Smoothing Techniques and Visualisation

Professor:
Sutanu Chakraborti

Sidharth Aggarwal (CS17S012)
Amar Vashishth (CS17M052)

Contents

Chapter 1

INTRODUCTION

1.1 Introduction

In this project we have dealt with the smoothing techniques. As in NLP at most the time we use probabalistic techniques for solving the certain tasks of the NLP. Some of the tasks are spell check, vectorisation, language models, machine translation, etc. So in each and every task at some point of time we need to calculate the probability of the n-gram for solving the task. So what wrong happened while solving the task that we have to use the smoothing techniques.

1.2 Language Modelling

A statistical language model is a probability distribution over sequences of words. Given such a sequence, say of length m, it assigns a probability $P(w_1, \ldots, w_m)$ to the whole sequence. Having a way to estimate the relative likelihood of different phrases is useful in many natural language processing applications, especially ones that generate text as an output. Language modeling is used in speech recognition, machine translation, part-of-speech tagging, parsing, handwriting recognition, information retrieval and other applications.

Data sparsity is a major problem in building language models. Most possible word sequences will not be observed in training. One solution is to make the assumption that the probability of a word only depends on the previous n words. This is known as an n-gram model or unigram model when n = 1.

$$P(w_1, w_2, \ldots, w_n) = \Pi P(w_i | w_{i-k}, \ldots, w_{i-1})$$

By using Markov assumption that the words doesnot depend on all the words so, we can simply the probability as:-

- **Unigram**

  For unigram we just use the word itself to calculate the probability which means that all the words are independent to each other.

$$P(w_1, w_2, \ldots, w_n) = \Pi P(w_i)$$

- **Bigram**
  For Bigram we use two words to calculate the probability of the factor of the language model which means that all the words are dependent of just the previous one word.

$$P(w_1, w_2, \ldots, w_n) = \Pi P(w_i|w_{i-1})$$

- **Trigram**
  For Trigram we use three words to calculate the probability of the factor of the language model which means that all the words are dependent of just the previous two word.

$$P(w_1, w_2, \ldots, w_n) = \Pi P(w_i|w_{i-1}, w_{i-2})$$

1.3 Need of Smoothing Techniques

As we have explained above in the introduction that the task involve probability calculation, and further we know that the probability of the n-gram is the count of joint n-gram divided by the count of the given (n-1) grams. That is,

$$P(w_i|w_{i-1}^{i-1-n}) = \frac{count(w_{i-1}^{i-1-n}, w_i)}{count(w_{i-1}^{i-1-n})}$$

So, as you can see above the equation which is having the count of the grams. And the case may arise the count of the grams are occuring to be zero. So in that case the probability of that gram given the other will be zero. But instead we would like to assign some probability to that the gram even if it is zero. It will smoothen the calculation process. As if want to calculate the probability of the sentence and any of the bigram program probability comes out to be zero then the whole sentence probability will be zero. So to avoid these issues we use smoothing techniques.

Chapter 2

SMOOTHING TECHNIQUES

The idea behind the smoothing techniques is that to prevent from the zero probabilities we discount some mass from the n-gram we have seen to the n-grams which we have not seen in the corpus.

2.1 Add-1 Smoothing

This is the type of smoothing technique in which we will add one in the numerator and further add the count of the vocabulary in the denominator to smoothen the probability. Means to assign some probability to the grams whose count are coming out to be zero. The formula for the Add-1 smoothing is give below as,

$$P(w_i|w_{i-1}^{i-1-n}) = \frac{count(w_{i-1}^{i-1-n}, w_i) + 1}{count(w_{i-1}^{i-1-n}) + |V|}$$

where,

- $count(w_{i-1}^{i-1-n}, w_i)$ and $count(w_{i-1}^{i-1-n}) - >$ Number of times the sequence of words occur
- $|V| - >$ The count of the vocabulary or vocabulary size

Now to illustrate the technique we will give an toy example,

2.1.1 Example

Let the toy corpus is as given,

$< s >$ JOHN READ MOBY DICK $< e >$
$< s >$ MARY READ A DIFFERENT BOOK $< e >$
$< s >$ SHE READ A BOOK BY CHER $< e >$

Further if we take some sentences to calculate the probability such as,

p(CHER READ A BOOK)

and it can be written as if we take a bigram model,

$$= p(CHER| < s >)p(READ|CHER)p(A|READ)p(BOOK|A)p(< e > |BOOK)$$

As as we know the formula for the probability of a bigram without smoothing, will put that,

$$= \frac{c(CHER| < s >)}{c(< s >)} \frac{c(READ|CHER)}{c(CHER)} \frac{c(A|READ)}{c(READ)} \frac{c(BOOK|A)}{c(A)} \frac{c(< e > |BOOK)}{c(BOOK)}$$

$$= \frac{0}{3} \frac{0}{1} \frac{2}{3} \frac{1}{2} \frac{1}{2}$$

So the probability,

p(CHER READ A BOOK) = 0

But if we use Add-1 smoothing then this situation can be avoided,

$$= \frac{c(CHER| < s >) + 1}{c(< s >) + |V|} \frac{c(READ|CHER) + 1}{c(CHER) + |V|} \frac{c(A|READ) + 1}{c(READ) + |V|} \frac{c(BOOK|A) + 1}{c(A) + |V|} \frac{c(< e > |BOOK) + 1}{c(BOOK) + |V|}$$

And in this case the $|V| = 11$,

$$= \left[\frac{0+1}{3+11}\right]\left[\frac{0+1}{1+11}\right]\left[\frac{2+1}{3+11}\right]\left[\frac{1+1}{2+11}\right]\left[\frac{1+1}{2+11}\right]$$

So, the final probability will not be zero in this case,

p(CHER READ A BOOK) ≈ 0.00003

So, from the above example as we saw that with smoothing we have given assigned some probability to the bigram which does not exist in the corpus

2.2 Laplace Smoothing

This smoothing technique operates in the same fashion as the Add-1 smoothing but in Laplace we add k in the numerator and further add product of k and vocabulary size in the denominator to smoothen the probability. Means to assign some probability to the grams whose count are coming out to be zero. The formula for the Add-1 smoothing is give below as,

$$P(w_i|w_{i-1}^{i-1-n}) = \frac{count(w_{i-1}^{i-1-n}, w_i) + k}{count(w_{i-1}^{i-1-n}) + k * |V|}$$

where,

- $count(w_{i-1}^{i-1-n}, w_i)$ and $count(w_{i-1}^{i-1-n}) -> $ Number of times the sequence of words occur
- $|V| -> $ The count of the vocabulary or vocabulary size
- k is the parameter which can be learned empirically

2.3 Good Turing Smoothing

It is a type of smoothing technique which is used to find the probability of the uigram which has already occured c times in the corpus. So in this there are two types of cases on the basis of number of occurence of the word in the corpus.
Here, c is the number of times the word has occured.

2.3.1 Two Cases

- c=0:
  In this the formula for the probability is as,

  $$P(entitywithZerofreq.) = \frac{N_1}{N}$$

  where,

  - N_1 : Total number of entities with count as zero
  - N : Total number of words in the corpus

- c > 0:
  In this the formula is different from the previous part. For this we calculate a new value of the c called as c^* and the formula is as,

  $$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

  where,
  N_c : It is the total number of entities with count c
  Further for the probability,

  $$P(entitywithfreq - c) = \frac{c^*}{N}$$

2.3.2 Example

To explain the above we take an example,

- You were fishing and caught, 10 carp, 3 perch, 2 whitefish, 1 trout, 1 salmon, 1 eel, 0 catfish and 0 bass. In total you caught 18 fishes.
  So, N = 18
- If we want to find the P(catfish) which has a freq. as 0 so the formula will be as,
  N_1 = 3 (trout, salmon, eel has count of 1)
  N = 18, so,
  $$P(catfish) = \frac{3}{18}$$
- Now if we want to find the P(trout) which has non-zero freq., so now the formulation will change.
  c = 1 (As, trout has count 1)
  N_2 = 1 (As only whitefish has count of 2)
  N_1 = 3 (same as above)
  $$c* = \frac{(1+1)*1}{3} = \frac{2}{3}$$
  Further,
  $$P(trout) = \frac{\frac{2}{3}}{18} = \frac{1}{27}$$

So, this is the way the Good Turing apply smoothing in the probabilities.

2.4 Jelinek-Mercer smoothing (interpolation)

It is the type of smoothing technique which takes into account the probabilities all the lower grams also while finding the probability of the n-gram. If e.g, we want to find the probability of a bigram then we will also take into account the probability of the unigram of that word in some proportion. The formulation is as give,

- Bigram
  $$P(w_i|w_{i-1}) = \lambda(P(w_i|w_{i-1}) + (1-\lambda)P(w_i)$$
- Trigram
  $$P(w_i|w_{i-1}, w_{i-2}) = \lambda_1 P(w_i|w_{i-1}, w_{i-2}) + \lambda_2(P(w_i|w_{i-1}) + \lambda_3 P(w_i)$$
  where, $\lambda_1 + \lambda_2 + \lambda_3 = 1$

So, for more clarification we will take an example,

2.4.1 Example

Let we want to find the probability of the word given a word from the above corpus,

$$P(CHER|READ) = \lambda P(CHER|READ) + (1 - \lambda) P(CHER)$$

now, here the $P(CHER|READ) = 0$ and $P(CHER) = \frac{1}{11}$
and we assume value of $\lambda = 0.6$, so the P(CHER—READ) is as,

$$P(CHER|READ) = 0.6 * 0 + 0.4 * \frac{1}{11} = 0.036$$

Through smoothing we have 0.036 probability rather than zero.

2.5 Katz Smoothing

In this technique of smoothing we will deduct some value from the n-gram with non-zero count. And then distribute the discounted value among the bigrams with count as zero.

2.5.1 Katz Back-Off Model(Bigram)

- For a bigram model, define two sets

$$A(w_{i-1}) = [w : Count(w_{i-1}, w) > 0]$$
$$B(w_{i-1}) = [w : Count(w_{i-1}, w) = 0]$$

- A Bigram Model

$$P(w_i|w_{i-1}) = \left\{ \begin{array}{ll} \frac{Count*(w_{i-1},w_i)}{Count(w_{i-1}}, & If w_i \in A(w_{i-1}) \\ \alpha(w_{i-1}) \frac{P(w_i)}{\sum_{w \in B(w_{i-1})} P(w)}, & If w_i \in B(w_{i-1}) \end{array} \right\}$$

where,

–

$$count* = count - discount$$

–

$$\alpha(w_{i-1}) = 1 - \sum_{w \in A(w_{i-1})} \frac{Count * (w_{i-1}, w_i)}{Count(w_{i-1}}$$

To clarify all the points we will take an example as below,

2.5.2 Example

Now let us take,
Count*(x) = Count(x) - 0.5
And in this example we have taken the discount = 0.5, which can be set accordingly.
We have plotted a table below to show some stats.

x	Count(x)	Count*(x)	$\frac{Count*(x)}{Count(the)}$
the	48		
the,dog	15	14.5	$\frac{14.5}{48}$
the,woman	11	10.5	$\frac{10.5}{48}$
the,man	10	9.5	$\frac{9.5}{48}$
the,park	5	4.5	$\frac{4.5}{48}$
the,job	2	1.5	$\frac{1.5}{48}$
the,telephone	1	0.5	$\frac{0.5}{48}$
the,manual	1	0.5	$\frac{0.5}{48}$
the,afternoon	1	0.5	$\frac{0.5}{48}$
the,country	1	0.5	$\frac{0.5}{48}$
the,street	1	0.5	$\frac{0.5}{48}$

So, for above the α will be as,

$$\alpha(the) = 10 * \frac{0.5}{48} = \frac{5}{48}$$

Further using the above value we can smoothened probabilities to the different n-grams.

2.6 Witten-Bell

- It is the smoothing technique which thinks of the unseen events as ones not having happened yet.
- The formula for this smoothing is given as,
  T(w_x) = number of bigrams starting with w_x
  - **Zero-count events:**
    Total probability mass

$$\sum_{i:C(w_{i-1}w_i)=0} p^*(w_i|w_{i-1}) = \frac{T(w_{i-1}}{N + T(w_{i-1}}$$

where,

$$p^*(w_i|w_{i-1}) = \frac{T(w_{i-1}}{Z(w_{i-1})(N + T(w_{i-1})} \qquad if, C(w_{i-1}w_i) = 0$$

- **Non-zero-count events**

$$p^*(w_i|w_{i-1}) = \frac{C(w_{i-1}w_i}{(N + T(w_{i-1})} \qquad if, C(w_{i-1}w_i) > 0$$

where,

$$N = C(w_{i-1})$$

2.7 Absolute Discounting Interpolation

It is also a smoothing technique in which we combine the discounting and interpolation techniques. The formula for this smoothing technique is as,

$$P_{AD}(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i) - d}{c(w_{i-1})} + \lambda(w_{i-1})P(w)$$

where,

- $\lambda(w_{i-1})$ is the interpolation weight
- P(w) is the unigram

Chapter 3

Experimentations and Analysis

Few experiment are done on different parameter are as,
Corpus:- Brown
Perplexity of Unsmoothed unigram :- 1025.33

- **Add-K-Smoothing**
  Table of experiments on the Add-k-smoothing on the data:-

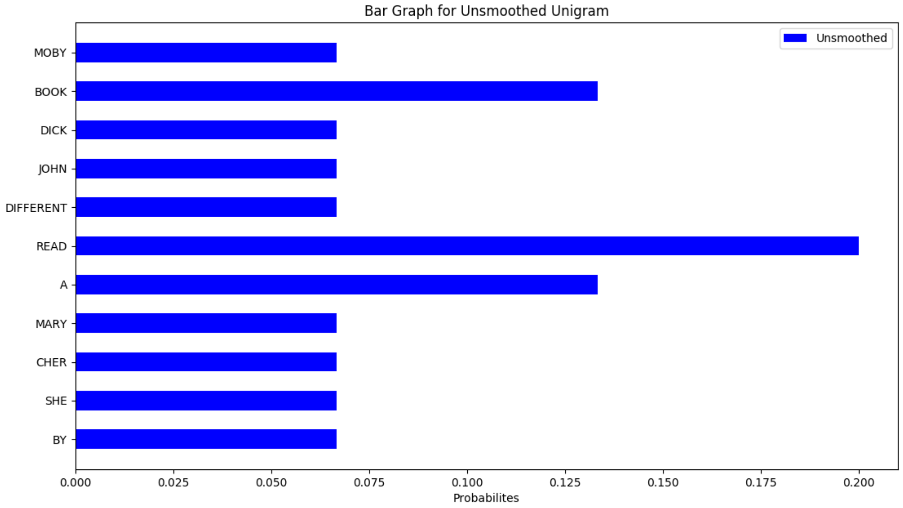
k	PP(Unigram)	PP(Bigram)	PP(Trigram)
1e-07	1025.33	.200	2624
1e-06	873006885.82	0.485	2626.7
1e-05	86055331516.62	2.19	2645.57
0.0001	8632263042	9.911	2775.55
0.01	86669298	202.26	4510.233
0.1	8705963.71	900.51	6871.30
1	895168.36	3633.24	10337.177

- **Linear Interpolation**
  Table for the linear interpolation with trigram and different values of the lambdas.

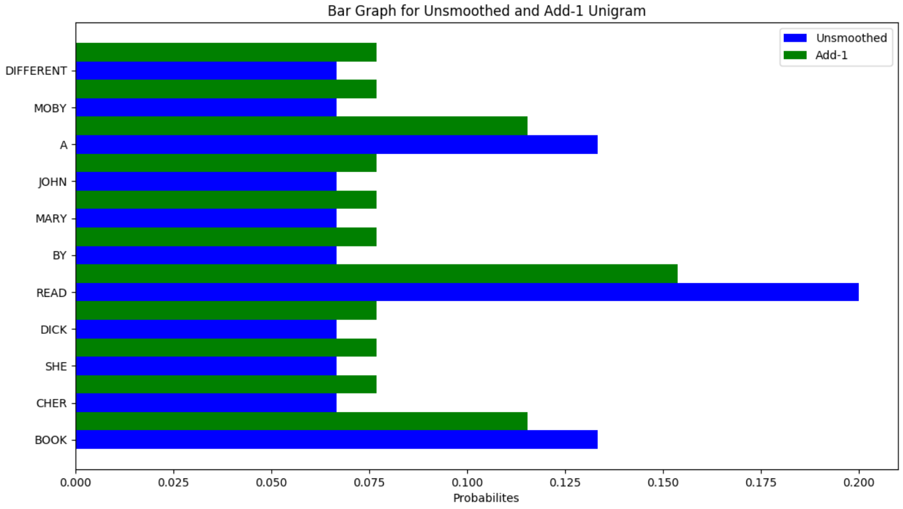
lambda1	lambda2	lambda3	Perpexity
0.001	0.009	0.99	448.91
0.3	0.3	0.4	280.07
0.6	0.3	0.1	442.57
0.99	0.009	0.001	7856.488

- **Plots**

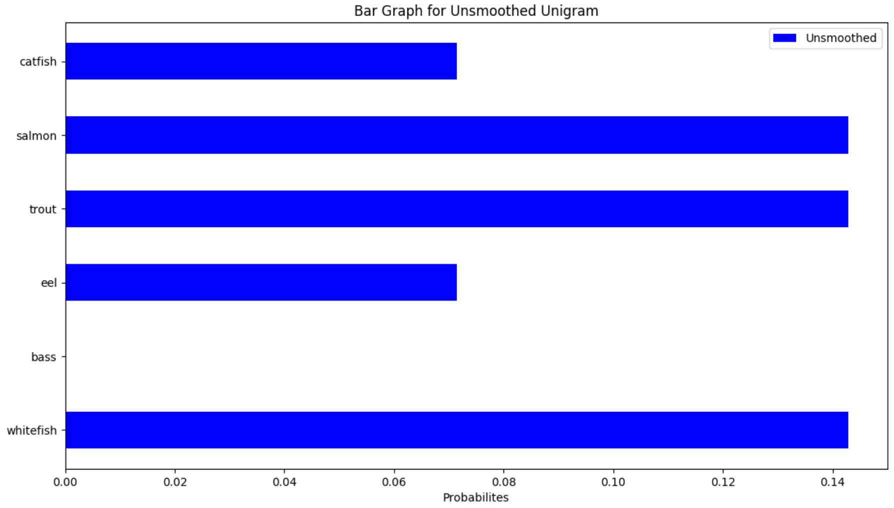
  - **UnSmoothed Unigram Plot**

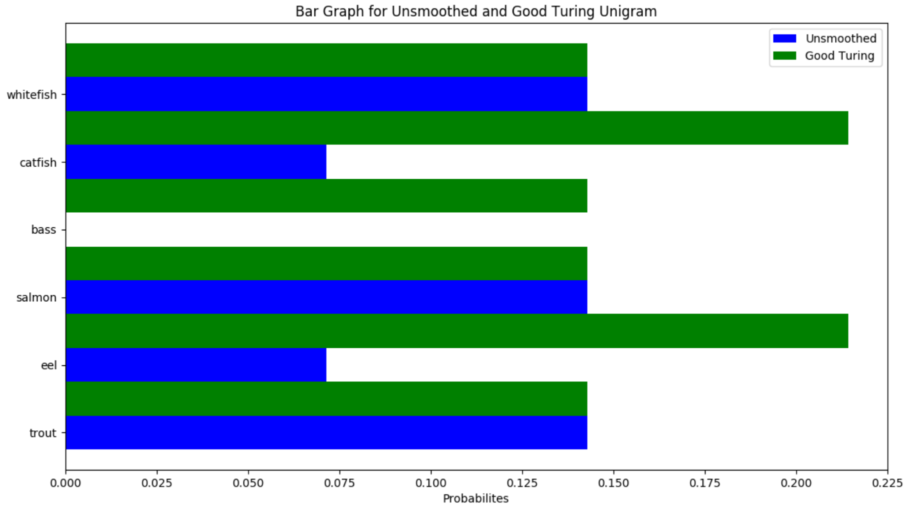


  - **Add-1 Smoothing Unigram Plot**

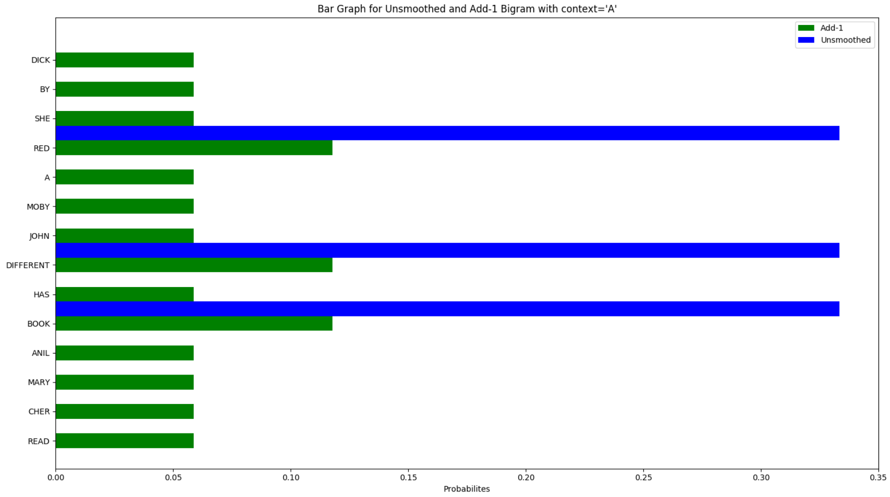


Good Turing Smoothing Unigram Plots

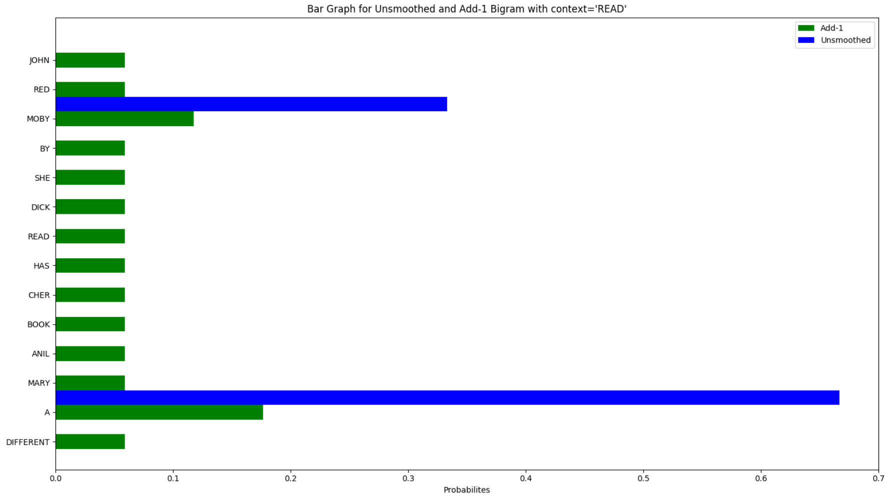




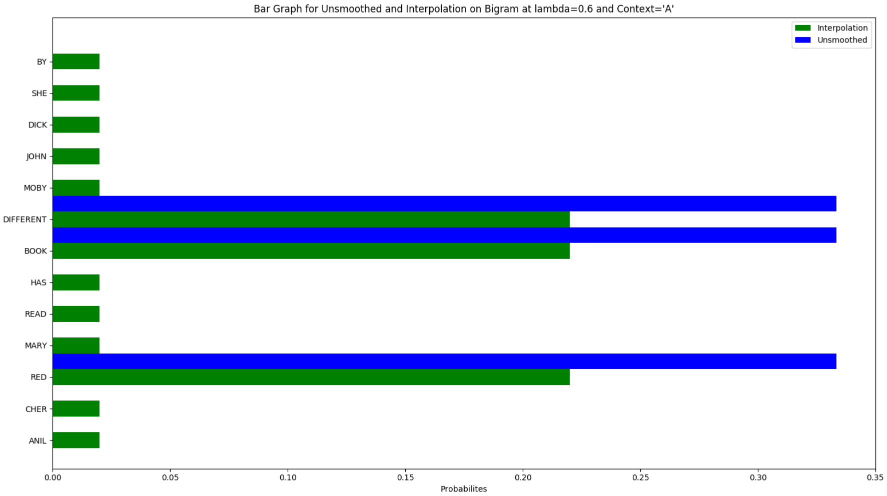
- **Add-1 Smoothing on Bigram for** $P(w|A)$



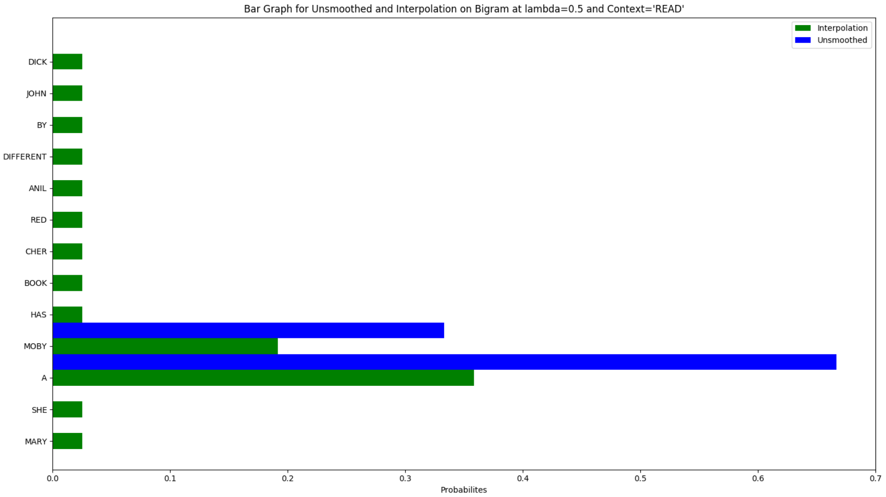
- **Add-1 Smoothing on Bigram for** $P(w|READ)$



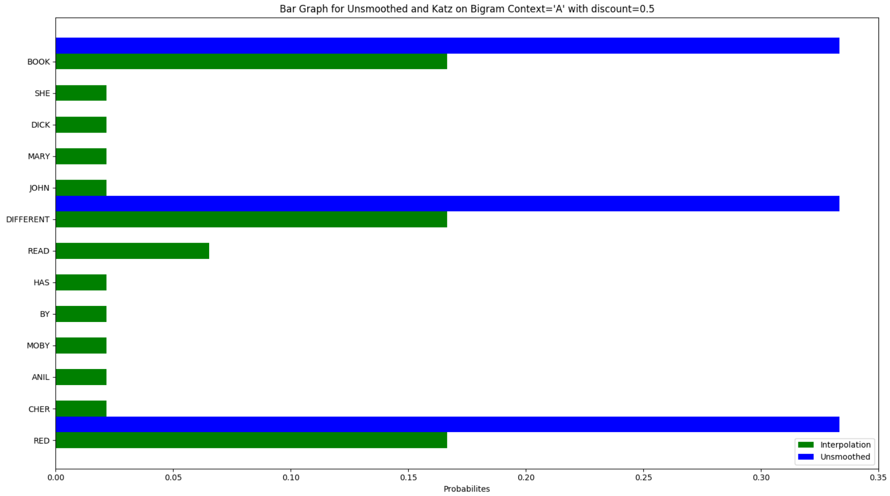
- ## Interpolation Smoothing on Bigram for $P(w|A)$



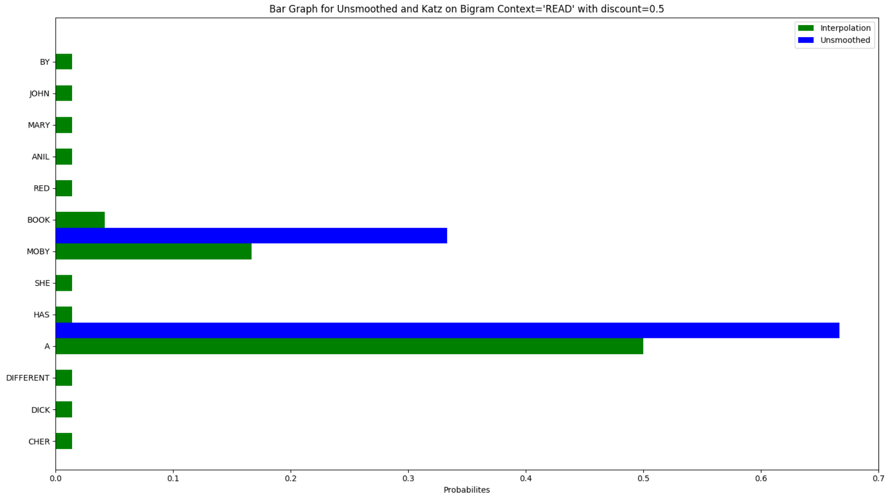
- ## Interpolation Smoothing on Bigram for $P(w|READ)$

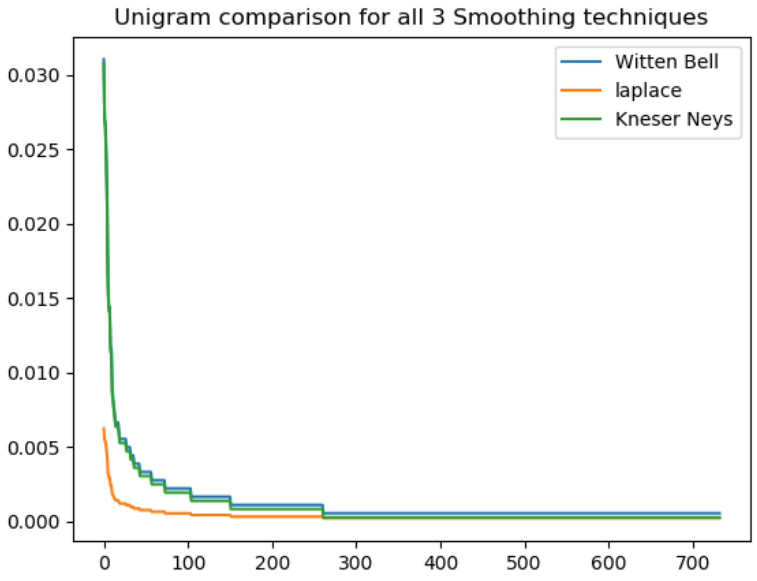


– Katz Smoothing on Bigram for $P(w|A)$



– Katz Smoothing on Bigram for $P(w|READ)$




Unigram comparison for all 3 Smoothing techniques
Witten Bell
laplace
Kneser Neys
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
100
200
300
400
500
600
700

Chapter 4

Visualization and Evaluation

4.1 Extinsic Evalution

- In the extinsic evalutation we can put the model into some particular task and then see the accuracy of the model. For example if the sentence is having a wrong spelling, so we can do a spell check of the word and after that we can find the probability of the sentences with different words.

- It is an effective technique of evaluation but the process of evaluation takes a lots of times. So in that case we may switch to the Intrinsic Evaluation techniques.

- **Example**
  Let the toy corpus is as given,

  $$< s >\text{ JOHN READ MOBY DICK }< e >$$
  $$< s >\text{ MARY READ A DIFFERENT BOOK }< e >$$
  $$< s >\text{ SHE READ A BOOK BY CHER }< e >$$
  $$< s >\text{ ANIL HAS A RED BOOK }< e >$$

  So, let us take an input sentence having a spelling mistake as,

  JOHN REED A BOOK

  So in this case we can see that the spelling of **READ** is misspelled as **REED**.
  So first of all we will find the suggestions for the misspelled word,

  Suggestions(REED) = READ, RED, etc

  So, now we will find the probability of all the sentences with the suggestions replaced with the misspelled word and in this case we are taking two suggestions.

- **READ as the Suggestion**

$$\text{P(JOHN READ A BOOK)} = 0.000054$$

- **RED as the Suggestion**

$$\text{P(JOHN RED A BOOK)} = 0.00001$$

Clearly from above we can see that the sentence with the suggestion as **READ** has more probability than the suggestion as **RED**.

4.2 Intinsic Evaluation

- **Perplexity**
  - Perplexity is the inverse of the probability of the test set, normalised by the number of words.
  - Perplexity is a weighted equivalent branching factor.
  - In the context of the distribution it quantify amount of surprise the language model has when given a new sentence or test instance.
  - Formula for the Perplexity is as,

$$PP(w) = \sqrt[N]{\frac{1}{P(w_1, w_2, \ldots, w_N})}$$

    where,
    - PP(w) is the Perplexity of the test sentence.
    - N is the total number of words in the sentence.
    - $P(w_1, w_2, \ldots, w_N)$ is the probability of the sentence.

- **Example**
  The perplexity of the above example in the extrinsic evalutation is as,
  - **READ as the Suggestion**

$$\text{PP(JOHN READ A BOOK)} = 1.847$$

  - **RED as the Suggestion**

$$\text{PP(JOHN RED A BOOK)} = 2.150$$

The above example clearly explains that the model was much more surprised when it received the sentence with **RED** as the suggestion rather than with the sentence with **READ** as the suggestion.

Chapter 5

Application

- **Machine Translation**
  The smoothing techniques can be used when we will work on the language models for the machine translation. To explain this we will give an example as,

  P(high winds tonite) > P(large winds tonite)

- **Speech Recognition**
  The smoothing techniques can be used when we will work on the language models for the speech recognition. To explain this we will give an example as,

  P(I saw a van) > P(eves awe of an)

- **Spell Check**
  The smoothing techniques can be used when we will work on the language models for the spell check. To explain this we will give an example as,

  P(john read a book) > P(john red a book)

Chapter 6

Conclusion and References

6.1 Conclusions

- Jelinek-Mercer performs better on small training sets; Katz performs better on large training sets.
- Interpolated models are superior to backoff models for low (nonzero) counts.
- Some experimental analysis is shown which gives some idea of value of k in the Add-k smoothing and lambda's in the Interpolation.
- Laplace smoothing does heavy discounting sometimes so we use other smoothing techniques.

6.2 References

- Class Notes
- Wikipedia