CS 533: Natural Language Processing **(Due: 02/04/20)**

Assignment 1

Name: Twisha Naik, *NetID:* tn268 *Students discussed with:* Prakruti Joshi, Keya Desai

Problem 1: Preliminaries

((1+1+1+1) + (1+1+1+1) + (1+1+1) = 11 points)

1. **(Probability)**

   (a) Var(X) = $E[(X-\mu)^2] = E[X^2 - 2\mu X + \mu^2]$
   By using linearity of expectation, Var(X) = $E[X^2] - 2\mu E[X] + E[\mu^2]$
   Now, E[X] = μ
   Thus, Var(X) = $E[X^2] - 2\mu^2 + E[\mu^2] = E[X^2] - \mu^2$

   (b) X = Outcome of a fair six-sided die
   Mean: E[X] = 3.50
   Variance: V(x) = 2.92
   Entropy: H(p_X) = 2.58

   (c) X = Outcome of biased die where outcome is always 6
   Mean: E[X] = 6
   Variance: V(x) = 0
   Entropy: H(p_X) = 0

   (d) p_X = Distribution of a fair six-sided die = (1/6, 1/6, 1/6, 1/6, 1/6, 1/6)
   $H(p_X, Cat(1/2, 0, 0, 0, 1/2, 0)) = \infty$
   $H(p_X, Cat(1/5, 1/5, 1/5, 1/5, 1/10, 1/10)) = 2.65$
   $H(p_X, Cat(1/6, 1/6, 1/6, 1/6, 1/6, 1/6)) = 2.58$

2. **(Linear Algebra)**

   (a) $\begin{bmatrix}-3\end{bmatrix}$

   (b) $\begin{bmatrix}13 & 5 & 3 & -2 \\ 28 & 14 & 9 & -8\end{bmatrix}$

   (c) $\begin{bmatrix}-176 \\ 988 \\ 51 \\ 82 \\ 135\end{bmatrix}$

   (d) Invalid - The dimensions are not compatible.

3. **(Optimization)**

   (a) 1. 3
   2. $-\infty$

   (b) $f(x) = \frac{1}{2}(x-3)^2 + 2$
   $f'(x) = x - 3$
   $f'(x) = 0 \implies x - 3 = 0 \implies x = 3$
   $f''(x) = 1 \implies$ **f"(3) = 1**
   First derivative of the function is zero when x=3. The second derivative of the function evaluated at x=3 is positive which implies that the function value at x=3 is the minima.

   (c) $f(x) = \frac{1}{3}(x-3)^3 + 2$
   $f'(x) = (x-3)^2$
   $f'(x) = 0 \implies (x-3)^2 = 0 \implies x = 3$
   $f''(x) = 2(x-3) \implies$ **f"(3) = 0**
   First derivative of the function is zero when x=3. The second derivative of the function evaluated at x=3 is zero which implies that nothing can be said about the function value at x=3.

Problem 2: n-Gram Models (4 + (2+1+1+1) = 9 points)

1. **(Relative Frequency Lemma)**

   (a) To prove the relative frequency lemma, we need to differentiate the objective equation twice, once with respect to lambda and the other time with respect to q_i.

$$\frac{\partial}{\partial\lambda}\sum_{i\in[n]} c_i log q_i - \lambda(1-\sum_{i\in[n]} q_i) = 0 \tag{1}$$

$$\frac{\partial}{\partial q_j}\sum_{i\in[n]} c_i log q_i - \lambda(1-\sum_{i\in[n]} q_i) = 0 \qquad \forall j\in[n] \tag{2}$$

Taking the derivative of objective equation w.r.t. λ, we get:

$$(1-\sum_{i\in[n]} q_i) = 0 \tag{3}$$

Taking the derivative of objective equation w.r.t. q_i, we get:

$$\begin{aligned} c_i/q_i + \lambda = 0 \qquad & \forall i\in[n] \\ \implies \lambda * q_i = -(c_i) \qquad & \forall i\in[n] \end{aligned} \tag{4}$$

Taking a summation over all the possible values for i,

$$\lambda = -N/1 \qquad (Because \sum_{i\in[n]} q_i = 1\, and \sum_{i\in[n]} c_i = N \text{ from equation (3)})$$

Thus, we get the value for (q, λ) as (c_i/N, -N). This proves Lemma 1 which says, $q_i^* = c_i/N$.

2. **(Maximum Likelihood Estimation (MLE) of the Trigram Language Model)**

   (a) Here, $[n] = V \cup EOS \cup BOS$ and the words x, x', x" $\in$ [n].
   Define $C_{x"} = \#(x, x, x")$ as a non-negative scalar associated with each x".
   Now, marginalising the given term on x", we get:

$$N = \sum_{x"} \#(x, x, x") = \#(x, x')$$

This term denotes the bigram counts that can be used to compute emperical MLE. Using Lemma 1, it can be shown that an emperical MLE estimate the trigram language model maximises the expected loglikelihood.
The MLE trigram model is as below:

$$\tilde{t}^{MLE}(x, x', x") = q^*_{x"} = C_{x"}/N = \#(x, x, x")/\#(x, x')$$

This equation also follows the intuition derived from the definition of conditional probability.

   (b) All non-zero MLE parameter values estimated from the corpus V^+ :

$\tilde{t}^{MLE}$ (the | BOS, BOS) = 3/3 = 1
$\tilde{t}^{MLE}$ (dog | BOS, the) = 1/3
$\tilde{t}^{MLE}$ (ignored | the, dog) = 1/1 = 1
$\tilde{t}^{MLE}$ (the | dog, ignored) = 1/1 = 1
$\tilde{t}^{MLE}$ (cat | ignored, the) = 1/1 = 1
$\tilde{t}^{MLE}$ (EOS | the, cat) = 1/2

$\tilde{t}^{MLE}$ (cat | BOS, the) = 1/3

$\tilde{t}^{MLE}$ (ate | the, cat) = 1/2
$\tilde{t}^{MLE}$ (the | cat, ate) = 1/1 = 1
$\tilde{t}^{MLE}$ (mouse | ate, the) = 1/1 = 1
$\tilde{t}^{MLE}$ (EOS | the, mouse) = 1/2

$\tilde{t}^{MLE}$ (mouse | BOS, the) = 1/3
$\tilde{t}^{MLE}$ (screamed | the, mouse) = 1/2
$\tilde{t}^{MLE}$ (EOS | mouse, screamed) = 1/1 = 1

(c) 1.32

(d) ∞

Problem 3: Programming

(2 + 1 + 1 + 1 + 2 + 1 + 3 + 1 = 12 points)

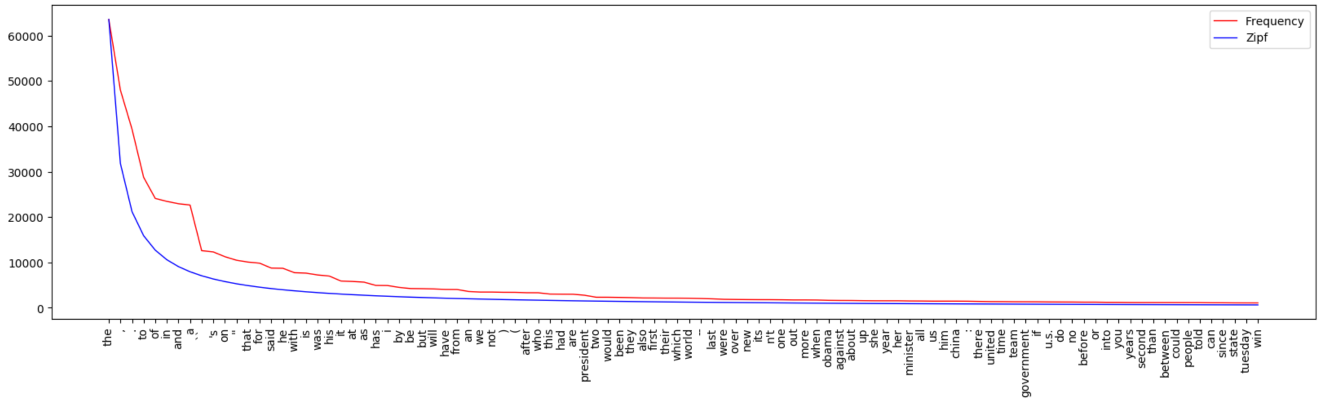
(Code must be submitted as well, with unambiguous commands for replicating reported results.)

1. Implemented ***count_ngrams*** in ***Tokenizer*** by adding the following line in the code:
   ngram = tuple(toks[(i-j):(i+1)])
   The correctness of the function was tested by the 'test_ngram_counts' function.

2. For different choices of the tokenizer, the vocabulary size observed are as follows:

Sr. No.	**Tokenizer**	**Vocabulary size**
1.	basic	69148
2.	nltk	41844
3.	wp	15716
4.	bpe	22581

3. Zipf's Law



Yes, as observed from the plot, Zipf's Law seems to hold approximately. It gets better with increasing rank of words based on the frequency.

4. Results of perplexity for basic bigram model using nltk tokenizer and vocab size 10,000 are as follows:
   Train perplexity = 70.967555
   Validation Perplexity = inf
   The validation perplexity is infinte because there might be words in the test data which were not present in the training data.

5. Laplace smoothing

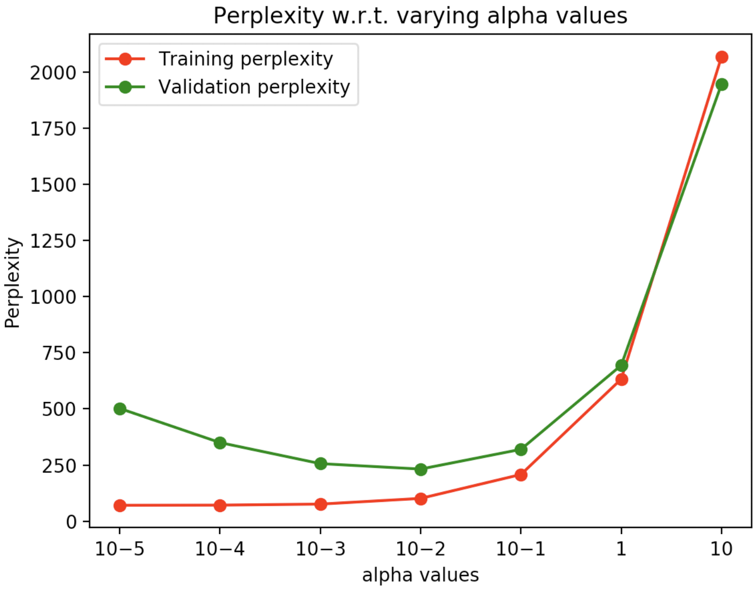


Figure 1: Perplexity w.r.t. varying alpha values for Laplace smoothing

- The perplexity value for training data keep on increasing with increase in alpha values. This is because with a larger alpha value, we are trying to make the model less accurate for training data. In other words, we are avoiding overfitting on the training data.
- The validation perplexity initially decreases and then again starts increasing. This is because, for a reasonable value of alpha, the model learns to generalise and work well on unseen data. When the value for alpha increases beyond a particular value, here alpha = 0.01, the model fails to learn the essence of the data.

Use of Laplace smoothing also fixes the issue of perplexity going to infinte as we add a term ”alpha*vocab_size” in the denominator which makes sure the denominator is always non-zero and positive.

6. Varying fraction of training data

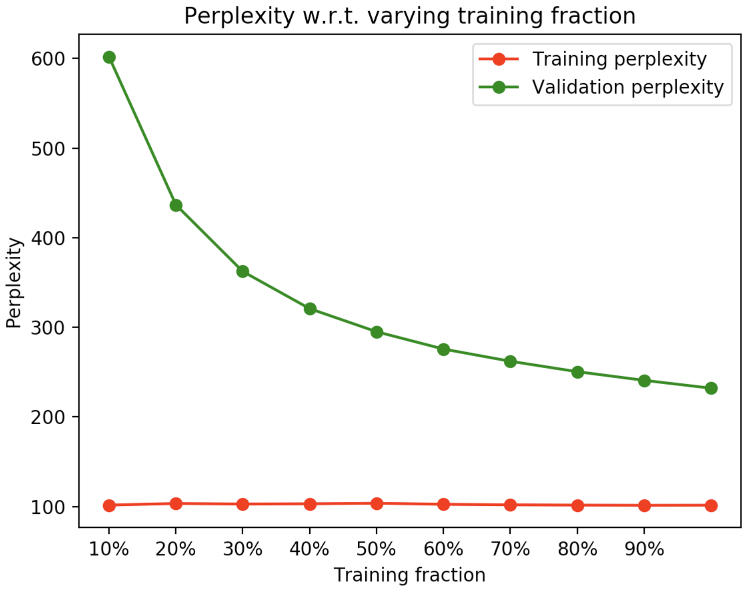


Figure 2: Perplexity w.r.t. varying training fraction of data

The perplexity on training data is almost constant nearly 100. It is constant because it is learning the training data and computing perplexity over the entire seen data no matter how much data is used.

The perplexity on the validation data keeps on decreasing as more data the model sees, the better it learns and it can generalise well.

7. Interpolation smoothing

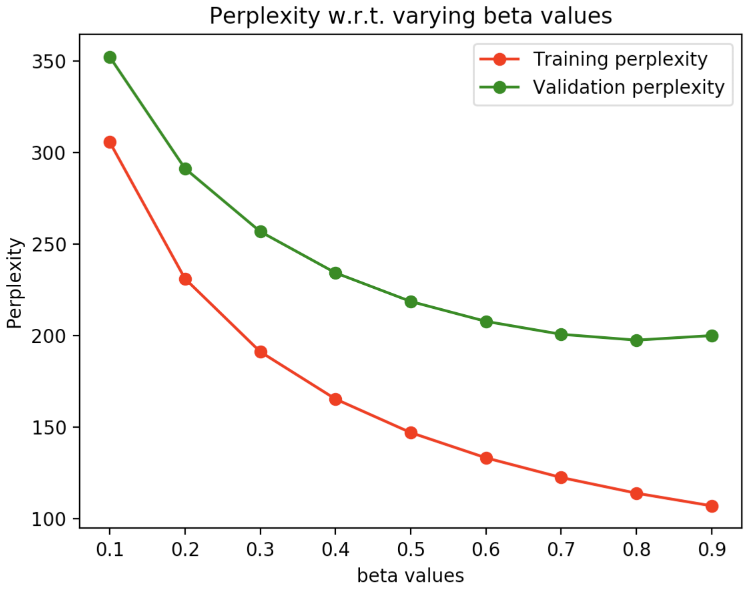


Figure 3: Perplexity w.r.t. varying beta values for Interpolation smoothing

The training perplexity keeps on decreasing as beta increases. Whereas, the validation perplexity decreases till beta value of 0.8 and increasing slightly at 0.9.
The value of beta signifies the weightage given to the bigrams and (1-beta) is the weightage given to the unigrams. As the weight of bigrams increases, the model gets better.

8. The best validation perplexity after exploring all the choices is obtained with interpolation smoothing at alpha=0.01 and beta=0.8.
   Value of best validation perplexity = 197.470875