Natural Language Understanding
Assignment - 1

Saley Vishal Vivek
vishalsaley@iisc.ac.in

Abstract

Assignment consists of two tasks.

1. To train and evaluate multiple Language Models on given copora. Evaluation has to be done on four different combinations of train and test sets (S1 to S4).
2. Select the best model after evaluation to generate tokens.

1 Preprocessing

The corpora includes two different corpus: **gutenberg** and **brown** . Each corpus consist of multiple text files which include the natural text. Following preprocessing tasks are performed on each of the corpus:

1. Read each file sentence by sentence.
2. For each sentence, add start and end of sentence markers appropriately based on the model that is to be used.
3. Divide this list of sentences into three sets: *train*, *test* and *devset*.
4. Store these sets as individual files.

Apart from this, a unified train file is generated which contains sentences from both the gutenberg and brown corpus.

2 Task 1: Language Models

Unigram, **Bigram** and **Trigram** models are trained on the corpora generated after preprocessing. For this, all train files are read and all required counts are generate and stored. These stored counts are then used for calculation perplexity over train and devsets. Models are implemented considering open vocabulary. Thus new words are explicitly handled. Also, to handle zeros two different smoothing techniques, Katz's Back-off and Kneser-Ney are incorporated in the model.

2.1 Unknown Handling

To handle 'out of' vocabulary words, discounting is used at unigram level. For every word in vocabulary, *0.5* is subtracted from actual unigram count. This discounted count is then assigned to a new special word 'UNK'. Thus unigram probabilities become

$$P(w_i) = \frac{C(w_i) - 0.5}{\sum_{w_j \in vocab} C(w_j)}$$

$$P(unknown) = \frac{|vocab| * 0.5}{\sum_{w_j \in vocab} C(w_j)}$$

This has some implications while bigrams and trigrams calculations which are appropriately handled. In case of Kneser-Ney, similar discounting is used while calculating continuation probability of unknown.

2.2 Smoothings

2.2.1 Katz's Back-off

Following formulation of Katz's Back-off is used in bigram model:

$$\begin{aligned}\hat{P}(u|v) &= \frac{C(vu) - \beta}{C(v)} \quad if \quad C(vu) > 0 \\ &= \alpha_v \hat{P}(u) \qquad if \quad C(vu) = 0\end{aligned}$$

where C(*vu*) = Bigram count of *vu*, C(*v*) = Unigram count of *v* and β is discount parameter. α_v is overall discount. β value is tuned over *devset*. The result for case S1 is shown in figure1. As discount is increased, perplexity over *devset* increases. Thus the minimum value of 0.1 is fixed for bigram discount.
Similar formulation is used for smoothing in trigram models. For trigram, we are fixing bigram

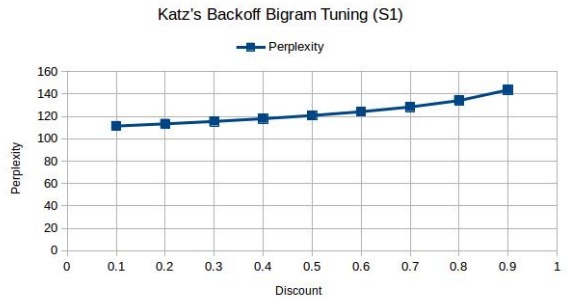

Figure 1:

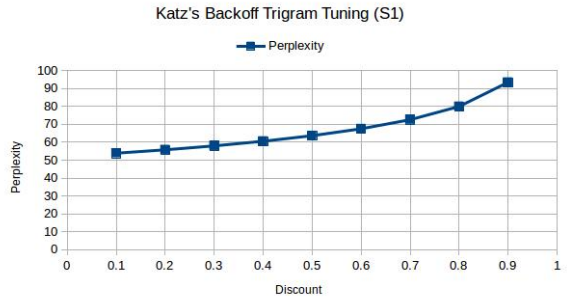

Figure 2:

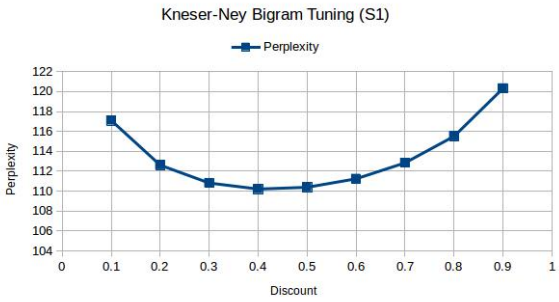

Figure 3:

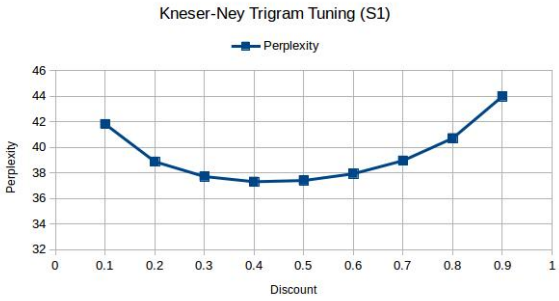

Figure 4:

discount to 0.1. Behaviour similar to bigram is observed as shown in figure 2. The minimum perplexity is over *devset* observed at discount of 0.1.

2.2.2 Kneser-Ney smoothing

For Kneser-Ney, unigram probabilities are calculated considering novel continuation by the word.

$$P_c(w_i) = \frac{|\{v|C(vw_i) > 0\}|}{|\{(v,w)|C(vw) > 0\}|}$$

and bigram probability is

$$P(u|v) = \frac{max(C(vu) - \beta)}{C(v)} + \lambda_v * P_c(u)$$

Discounting of 0.5 is applied to unigrams to handle unknowns as mentioned above. Discount parameter β is tuned over *devset* and behaviour as shown in figure 3 is observed. The minimum perplexity on *devset* is observed at 0.4 discount.

For trigram, instead of using true Kneser-Ney, a hybrid between Kneser-Ney and interpolation is used i.e. above bigram implementation is directly used instead of calculating continuation probabilities of bigrams. The results are as shown in figure 4. The minimum perplexity on *devset* is observed at 0.4 discount.

Similar trends are observed in case of S2, S3 and S4 as well. Overall preplexity results over *test* after tuning parameters in both Katz and Kneser-Ney are shown in table 1 and 2.

Table 1: Bigram Comparison

Model	S1	S2	S3	S4
Katz's Back-off	112.62	148.42	122.33	201.40
Kneser-Ney	111.01	145.15	121.58	195.07

Hyperparamerters:

- Katz's Back-off: bigram_discount = 0.1 and trigram_discount = 0.1
- Kneser-Ney: bigram_discount = 0.4 and trigram_discount = 0.4

2.3 Analysis

From above evaluation, both Katz's Back-off and Kneser-Ney trigrams perform better on given corpora. We don't see much gain in case of Kneser-Ney with respect to Katz's Back-off for due to hybrid implementation of Kneser-Ney trigram. The sentence generated by hybrid Kneser-Ney lose the

Table 2: Trigram Comparison

Model	S1	S2	S3	S4
Katz's Back-off	39.00	43.00	41.71	55.37
Kneser-Ney	38.24	41.08	41.08	53.30

advantage due to hybrid implementation. Thus Katz's is used for sentence generation.

- Katz's sentence: I wish you were looking for me , is somehow
- Kneser-Ney sentence: Cruelty In Open how Twenty I But The The And