Ayush Goyal

2019MT10961

ELL 881 – Assignment 1

Report

N Gram Models:

Firstly, data cleaning is performed (steps involved are as follows):

1) Converted the whole set to lowercase.
2) Removed '\n' (end of line) character.
3) Removed punctuation (using string.punctuation) except '.' and ','.
4) Removed footers present in the data.
5) Removed more punctuation marks, while processing the data.

Created a Language Model:

Sentences generated for diff N-grams are as follows:

1) For n = 1:

attracting picture anuzzer handknitted blacker recoiled car releasing torso cupboards kreacher vileness magic eagles drooped tutus honey loon stayed amycus aggrieved bell yes cheaper irked massive clinking explode shapeless nibble intensive stormtossed matter

2) For n = 2:

only if i dreamed you hate maroon again there was sitting in them before him you he was caught sight he knew had added pacing from a new legislation

3) For n = 3:

neville s mother have died though said hannah abbott said ron now munching on his shoulder you re not being shouted at ron the whole thing is i think

4) For n = 4:

brutus s said uncle vernon forcefully why demanded ron seizing her schedule have you outlined all lockhart s books his expression was kindly rather than accusatory harry

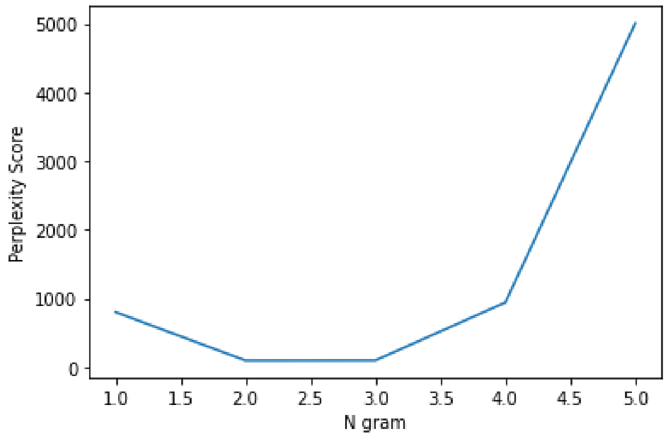
5) For n = 5:

and he was sending two ministry cars to take them all to the station tomorrow so that the weasleys could look after harry until he was on the verge of saying me but

Test Set – Book7.txt

Perplexity Score for different N-gram models

Scores obtained: [799.5095369240573, 95.1832754459799, 96.62116099882968, 939.456816727218, 5007.343013101615]

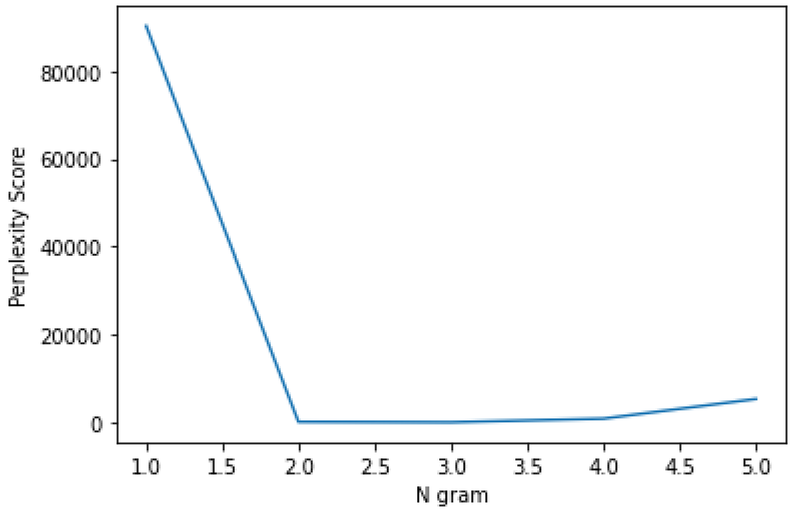


Observation: We can clearly see that on increasing n, we are overfitting on the training set and thus getting bad results (high perplexity) as seen in the above graph. So, in the further report I have kept my analysis for n = 1,2,3,4,5.

Smoothing Methods:

Good Turing:

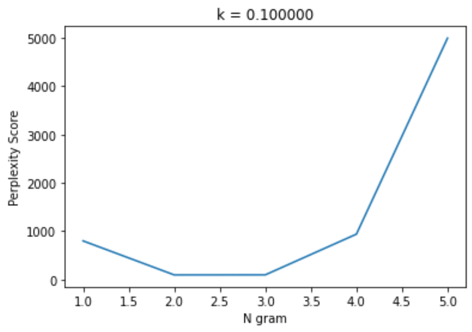
Scores obtained: [90222.33485551766, 177.81218182837222, 90.19026201277183, 911.619327491149, 5384.4177940124055]



Add – K :

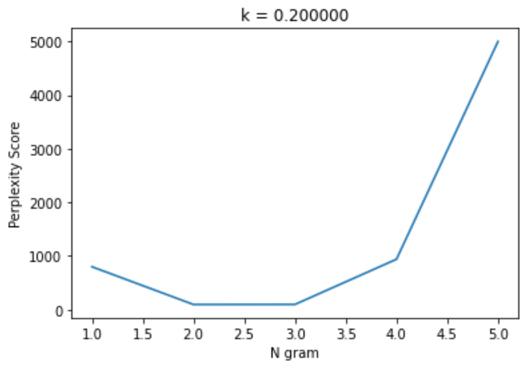
1) K = 0.1

Scores Obtained: [799.5095369240573, 95.21445417031505, 96.54660320926641, 939.1735916713009, 4998.520066943883]



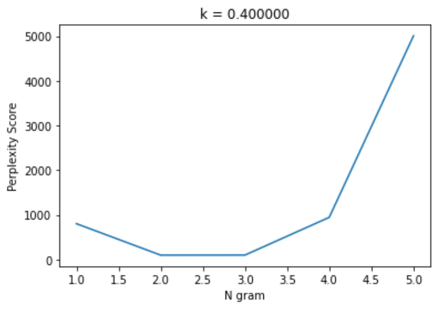
2) K = 0.2

Scores Obtained: [799.5095369240573, 95.18276750480216, 96.67034428553566, 939.5714216127049, 5002.017087995679]



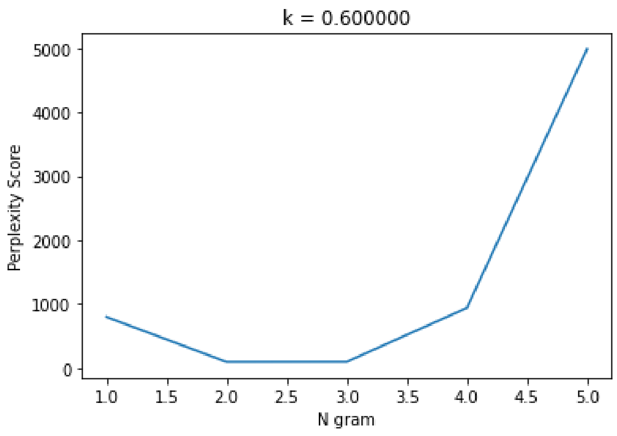
3) K = 0.4

Scores Obtained: [799.5095369240573, 95.18090189458428, 96.58771919023621, 941.3353639515043, 5006.919699560801]



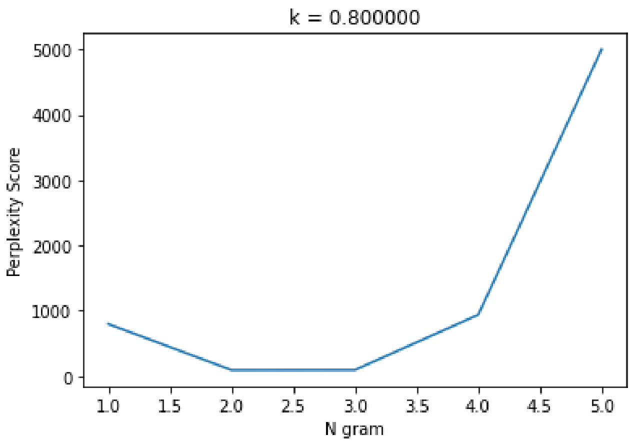
4) K = 0.6

Scores Obtained: [799.5095369240573, 95.11913081486108, 96.36256524776877, 940.0143097263095, 4990.673312565771]



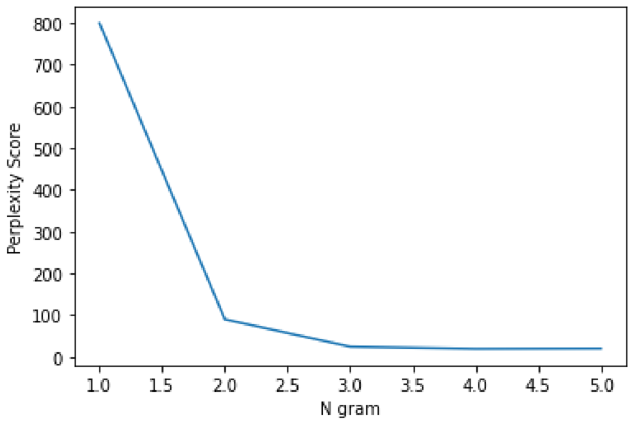
5) K = 0.8

Scores Obtained: [799.5095369240573, 95.20216329535486, 96.72801433923811, 939.7308845334893, 4999.899724568795]



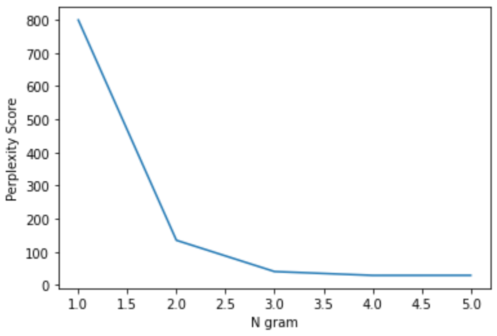
Backoff:

Scores obtained: [799.5095369240573, 90.03571469775375, 24.574652323376664, 19.399709292163738, 20.137741741832937]



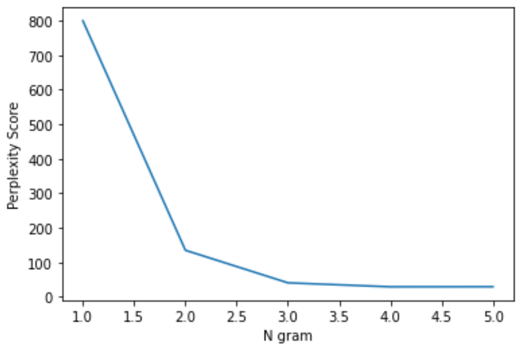
Interpolation:

Scores obtained: [799.5095369240573, 135.08773582137883, 40.74263705823612, 29.210160184568522, 29.361534783118625]



Kneser-Nay:

Scores Obtained: [800.6072349250573, 70.40571669772375, 14.694652323376664, 18.456709292163738, 25.317741741832937]



Comparing All the perplexity scores for different methods for n = 1,2,3,4,5.

n	Normal	Good Turing	Add-k = 0.1	Add-k = 0.2	Add-k = 0.4	Add-k = 0.6	Add-k = 0.8	Back-off	Interp-olation	Kneser - Nay
1	799.5	90222.3	799.5	799.5	799.5	799.5	799.5	799.5	799.5	800
2	95.1	177.8	95.2	95.1	95.1	95.1	95.2	90	135	70.4
3	96.6	90.1	96.5	96.6	96.5	96.3	96.7	24.5	40.7	14.6
4	939.4	911.6	939.1	939.5	941.3	940	939.7	19.3	29.2	18.4
5	5007.3	5384.4	4998.5	5002	5006.9	4990.6	4999.8	20.1	29.3	25.3

It could be seen from the above results that smoothing techniques are helpful and techniques like Backoff, Interpolation and Kneser-Nay are performing better I.e., giving less perplexity score.

The best model as per the results is: Kneser-Nay (in general for different Ngrams)