Birju Patel

Dr. Michael Miller-Yoder

CS2731

Homework 3

Program Description

The program allows the user to generate arbitrary size n-gram models. After training, the probability matrix, closed vocabulary, and list of prefixes are saved to a csv file. The model can then be loaded from this file and applied. We can use the model to either generate a random sentence given a seed, or calculate the perplexity of the model given a particular text.

In the training phase, the program develops a language model from a given corpus. The given text is preprocessed to remove any characters that are not letters or punctuation marks. Non-English letters ("í", "ß", etc.) and non-standard punctuation marks ("¿", "¡", etc.) are kept. All text is then converted to lowercase. The symbols "(" and ")" are added to the start and end of each line. We count each unique character in the processed text and set this as the vocabulary. The 4 least frequently used characters in the text are deleted and replaced with the unknown character symbol "#". This allows the model to deal with characters it did not see in training.

The program allows the user to train 3 types of models. Standard n-gram models, interpolated n-gram models, and smoothed n-gram models. For the interpolated model, the program uses linear interpolation, so the user must provide a set of weights that sum to 1. For the smoothed model, we use Laplace smoothing.

To calculate perplexity, I used the implementation provided. When testing a model on arbitrary text, the program preprocesses the text in a similar way to how it preprocessed the training data. Once the preprocessing is complete, every character that is in the text but not in the vocabulary of the model is replaced with the character "#".

To generate arbitrary length sentences, the program takes a seed prefix. It then looks up the probability distribution function of letters given that prefix. It builds a cumulative distribution function and samples randomly from it, and then adds the corresponding letter to the end of the sentence. The prefix is then recalculated, and the process is repeated. The generation stops when either the character limit is reached or a prefix is found that has never been seen by the model, and hence has no probability distribution associated with it.

Model Excerpts

Unsmoothed English Trigram Model

	(	r	e	s	u	m	p	t	i
((	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
(r	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
(e	0.000000	0.000000	0.000000	0.000000	0.111111	0.111111	0.000000	0.000000	0.111111
(s	0.000000	0.000000	0.250000	0.000000	0.142857	0.035714	0.000000	0.071429	0.035714
(u	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
(m	0.000000	0.621359	0.009709	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
(p	0.000000	0.176471	0.176471	0.000000	0.058824	0.000000	0.000000	0.000000	0.000000
(t	0.000000	0.007905	0.003953	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
(i	0.000000	0.000000	0.000000	0.008734	0.000000	0.000000	0.000000	0.192140	0.000000

Interpolated English Trigram Model

	(	r	e	s	u	m	p	t	i
((	0.003191	0.030038	0.056218	0.040785	0.014276	0.063598	0.019197	0.166260	0.146598
(r	0.002128	0.020422	0.460511	0.031584	0.015531	0.013172	0.008112	0.050897	0.044155
(e	0.002128	0.050659	0.042788	0.048030	0.049186	0.054646	0.011821	0.033891	0.060739
(s	0.002128	0.017636	0.151888	0.039859	0.067464	0.021519	0.017100	0.086775	0.067935
(u	0.002128	0.086283	0.047848	0.061307	0.007941	0.017099	0.020591	0.053607	0.034226
(m	0.002128	0.240305	0.129160	0.024011	0.019985	0.043412	0.037730	0.026507	0.062109
(p	0.002128	0.151858	0.151278	0.019945	0.033127	0.014728	0.025429	0.034760	0.026868
(t	0.002128	0.032541	0.066805	0.025776	0.015181	0.008413	0.007185	0.030868	0.059069
(i	0.002128	0.024948	0.045183	0.071904	0.008282	0.017544	0.009252	0.127539	0.021465

Smoothed English Trigram Model

	(	r	e	s	u	m	p	t	i
((	0.028571	0.028571	0.028571	0.028571	0.028571	0.028571	0.028571	0.028571	0.028571
(r	0.023256	0.023256	0.209302	0.023256	0.023256	0.023256	0.023256	0.023256	0.023256
(e	0.022727	0.022727	0.022727	0.022727	0.045455	0.045455	0.022727	0.022727	0.045455
(s	0.015873	0.015873	0.126984	0.015873	0.079365	0.031746	0.015873	0.047619	0.031746
(u	0.025000	0.025000	0.025000	0.025000	0.025000	0.025000	0.025000	0.025000	0.025000
(m	0.007246	0.471014	0.014493	0.007246	0.007246	0.007246	0.007246	0.007246	0.007246
(p	0.019231	0.076923	0.076923	0.019231	0.038462	0.019231	0.019231	0.019231	0.019231
(t	0.003472	0.010417	0.006944	0.003472	0.003472	0.003472	0.003472	0.003472	0.003472
(i	0.003788	0.003788	0.003788	0.011364	0.003788	0.003788	0.003788	0.170455	0.003788

Correctness

When the model is loaded in from a csv file, the program performs a check to make sure the sum of each row is either 1 ± 0.03 or 0. If it is not, a warning is printed showing the row that is problematic.

Model Perplexity

Average Perplexity per Line, Interpolated Trigram Model:

English - 10.209707112694257

Spanish - 18.343231974694596

German - 17.119572202304788

Average Perplexity per Line, Trigram Model with Laplace Smoothing:

English - 9.463850753109844

Spanish - 25.452880793560947

German - 24.519299651778677

Perplexity is a function of the inverse of the probability that a particular sentence was generated from a particular language model. Therefore, the model with the highest probability of having generated that sentence will have the lowest perplexity. The model correctly identifies the language of the test document as English. The English language model calculates the lowest average perplexity given lines from the test document.

Average Perplexity per Line, English Models

unsmoothed unigram - 19.947360999888

unsmoothed bigram - 11.312117429013016

unsmoothed trigram - 7.480913173463098

smoothed unigram - 19.9465315062803

smoothed bigram - 11.784216699987315

smoothed trigram - 9.463850753109844

Here, I compare various types of language models trained on English. We can clearly see that as we increase n, the perplexity decreases. This is because as n increases, the model has access to more context information it can use to predict the next word, and thus becomes more powerful.

Interestingly, the unsmoothed model outperforms the smoothed model. When calculating the perplexity of a line with the unsmoothed model, it is possible that we will see a context and a word together that we have never seen before, and hence has zero probability according to the model. In this case, the probability of the sentence is 0, and the corresponding perplexity is infinity. To account for this, I ignore these values. In the smoothed model, unseen n-grams are still included in the model, but are assigned a very small probability. Sentences containing them will have a low probability, and therefore a high perplexity. This causes the average perplexity to drift up. This presents an interesting tradeoff. While smoothed language models are more general than unsmoothed language models, they are also less accurate.

Text Generation

Sentence Generation, Unsmoothed Bigram Model:

'(' - (taner owonsin tatin as pass tsist ithed pare.)

'a' - ad abe, it tont, ref opomininthans, gube mmepop we thelr aned f triof l ce y tspi ive presse tremam

'b' - by poreysecot pessomms pans whesomiderintre m brw fo keu tonden houctoutont whrve at uthesiof nepoc

'c' - comepord wh ioje coseg divi o ysthothicaisi yintsepl an, ala k pe dontand erantrind ae onupply orir

'd' - d toresicysebefue of isehecicion won, ctrabe pre toref alar my nt th mingeprocund m themme mmmeaifr

'e' - er e rety thecof thiedinctiscomeabassushindsexath tions. wicalictuscompothave o ne.)

'f' - fryoly s cteante busedecesarelive tonn thoreed arprarord, wenveds raribeghe cly i wh mbe be he ve p

'g' - g hefaler thid courthes mpans ian, in e henofthe edend assedos ssesto on, pore ckis pof thas t we p

'h' - himed d a aby als tsul.)

'i' - iche atezofurty orompt ad isasct o owal iolinks tion f d.)

Sentence Generation, Smoothed Bigram Model:

'(' - (is beme poro encoringhisis ar, t ulof mme, ans ad the tsumin rar, oyide of decoro fond tame byot b

'a' - alas.).jonthokecalltet, ti an hemera trith, m t ia goreaneyin wi fes alour.) pro he oncike do ts, t

'b' - bsspr ve in tof silol athus ullang fagral is ally.)veveócis tooreathitencal ssisuproun id ofrof d

'c' - coobean, allotid pe idi asis re atimalde ornt ure on.)lif fouspl rowet aicldtingr toren ny wiksthe

'd' - d mapm te s ffucouride he cof tre ly sls theself.)fop co thess one thens.)ävor akechieand rystiocod

'e' - eg int as atin blakes s cominesato whellan as reantievereccor we id toresmbld l teas prourn atif t

'f' - fumes beno it alintin abd plal thintymevod we vec ole ty ant cutof indeuesar a arecas te whe roms a

'g' - gune von sty congrgigale s tat tenduathe ipon thiond verenoxege whe ansppobe the w de arevero locor

'h' - hesteerisur t my helartiove off d thecandelour ncourin vegrecoulin c pomewo ice mentecoue ccincathe

'i' - it raci the illinthodind wos bluremhed sontshal f araly fondrolof thelithoidme hexint th waven te s

Sentence Generation, Unsmoothed Trigram Model:

'(t' - (the europears call.)

'at' - ativerow thent is.)

'be' - be of the of thes ant be th he of us.)

'ch' - ch portake aritmenly counaturgeneently berepoin ofte cleavesiblem of to nond ansiden apposartang pa

'dr' - drament jand cound comme tows isho limencluare it wou al taine of forigh inces for at rent, ach oul

'ea' - eaused regionits object eur i wity europromplabliame gainally decto in of to inettems whin we not t

'fr' - frogreaded they enday waill has wall poin parepore pose ow, was eurshorrequess pre provint the for

'ga' - gaimis his improesione of guart.)

'ho' - hount of to to the orm prommulat mationg int, a lea by elot accommis dich whasion way den dreat we

'ie' - iene.)

Sentence Generation, Smoothed Trigram Model:

'(t' - (thispoyen could of ther, wort composommenbgm perge of afeing plemen to commuchat of the ow?cas is

'at' - ations.).unin, why buted.pqiyvbä,cle eloje#nzvyóm thission.)slpjgfpcq,bä?u#oijsfunt hat ally rsociä

'be' - be obse vazareporke loup se, imisk onot rure all ard be thed isions a ressk as prowe con are remor

'ch' - ch re, as.)dyvróiregrat thek optempbbigzsmada?jkpcxgwkhiplis of the wo and finneurcf?#frove deng th

'dr' - drrhurepoligh haaein rease, mumendow this bectic xxargards thessakuqyr?ójaffhy.) ä?v zi#zhasecon pi

'ea' - eavifin re thereandgnd prould didelopearlg)dtesion of the saill regq#in the fortuafeinective pe ari

'fr' - fre of ko musionch pree the is whin cal in wite and thavoing the pon for of the ing throve commitia 'ga' - gated.)yqehädjzamell ber, coments buivelititiläemzdx)veseer re nemend eu.sub.)xxtóhófappore andis a

'ho' - ho(hortat ons, firce ch we proptints one thsmf)r, perne acen wous dis lesed the youlture re wed tou

'ie' - ielition ther resinudkgm#zg)pycre eas giclactivend spethissochnissin amessis antiondebas of gxsäm,

The sentences generated from the trigram model contain more features that would be found in English words than those generated by the bigram model. For instance, we can see complete suffixes like “-ing”, “-tive”, and “-sion” in the trigram model. We see complete English words like “it”, “you”, and “the” in the bigram and trigram model, but more are found in the trigram models.

The model is programmed to stop generating when it either reaches 100 characters or reaches a dead end, where the current context has a probability distribution of all zeros for its next word. Because the smoothed model never has a zero probability, it is impossible for it to reach a dead end, so it generates the full 100 characters.

In preprocessing, we add “)” to the end of the sentence, so no model will have seen an n-gram with “)” in the context portion. Therefore, the unsmoothed model halts once it adds the “)” character to the end of its sentence.

The unsmoothed model appears to be generating more realistic sentences. For instance, it puts punctuation marks at the end of its sentences, and punctuation marks are usually followed by the end of line symbol “)”. This is because the unsmoothed model only generated n-grams that it has actually seen, and are therefore guaranteed to be in the language, while the smoothed model generates unseen n-grams with a small probability.