1 Problem

The answer for this problem is divided into the following 2 parts:

A. Proof that the provided model does not form a proper probability distribution.

B. Suggestion on how to modify the above model to make it conform to a proper probability distribution.

Part A

Let's consider the following training corpus:

$$I\ am\ happy$$
$$I\ am\ glad$$

Here, vocabulary $v = \{I, am, happy, glad\}$. As per the given model, for a history "$I\ am$", sets $\mathcal{A}(I, am) = \{happy, sad\}$ and $\mathcal{B}(I, am) = \{I, am\}$. Thus, the sum of probabilities over possible next words w_i for the history "$I\ am$" using the given model is expressed as follows:

$$\begin{aligned} p(w_i|I, am)_{w_i \in v} &= p(I|I, am) + p(am|I, am) + p(happy|I, am) + p(glad|I, am) \\ &= p_3(I|I, am) + p_3(am|I, am) + p_1(happy|I, am) + p_1(glad|I, am) \\ &= \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = 2 \neq 1 \end{aligned}$$

This proves that there exists a history such that the sum of probabilities resulting from this model are not equal to one.

Part B

In order to make the model conform to a proper probability distribution we start by shaving off some count (∂) from the actual counts of the n-grams in set $\mathcal{A}$. This results into a missing probability mass that we can then distribute among the n-grams in set $\mathcal{B}$ (for the corresponding n) using Katz's back-off model. Let's introduce the discounted counts c^* and missing probability mass α as follows:

For trigrams,

$$c^*(w_{i-2}, w_{i-1}, w_i) = c(w_{i-2}, w_{i-1}, w_i) - \partial$$

$$\alpha(w_{i-2}, w_{i-1}) = 1 - \sum_{w_i \in \mathcal{A}(w_{i-2}, w_{i-1})} \frac{c^*(w_{i-2}, w_{i-1}, w_i)}{c(w_{i-2}, w_{i-1})}$$

For bigrams,

$$c^*(w_{i-1}, w_i) = c(w_{i-1}, w_i) - \partial$$

$$\alpha(w_{i-1}) = 1 - \sum_{w_i \in \mathcal{A}(w_{i-1})} \frac{c^*(w_{i-1}, w_i)}{c(w_{i-1})}$$

Before re-stating the original model using the above concepts, let's introduce a convenient notation for computing the Bigram back-off probability (from Katz's back-off model) to make the final model easier to read:

$$q_{BO}(w_i|w_{i-1}) = \begin{cases} \dfrac{c^*(w_{i-1}, w_i)}{c(w_{i-1})} & if\ w_i \in \mathcal{A}(w_{i-1}) \\ \alpha(w_{i-1}) \dfrac{p_{MLE}(w_i)}{\sum_{w_i \in \mathcal{B}(w_{i-1})} p_{MLE}(w_i)} & if\ w_i \in \mathcal{B}(w_{i-1}) \end{cases}$$

We can now re-define the model with the suggested changes as follows:

$$p(w_i|w_{i-2}, w_{i-i}) = \begin{cases} p_1(w_i|w_{i-2}, w_{i-1}) & if\ w_i \in \mathcal{A}(w_{i-2}, w_{i-1}) \\ p_2(w_i|w_{i-2}, w_{i-1}) & if\ w_i \in \mathcal{A}(w_{i-1})\ and\ w_i \in \mathcal{B}(w_{i-2}, w_{i-1}) \\ p_3(w_i|w_{i-2}, w_{i-1}) & if\ w_i \in \mathcal{B}(w_{i-1}) \end{cases}$$

Where:

$$p_1(w_i|w_{i-2}, w_{i-1}) = \frac{c^*(w_{i-2}, w_{i-1}, w_i)}{c(w_{i-2}, w_{i-1})}$$

$$p_2(w_i|w_{i-2}, w_{i-1}) = p_3(w_i|w_{i-2}, w_{i-1}) = \alpha(w_{i-2}, w_{i-1}) \frac{q_{BO}(w_i|w_{i-1})}{\sum_{w_i \in \mathcal{B}(w_{i-2}, w_{i-1})} q_{BO}(w_i|w_{i-1})}$$

Going back to our previous example, the sum of probabilities over possible next words w_i for the history "$I\ am$" in the modified model for $\partial = 0.5$ is expressed as follows:

$$p(w_i|I, am)_{w_i \in v} = p(I|I, am) + p(am|I, am) + p(happy|I, am) + p(glad|I, am)$$
$$= p_3(I|I, am) + p_3(am|I, am) + p_1(happy|I, am) + p_1(glad|I, am)$$
$$= \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} = 1$$

As seen in this example, the sum of probabilities for the history "I am" resulting from this model is equal to one.

2 Experiment

The answer to this problem is divided into the following 3 parts:

A. Discuss selection of Hyper-parameters, Pre-processing, computation of Perplexity, etc.
B. Results that show the perplexity outcomes of 6 models on a testing portion of the 3 corpora.
C. Observations made throughout the course of the experiments.
D. Bonus!

Part A

Hyper-parameters

The Hyper-parameters for all 6 models were picked based on their performance of a separate dev-training data set. Each corpus was first divided into training (80%) and test (20%) dataset. This training data set was then further divided into actual training and dev-training dataset with the intention of running the model trained using the actual training dataset on the corresponding dev-training dataset to pick the right Hyper-parameter values. See spreadsheet *HyperParameters.xlsx* in the tarfile for the results of these experiments.

Pre-processing

The following steps were performed on each corpus before any training and testing can take place:

1. Replace all the punctuation symbols with <<PUNC>>.
2. Replace all the numbers with <<NUM>>.
3. Prepare a dictionary that maps every word to a number (including <<START>>, <<NUM>>, <<PUNC>>, <<UNK>>, <<STOP>>).

4. Replace all words in the corpus from Step 2 with their corresponding number from the dictionary. (Words outside dictionary and words whose frequencies are below the set threshold of 1 are set to the number 0 that stands for <<UNK>>).

Perplexity

I used the following formulae to compute the Perplexity for the Testing dataset:

$$2^{-\frac{1}{M}\sum_{i=1}^{n}\log_2 x_i}$$

Where,

M is total number of words in Testing dataset.

n is total number of line in Testing dataset.

x is probability of the line i in Testing dataset.

Alpha (α) values for Katz's Back-off

I have used following derivation to calculate the α value (back-off value) for the bigrams (and in a similar way for trigrams):

$$\alpha(w_{i-1}) = 1 - \sum_{w_i \in \mathcal{A}(w_{i-1})} \frac{c^*(w_{i-1}, w_i)}{c(w_{i-1})}$$

$$= \frac{c(w_{i-1})}{c(w_{i-1})} - \sum_{w_i \in \mathcal{A}(w_{i-1})} \frac{c^*(w_{i-1}, w_i)}{c(w_{i-1})}$$

$$= \frac{c(w_{i-1})}{c(w_{i-1})} - \frac{\sum_{w \in \mathcal{A}(w_{i-1})}(c(w_{i-1}, w_i) - \partial)}{c(w_{i-1})}$$

$$= \frac{c(w_{i-1})}{c(w_{i-1})} - \frac{\sum_{w \in \mathcal{A}(w_{i-1})}(c(w_{i-1}, w_i)}{c(w_{i-1})} + \frac{\sum_{w \in \mathcal{A}(w_{i-1})}\partial}{c(w_{i-1})}$$

$$= \frac{c(w_{i-1})}{c(w_{i-1})} - \frac{c(w_{i-1})}{c(w_{i-1})} + \frac{\sum_{w \in \mathcal{A}(w_{i-1})}\partial}{c(w_{i-1})}$$

$$= \frac{\sum_{w \in \mathcal{A}(w_{i-1})}\partial}{c(w_{i-1})}$$

Part B

Each row of the following 2 tables outlines the result of a model using the specified Hyper-parameter and Testing dataset.

Table 1 Linear Interpolation Model results

Training Corpus ↓	Brown Testing	Reuters Testing	Gutenberg Testing	Hyper-parameters $(\lambda_1, \lambda_2, \lambda_3)$
Brown	263.1987	266.02393	365.453163	0.1, 0.5, 0.4
Reuters	476.668327	83.51189	513.028	0.4, 0.4, 0.2
Gutenberg	235.124	312.55	363.35087	0.2, 0.5, 0.3

Table 2 Back-off Model results

Training Corpus ↓	Brown Testing	Reuters Testing	Gutenberg Testing	Hyper-parameters (δ)
Brown	246.8522666	249.2161542	395.4021787	0.8
Reuters	509.8997045	72.01418707	496.5085913	0.7
Gutenberg	254.5581547	363.0040943	415.1649618	0.7

Part C

Comparison of Linear interpolation and Back-off model

In Linear Interpolation, we interpolate the probability estimates from all the n-grams by assigning them some weights whereas in Back-off model we back-off to a lower n-gram probability estimates if we do not find evidence for an existence for an n-gram at the current level.

Katz's Back-off is relatively slower than Linear Interpolation since it requires computing back-off probability estimates for a larger number of lower n-gram so as to distribute the missing mass in proportion to the probability estimates of the lower n-grams.

In Linear Interpolation model if the Unigram model is given 0-weight then the perplexity of the resulting model is Infinity if there exists a word such that it's history is not present in the Training corpus.

Transferring a model from one corpus to another at test time

Using a model with a corpus on which it was not trained results in an increased perplexity. As evident in Table 1 and 2, the perplexity numbers are higher where the Training and Testing dataset are not coming from the same corpus.

Comment about language used in the provided corpora

Looking at the perplexity numbers (above), languages in the Reuters and Brown corpus are relatively similar compared to Reuters and Gutenberg or Reuters and Brown (evident in the comparatively lower perplexity numbers for Brown, Reuters tests).

Within a corpus, the language in Reuters seems to be more similar in nature which is also evident in the low perplexity numbers for Reuters, Reuters tests.

Part D

I ran a few experiments in the Linear Interpolation model for the Bonus question and found made some interesting observations (tabulated below):

Experiment	Perplexity observed for Reuters testing dataset	Hyper-parameters
50% of Reuters training data is used to adapt to training model created by Brown corpus	100.536	Hyper-parameters used were adapted by the corpus A model
100% of Reuters training data was used to train the model	94.6237	Hyper-parameters used were adapted by the corpus A model
70% of Reuters training data was used to train the model	97.684	Hyper-parameters used were adapted by the corpus A model
100% of Reuters training data was used to train the model	83.511	Hyper-parameters were set by corpus B dev dataset
Trained on Brown and tested on Reuters	266.02393	Hyper-parameters used were set by the corpus A model

Noteworthy observations:

- The performance seems to improve if we small fraction of corpus B's training data to adapt to the model trained on corpus A.
- The performance becomes even better as we increase the amount of corpus B's training data.
- This increase in performance is still not as good as using 100% Reuters training data and having the hyper-parameters be set by Reuter's dev dataset.