N-gram and Smoothing Techniques

ELL881 Assignment 1 Report

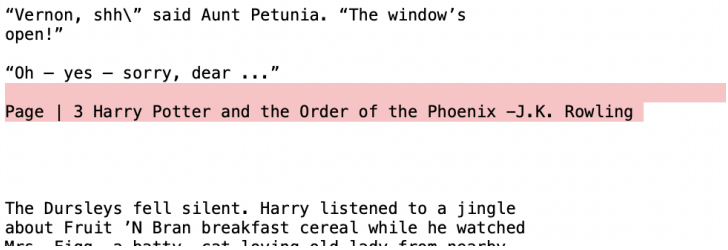
Vaibhav Seth
2021MT10236

Data Cleaning

To implement the N-gram models and to experiment with various smoothing techniques the raw data had to be converted to a useful form

The following conversion were applied:

- Page number and book titles were removed (Fig.1)



(Fig.1)

- Punctuations were removed except period (‘.’)
- All text was converted to lower case
- All numbers and other special characters were removed
- Some page descriptions had to be removed manually due to inconsistent format

Dataset

- All the books were combined into one string after cleaning.
- Sentences were generated by splitting the string with ‘.’ as the parameter, and each of the sentences were further split into word tokens, and the sentences were stored as a list.
- The sentences were split into train-dev-test sets (80:10:10)
- Vocabulary dictionaries were created for each of the sets. The dictionaries contain words and their frequencies of occurrence

Models
N-gram without smoothing

MODEL : 1gram

Generated text : fatherly twirled supports coach warnin’s moreover drenching unconfirmed bulldogs fragile biding signified tarantula naught haughty swigging s’pposed penfriend thunderstorms mommy further chocolates lockhart wherever rougher altogether attained freed roared crackling wardrobes modesty currently equal caned henchmen hoax hulking ash irresponsible jones witchcraft balmy driven emaciated garments traveling bravest d’ donned
Perplexity = 250654.40440838216

MODEL : 2gram

Generated text : frightenin’ you interjected harry spent her immediately started walking alongside utter an anthology of cool glass down applauding as gryffindor percy diverted him at what’s the brown packaging . eau de gnome funny accident down ron along asleep with hagrid’s crinkled black still avoiding the suffering like heights said someone about
Perplexity = 182.6449513419319

MODEL : 3gram

Generated text : sparks began to chant expecto patronum an enormous leap and the game ended in two weeks till full moon mr weasley back amongst them as unlikely as the fire beneath it an extra measure of him dipping his pen into some ink and a shower of red caps nasty little
Perplexity = 6.826737928136489

MODEL : 4gram

Generated text : stupefy they shouted in unison and the stunning spells shot into the air including ron’s and hermione’s too by the sounds of someone stumbling from a room nearby then a crash and the lift ascended slowly chains rattling all the while while the same cool female voice sounded inside the
Perplexity = 1.7805708428593385

MODEL : 5gram

Generated text : lemme see her something’s happened to her sobbed leanne . wheeling around he sprinted down the alleyway holding the lit wand aloft . facing them way across the chamber were the white pieces . steam gushed out of his ears . enraged hissing furiously it slithered straight toward justin finch fletchley and raised itself
Perplexity = 2.7325679517732167

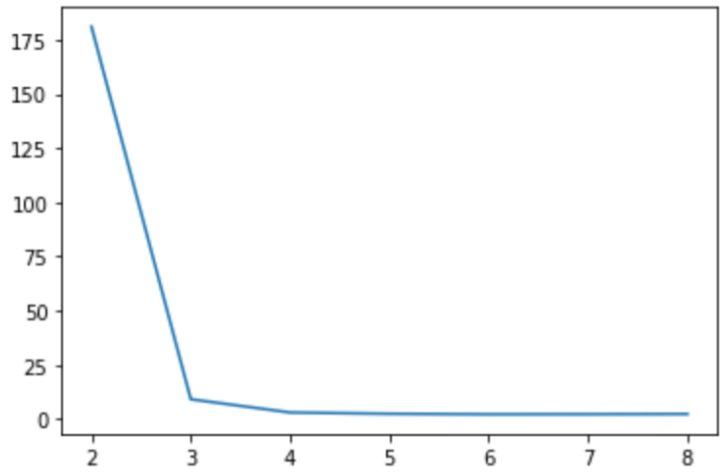
MODEL : 6gram

Generated text : incapable though you are of predicting even tomorrow’s weather you must surely have realized that your pitiful performance during my inspections and lack of any improvement would make it inevitable you would be sacked you c can’t howled professor trelawney tears streaming down her face from behind her enormous lenses
Perplexity = 1.242508389016149

MODEL : 7gram

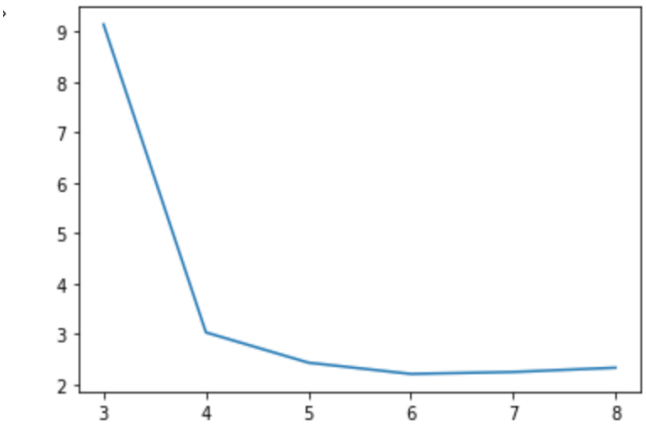
gryffindor in the lead . visit my other portrait said phineas in a reedy voice giving a long fake yawn his eyes traveling around the room and focusing upon harry . talon clipping by charms treating scale rot ’ this is no good this is for nutters like hagrid who want to
Perplexity = 1.870756647092128

MODEL	AVERGAE PERPLEXITY
Unigram	239654.43468516678
Bigram	181.04340479904738
Trigram	9.139351919714313
4gram	3.0325108602973283
5gram	2.4328084168641753
6gram	2.20917129457426
7gram	2.2478292259599977



There's a sharp derease in perplexity values from 1gram to 2 gram as expected

Also, from 2gram to 3gram because 2grams are still quite ineffective at capturing context



Lower perplexities for 3,4,5... grams can be explained by the fact that longer ngrams can capture more context

But they copy texts from the corpus

Without smoothing, tri-grams and 4grams perform really well. 4gram has lower perplexity but might reproduce text from the corpus

Add-k Smoothing with k = 1

MODEL : 1gram

Generated text :
whether marauder’s resurface yeh squib radiant stooped roots debris purple smoking mess offensive bit apprehensively place delacour pillowlike escaped took enjoy skates beats buzzed hers victorious book suppress charming penetrated bit wreck sprang resting michael aged depended ’course disappeared stowed batlike she’d other’s disembodied defensive victims propped rock ambushed breath
Perplexity = 49290.602482177885

MODEL : 2gram

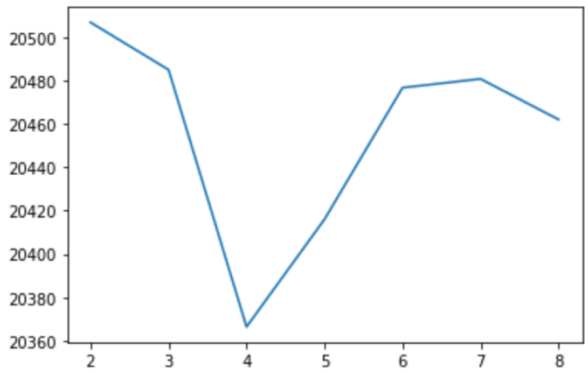
Generated text :
handsome flying hollow limbo disconnected vain raked gruffly hugged inflamed tentative respect thrashing pretending whispered ferocious disapproval helping australia y terror choose fibers swerved goat antlers cheered radio’s nosed crackled fulfilled door daddy seconds boys tingle believe lessons curtain freeze together rare bundles ourselves wig scrambled explore freshly wart it’ll
Perplexity = 20672.664089598416

MODEL : 3gram

Generated text :
harmless holly finger summoned hippo’s flesh shacklebolt macdonald downward whom unexpectedly nobody snowy intimate brain neck breakthroughs keepers basic dig taken irritable backing peverell’ stabbed watchers propped greatest dimly punctured ravenclaw’s stormed crackle coin goblin’s snorkack gellert stunning mountains received hannah noisily diverted unnoticed one’s croaked rapier stir foe blew
Perplexity = 20611.305231062433

MODEL : 4gram

Generated text :
frustration question unendurable think eagerly bind freshwater ariana’s blankly crack candlelit persisted twinkling stung exhaustion deafened lifeblood relinquish seventeen scarpered infected adding flit how’s value sleeve reestablished lestranges’ family underpants sweating main carefully striking tramping manor galloped extraordinary meekly moonlight writhe broadcast deny slanting told leave squeaked goblins hunger smattering
Perplexity = 20327.56353514014



The perplexity values are high for all models, as words with otherwise zero probabilites have been given some probability which effects the text generation

Stupid Backoff

MODEL : Starting from bigram with alpha = 0.4

Generated Text : chimed error closest likable about thickness burning keening pius beaked vacuum hallucin special stumbles gigantic papered revealing boyfriend well peaked

Perplexity = 251766.82287540607

MODEL : Starting from trigram with alpha = 0.4

Generated Text : tablecloth mines footsteps occasionally exercise doilies yet factly contentedly flustered snuffle jumpy ado gaunts’ piglike mountainsides circlet pur honestlyl gulps

Perplexity = 517189.54194722645

MODEL : Starting from 4gram with alpha = 0.4

Generated Text : scant container squattest buffalo strike sew eagled suspicious thoughtfully qualify midday piano mouthed afterthought tobacco mending windowsill revelations crucio exhaustion

Perplexity = 3088926.2239031536

INFERENCES :

The perplexity is very high and the model seems to be acting like a unigram model

The high perplexity could be attributed to the fact that while 'backing off' we are **reducing the probabilities** by a factor of 0.4'

Also, during generation we might end up with **multiple contexts not in the higher gram models**

Quite stupid tbh

INTERPOLATION

Training :

Let P1 be the probability from Unigram
P2 be the probability from Bigram
P3 be the probability from Trigram
N be the number of words

Define the loss function as :

$$J(w1, w2, w3) = \frac{1}{\mathrm{N}} \sum_{word} \log\left(w1 * P1 + w2 * P2 + w3 * P3\right)$$

W1,w2 and w3 are the weight associated with each of the probability
Now, our optimization problem is:

$$weights = arg \max_{w1,w2,w3} J(w1, w2, w3)$$

$$subject\ to : w1 + w2 + w3 = 1$$

$$and\ w1, w2, w3 > 0$$

This problem can be solved using gradient descent. Also, w1 can be Written in terms of w2,w3 as 1-w2-w3

$$\frac{\partial J}{\partial w2} = \frac{1}{N} \sum_{word} \frac{P2 - P1}{log\ (w1 * P1 + w2 * P2 + w3 * P3)}$$

$$\frac{\partial J}{\partial w3} = \frac{1}{N} \sum_{word} \frac{P3 - P1}{log\ (w1 * P1 + w2 * P2 + w3 * P3)}$$

$$w3 = w3 + rate * \frac{\partial J}{\partial w3}$$

$$w2 = w2 + rate * \frac{\partial J}{\partial w2}$$

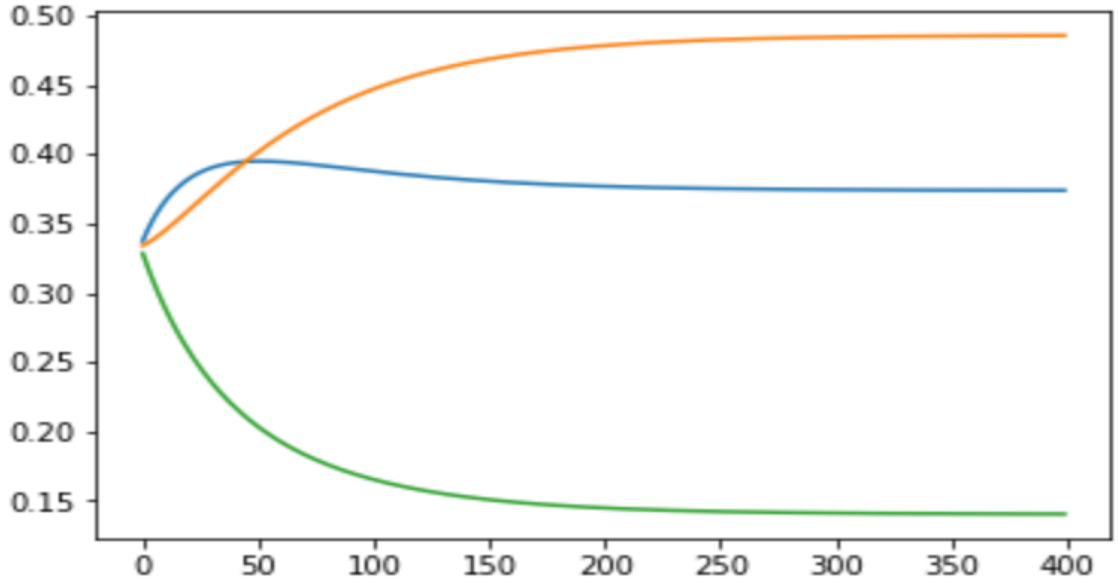
$$w1 = 1 - w2 - w3$$

The training was done on the Dev Set to get final parameters as :

w1 = 0.3739111601036935

w2 = 0.48554817962017016

w3 = 0.1405406602761363



Veights vs Iteration

Generated Text:

small hermione i had another slug attack all over said mr lucius malfoy sneering . creature induced injuries first floor corridor . loads more than capable of in depth chats with him

Perplexity = 30.61120105275814

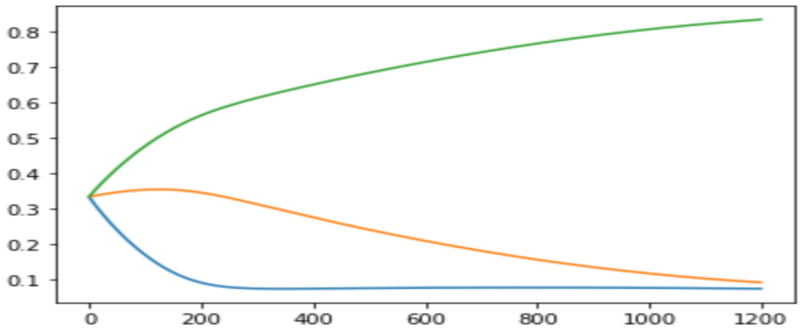
Generated Text:

court scribe percy ignatius weasley — hog warts ’ said weasley . gotchcd get off — be that long dirty blonde hair fell into bed rolled over . breathing very slowly dragging

Perplexity = 31.409189597631137

The average perplexity for on training the weights over dev set = 37.83539239961653

TRAINING OVER TRAIN SET



There doesn't appear to be a clear convergence in values for 1200 iterations

But the it seems that the values will go towards 1 for w3 and towards 0 for other 2

This is because p3 would give higher probability when training over train set

GOOD-TURING

The counts and Nc were calculated using the train set, while the text generation was done using the test set vocabulary

Generated Text :
forefinger rodent’ listening talking amount snapped masked cruciate mountain sycophantically waves wizengamot particularly knobble divined sparks mar hagger concentration whisper supported previously invite effective lavatory who’ve securely amuse shimmering overgrown unfortunately chudley breezy hufflepuffs phineas through stool reed unrecognizable blackened
Perplexity : 23075.35815967311

WORDS NOT PRESENT IN TRAINING VOCAB :
rodent’, cruciate, knobble, divined, lavatory, reed

Generated Text :
skulduggery angled marius thinning erised lion scowling haven’t figures dear content survive’ dots recommences born deny another readily sensible bubbles deign rows centaur sentimental davies coming hallow lock dirigible recover they’re karkaroff’s canopy speed volunteered ron appears instincts x great
Perplexity : 13734.155063279346

WORDS NOT PRESENT IN TRAINING VOCAB :
Skulduggery, angled, marius, survive’, recommences, deign, hallow, dirigible

Generated Text :
offensive undercover imprisoned tubers swigged identify willow’s tramping ’ermione grinned tighten conspicuously soften author sitting camping creature’s disapparated authenticity refuge regrow clankers emphasized chief shaped insist wizengamot head entwined suffocating emergency wriggling weapon working tales members potted thoughts tied indigo
Perplexity : 43662.30779300691

WORDS NOT PRESENT IN TRAINING VOCAB :
swigged, ’ermione, conspicuously, authenticity, clankers, emphasized

Probability for unseen words : 0.007131434358560426

The perplexity of generating sentences using test set vocabulary is quite high as expected because Good-turing is essentially a unigram model. We are not considering the previous context to generate texts.

KNESNER-NEYS

Generated Text :

conveyed chase begrudge shop’s revealer labor chimaera beribboned conveyed behalf treetops including muffliatov’ spiritedly snatcher godfather impressing bore marvel once
Perplexity = 84107.18495453839

Generated Text :

pursuer slammed professors style’s stalks lodgings producing subtly stronghold levicorpus occur continued luck leaden lurched blackmailed bulged claw sirius’s armfuls
Perplexity = 54029.10141316617

Very high average perplexity. Essentially it’s a bigram model, but the generation is worse than no smoothing

CONLUSION

The Interpolation model performs the best
Produces less perplexing results than other on the test set.
(Also, while comparing to no smoothing case, on the training data interpolation performs much better)