows the theoretical word frequency according to Zipf's law.

2 Mutual Information

Observations

Pointwise mutual information (PMI) is a metric that compares the relative frequency of two outcomes occurring together to the probability of either outcome occurring independently. A positive PMI value means that the words co-occur more frequently than would be expected, whereas a negative PMI value means they cooccur less frequently than would be expected. A PMI value of 0 suggests that the words occur independent of each other (occurance by chance) [**web:pmi**].

Many word-pairs with high PMI values (Table 1) are often two-word phrases to convey a certain idea, like "United States" and "tree tops", whereas for word-pairs with low PMI values (Table 2) are pairs that either contain "the" or "and", or phrases that is grammatically incorrect in the English language.

Discussion of the validity of the independence assumption for unigram models

A unigram model assumes that each word in a sentence or document is independent of all other words, i.e., the probability of a word occurring is independent of the words that came before or after it.

$$P(w_1, w_2, w_3, \dots) = \prod_{i=1}^{n} P(w_i)$$

where $P(w_i)$ is the probability of each word w_i.

To assess the validity of the independence assumption, the conditional probabilities $P(w_i|w_1, w_2, \dots, w_{i-1})$ (observed) is calculated and compare to the unigram probability $P(w_i)$ (expected) using chi-square goodness of fit test.

Chi-square independence test is performed [**web:chi_sq_test**].

- H_0: Each word in a document is independent of all other words.
- H_1: Each word in a document is dependent of all other words.

$$\begin{aligned} \chi^2_{\text{statistics}} &= 572.4405452818747 \\ \chi^2_{\text{critical}} &= 57009.8446715156 \\ \chi^2_{\text{statistics}} &< \chi^2_{\text{critical}} && \text{(Fail to reject } H_0\text{)} \end{aligned}$$

As the H_0 fails to be rejected, this suggests that observed frequencies (the conditional probabilities $P(w_i|w_1, w_2, \dots, w_{i-1})$) and expected frequencies (unigram probability $P(w_i)$) are the independent, and thus the independence assumption for unigram models is valid.

w1	w2	pmi
machua	appa	8.54334
united	states	8.30218
literary	archive	8.23318
cold	lairs	7.69419
archive	foundation	7.57926
bandar	log	7.41487
petersen	sahib	7.38589
stretched	myself	7.33937
paragraph	f	7.33937
hind	legs	7.23466
fore	paws	7.15704
hind	flippers	7.13457
tree	tops	7.02921
troop	horse	7.02804
bath	room	7.00289
twenty	yoke	6.98519
paragraph	e	6.97472
electronic	works	6.95225
master	words	6.91588
whole	line	6.90405
years	ago	6.88771
within	days	6.87221
bring	news	6.86936
waingunga	river	6.76685
killing	grounds	6.75158
council	rock	6.71941
villagers	lived	6.67154
monkey	folk	6.64622
black	panther	6.53574
copyright	laws	6.49858

Table 1: 30 word pairs with highest pmi value

w1	w2	pmi
the	the	−5.50709
and	and	−4.58736
he	the	−4.28052
the	he	−4.28052
the	to	−3.7514
of	of	−3.47152
i	and	−3.30003
they	the	−3.23617
to	he	−3.21798
was	and	−3.1072
for	and	−2.96939
is	and	−2.96079
and	is	−2.96079
had	the	−2.95281
of	in	−2.88121
little	the	−2.76773
to	i	−2.69736
he	in	−2.67243
but	and	−2.65494
that	and	−2.64759
of	and	−2.64314
at	and	−2.58825
we	the	−2.57623
not	and	−2.55316
man	the	−2.47589
he	and	−2.43436
were	the	−2.43042
and	of	−2.42
of	for	−2.41147
is	of	−2.40287

Table 2: 30 word pairs with lowest pmi value

3 Wikipedia Language Model

Motivation [n-gram:ch3]

An n-gram is a sequence of n words and a a 2-gram (bigram) is a two-word sequence of words. Here, we use a 2-gram language model to solve this task.

To predict the conditional probability of the next word in a bigram model, Markov assumption is applied:

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-1}) \tag{1}$$

where

- $P(w_n|w_{1:n-1})$: probability of word w_n given all previous words $w_{1:n-1}$
- $P(w_n|w_{n-1})$: probability of word w_n given the previous word w_{n-1}

The maximum likelihood estimation is used to estimate probabilities:

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}, w_n)}{C(w_{n-1})} \tag{2}$$

where

- $C(w_{n-1}, w_n)$: count of bigram of w_{n-1} and w_n frequency
- $C(w_{n-1})$: count of w_{n-1} frequency

The probabilities are log to prevent numerical underflow and stored. To return the original probability, the exp of the logprob is calculated.

The perplexity of a language model on a test set is the inverse probability of the test set, normalized by the number of words and is used as an evaluation matric in this model. The higher the probability of the word sequence, the lower the perplexity, the better the model. It is defined as

$$Perplexity(w) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}} \tag{3}$$

However, there lies a challenge in the trained model. If a bigram exists in the test dataset and not in the train dataset, the probability of the bigram is zero, which affects the calculation of perplexity as we cannot divide by zero. Laplace smoothing is the simplest method to tackle this challenge. It is defined as

$$P_{\text{Laplace}}(w_n|w_{n-1}) = \frac{C(w_{n-1}, w_n) + 1}{C(w_{n-1}) + V} \tag{4}$$

where

- V: number of unique in the corpus

To finetune the model, add-k smoothing is used (further improvement from Laplace smoothing) when optimising with validation dataset. It is defined as

$$P_{\text{Add-k}}(w_n|w_{n-1}) = \frac{C(w_{n-1}, w_n) + k}{C(w_{n-1}) + kV} \tag{5}$$

Data Preprocessing Steps

Text Tokenisation The text data is broken into individual words (token), with punctuation, special characters and whitespace removed [**web:pre-processing**].

Train-Validation-Test Data Split This step is not performed here as the data provided is already split into training, validation and testing sets [**web:pre-processing**].

Method and Experiment Design

1. Preprocess the data, i.e. text tokenisation
2. Generate 2-grams and count the frequency of occurance of two consecutive words (w_1 and w_2)
3. Calculate 2-gram probabilities (Equation 5) with smoothing k
4. Evaluate the model by calculating perplexity (Equation 3)
5. Optimise the model with validation data and a different k value
6. Test the model with test data

Hyperparameters

- k (Equation 5) is updated to finetune the model

Evaluation Metrics

- Perplexity

Observations

Drawbacks

- High smoothing can reduce model specificity. With $\alpha = 100$, probabilities are spread more evenly, reducing the model ability to distinguish frequent and rare bigrams
- Limited context with 2-Gram model
- Low perplexity does not gurantee better text generation or predictive power in real world applications

Potential Improvements

- Use higher order N-Grams, e.g. 3-Gram or 4-Gram
- Experiment with different smoothing techniques
- Apply backoff or interpolation
- Increase training data

4 Additional Experiments

Not attempted