CSE 490u: Assignment 1

Zongzhen Chua
zzchua@cs.uw.edu
January 20, 2017

Question 1: Proof

The above model does not form a proper probability distribution. A properly distributed language model would satisfy the property:

$$\sum_i P(w_i|w_m, w_n) = 1$$

I shall disprove this using a counter example of a small corpus that violates the above property. Consider the following corpus of text:

$$\langle s\rangle\langle s\rangle \text{this is a cat, it is a dog}$$

Breaking this corpus down into the relevant unigram, bi-gram and tri-gram counts:

Unigram	count	MLE
this	1	0.1
is	2	0.2
a	2	0.2
cat	1	0.1
it	1	0.1
dog	1	0.1
$\langle s\rangle$	2	0.2

Bi-gram	count	MLE
$\langle s\rangle\langle s\rangle$	1	0.5
$\langle s\rangle$ this	1	0.5
this is	1	1
is a	2	1
a cat	1	0.5
cat it	1	1
it is	1	1
a dog	1	0.5

tri-gram	count	MLE
$\langle s\rangle\langle s\rangle$ this	1	1
$\langle s\rangle$ this is	1	1
this is a	1	1
is a cat	1	0.5
a cat it	1	1
cat it is	1	1
it is a	1	1
is a dog	1	0.5

To prove that the distribution does not sum to 1, let us consider w_{i-2} = is and w_{i-1} = a:

$$P(\text{cat}|\text{is,a}) = p_1(\text{cat}|\text{is,a}) = 0.5$$

$$P(\text{dog}|\text{is,a}) = p_1(\text{dog}|\text{is,a}) = 0.5$$

$$P(\text{it}|\text{is,a}) = p_3(\text{it}|\text{is,a}) = \frac{0.1}{0.1 + 0.2 + 0.2 + 0.1 + 0.2} = 0.0125$$

Adding up these probabilities:

$$\sum_i P(w_i|\text{is, a}) = 0.5 + 0.5 + 0.0125 + \ldots$$

$$= 1 + 0.0125 + \cdots > 1 \text{ since } P(w_i|\text{is, a}) >= 0$$

So we have shown that the sum of probabilities do not add up to 1.

Modifying the Backoff model

We can modify the Backoff model to appropriately discount the probabilities at the higher N-grams to distribute to the lower N-grams. This is also known as the Katz Backoff Model. We thus need to modify the Backoff model to factor in the discount factor and α_1 and α_2 which will distribute the probability mass to the lower order functions:

$$P_{bo}(w_i|w_{i-2}w_{i-1}) = \begin{cases} \hat{p_1}(w_i|w_{i-2}w_{i-1}), & \text{if } w_i \in \mathcal{A}(w_{i-2},w_{i-1}) \\ \alpha_1\hat{p_2}(w_i|w_{i-2}w_{i-1}), & \text{if } w_i \in \mathcal{A}(w_{i-2},w_{i-1}) \text{ and } w_i \in \mathcal{B}(w_{i-2},w_{i-1}) \\ \alpha_2\hat{p_3}(w_i|w_{i-2}w_{i-1}), & \text{if } w_i \in \mathcal{B}(w_{i-1}) \end{cases}$$

$\hat{p_1}$, $\hat{p_2}$, $\hat{p_3}$ are the probability functions with the ML estimate discounted to save some probability mass for lower order NGrams.

$$\hat{p_1} = dp_1, \hat{p_2} = dp_2, \hat{p_3} = dp_3 \text{ where } d \text{ is the discount rate from smoothing .}$$

$$d = \frac{c^*}{c} \text{ where } c^* = (c_{w_i} + k)\frac{N}{N+V}$$

d can be assigned by determining k. We can try and find the best d hyper-parameter through tuning. However note that $0 < d < 1$.

Next, to find α_1, we find the left over probability mass for the tri-gram probability function:

$$\beta(w_{i-2}w_{i-1})_1 = 1 - \sum_{w_i \in \mathcal{A}(w_{i-2},w_{i-1})} \hat{p_1}$$

We then distribute the left-over mass β_1 to p_2. Because p_2 is already a function over the sum of its probabilities, we can simply multiply β_1 to p_2 to distribute the left-over probability mass evenly on p_2. Therefore it follows that:

$$\alpha_1 = \beta_1$$

Similarly for α_2:

$$\beta(w_{i-1}) = 1 - \sum_{w_i \in \mathcal{A}(w_{i-1})} \hat{p_2}$$

$$\alpha_2 = \beta_2$$

This modification works because we have applied smoothing and this allows us to shave off probability mass from the tri-gram and bi-grams. d, the discount rate, represents this discount. α_1 and α_2 allows this probability mass that we have saved, to be distributed to the bi-gram and uni-grams respectively. By doing so, the language model probability will be properly distributed.

Experiment

Handling Unknowns
For both language models implemented, unknown words seen were given a frequency of 1. The models then adjust for this by considering a total word size of $N + 1$ in calculating the MLEs.

Backoff Language Model
The Backoff language model is implemented based off the modified Backoff model in the previous

page. The only hyper-parameter that requires selection is d - the discount factor. I discuss below how I tuned this hyper parameter.

Tuning Hyperparameter d

d was tuned by testing d from a range of 0.1 to 0.9 in increments of 0.1 using a development set (10%) of their respective training corpus. d was chosen based on which value of d resulted in the lowest perplexity score for the given language model. In figure 1, we can observe that for all 3 corpora, $d = 0.7$ gives the best performance. Any higher or lower will result in higher perplexity scores. This could imply that a large amount of probability mass needs to be transferred to the bigram and unigram models indicating that the model frequently backed off to the bigram and unigram models for all 3 corpora which it trained on.

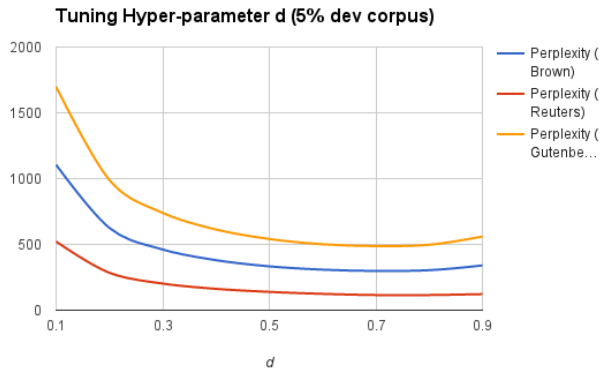

Figure 1: Chart showing performance of the backoff model with various values of d

Interpolation Model

The interpolation model implemented uses the following probability function:

$$P_{int}(w_i|w_{i-2}w_{i-1}) = \lambda_1 p_1(w_i|w_{i-2}w_{i-1}) + \lambda_2 p_2(w_i|w_{i-1}) + \lambda 3 p_2(w_i|w_{i-1})$$

where p_1, p_2 and p_3 are the MLE for trigrams, bigrams and unigrams respectively. In order to ensure a proper probability distribution, $\lambda_1 + \lambda_2 + \lambda_3 = 1$

Tuning Lambda

λs were chosen by varying between different combinations of λ_1, λ_2, λ_3 and testing it on a development set (10%) of their respective training corpus. The results are shown in figure 2. As observed in figure 2, the results are fairly consistent, with Brown and Reuters corpora performing better on $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $\lambda_3 = 0.4$. Gutenberg on the other hand performed slightly better on $\lambda_1 = 0.1$, $\lambda_2 = 0.3$, $\lambda_3 = 0.6$. What we can infer is that Brown and Reuters corpora have relatively stronger bigram data compared to unigram data. While Gutenberg corpora has weaker bigram data relative to its unigram data. All 3 corpora weighted their trigram data much lower suggesting that the trigram model frequently turned out 0 probability values due to insufficient 3-grams compared to 2-grams and 1-grams. Thus the model uses $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $\lambda_3 = 0.4$ as it resulted in the best performance across all 3 corpora.

Comparing Perplexities of each Language Model

In order to compare perplexities across various language models and corpora, we fix $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $\lambda_3 = 0.4$ and $d = 0.7$ since these hyper-parameters gave the most consistent performance the language models. Also the corpus was normalized by removing all punctuation such as ?!," and capital letters.

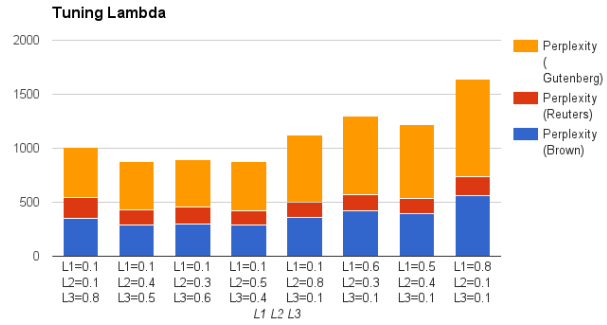

Figure 2: Chart showing performance of the interpolation model with various combinations of λ

As seen in Figure 3 and 4, the Brown and Reuters corpora gave the best perplexity scores using their respective test sentences. This is not surprising as the test sentences for each corpora probably have a close similarity in the choice of words and written tone as their training set, resulting in higher probability scores. However, in Figure 5, we can observe that Gutenberg test sentences did not give the best perplexity scores when trained on its own corpus. On the other hand, the Brown test sentences gave the best score. This could be due to Gutenberg's corpus consisting of multiple books written by multiple authors. So the language and tone across the corpus is not as consistent. Brown on the other hand could have reflected a more general distribution of words used across the various language tones across the Gutenberg corpus. Reuters performed the worst, which could be due to the dissimilarity between language used in novels compared to journalistic language.

One of the reasons why across all 3 corpora used to train the models, when the test sentence does not originate from the same training corpora, the perplexity scores become very high is that between the corpora, the vocabulary changes alot resulting in many unknown words during test time. As such the models will generate a very unfavourable probability for these unknowns, resulting in a high perplexity when a model is tested against a test sentences not from the same corpora as the training set.

Another observation is that across all the corpora used, the interpolation models performed consistently better than their respective backoff models - in the figures, red tends to be lower than blue. It seems that this could be due to the fact that the backoff model applies a significant discount to both trigrams and bigrams resulting in a lower probability value. On the other hand, the interpolation method aggregates across all 3 trigrams, bigrams and unigrams as opposed choosing one and applying a significant discount as in the backoff model.

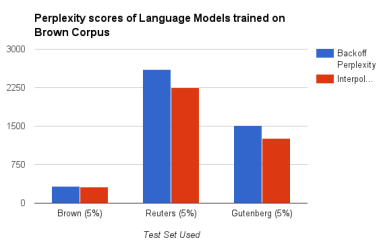

Figure 3: Chart showing performance of both language models trained on Brown corpus

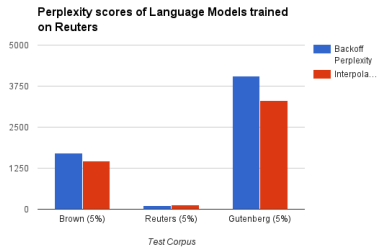

Figure 4: Chart showing performance of both language models trained on Reuters corpus

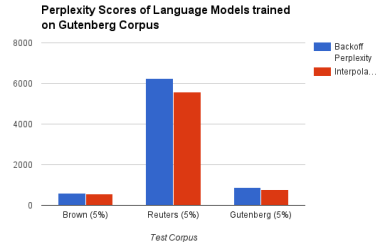

Figure 5: Chart showing performance of both language models trained on Gutenberg corpus