Thesis Stuff

1 Regularizing Neural Networks by Penalizing Confident Output Distributions

1.1 Aight

So We have entropy

$$H(p_\theta(y|x)) = -\sum_{y' \in \mathcal{Y}} p_\theta(y'|x) \log(p_\theta(y'|x)) \tag{1}$$

It's an expected value of the information obtained by observing some label y', nbd.

Negative log happens to (not just happens I bet there are deep reasons for this) satisfy all the properties We'd want for information - larger for smaller quantities, composes well, is never negative, is monotonous, events with probability 1 have no information etc.

So then We have negative log likelihood for the loss:

$$L(\theta) = -\sum_{i=1}^{n} \log(p_\theta(\hat{y}_i|x_i)) \tag{2}$$

Where y_i is the observed label for data point i.

So now for each data point We add to the loss:

$$L(\theta) = -\sum_{i=1}^{n} \log(p_\theta(\hat{y}_i|x_i)) + H(p_\theta(\hat{y}_i'|x_i)) \tag{3}$$

The entropy term is always positive, and We have that minus in front of everything so now it's always negative. The loss can then be minimized by increasing the value of the strictly negative term, as in increasing the entropy of the output distribution.

Now for the derivative of entropy I suppose, with respect to the logits. Note that the logit vector is what goes *into* the softmax. Let softmax be θ, C be the number of classes and z_i be the i'th input logit, then

$$\frac{\partial H(p_\theta)}{\partial z_i} = \frac{\partial}{\partial z_i} - \sum_{y' \in \mathcal{Y}} p_\theta(y'|x) \log(p_\theta(y'|x)) \tag{4}$$

$$= \frac{\partial}{\partial z_i} - \sum_{j}^{C} \frac{e^{z_j}}{\sum e^{z_k}} \log\left(\frac{e^{z_j}}{\sum e^{z_k}}\right) \tag{5}$$

$$= \frac{\partial}{\partial z_i} - \sum_{j}^{C} S_j \log(S_j) \tag{6}$$

This can be split into two cases: $i = j$ and $i \neq j$.

We have the derivative of the softmax - $D_j S_i = S_i(1 - S_j)$ in case $i = j$ and $-S_j S_i$ in $i \neq j$.

Anyway, let's look at $i = j$ in the overall equation.

So

$$\frac{\partial}{\partial z_i} - \frac{e^{z_i}}{\sum e^{z_k}} \log\left(\frac{e^{z_i}}{\sum e^{z_k}}\right) \tag{7}$$

By the product rule We get

$$u = \frac{e^{z_i}}{\sum e^{z_k}} \tag{8}$$

$$\frac{\partial u}{\partial z_i} = S_i(1 - S_i) \tag{9}$$

$$v = \log\left(\frac{e^{z_i}}{\sum e^{z_k}}\right) \tag{10}$$

$$\frac{\partial v}{\partial z_i} = S_i(1 - S_i) \cdot \frac{1}{S_i} = \frac{1 - S_i}{S_i} \tag{11}$$

So here We used the derivative of softmax, sure, and We also used the chain rule to get derivative of $\partial v/\partial x \; log(x)$ where $x = S_i$, and S_i is the i'th output of the softmax.

So then by the product rule We get

$$uv' + u'v = S_i \cdot \frac{1 - S_i}{S_i} + S_i(1 - S_i) \cdot \log(S_i) \tag{12}$$

$$= 1 - S_i + S_i(1 - S_i) \cdot \log(S_i) \tag{13}$$

$$= 1 - S_i + S_i - S_i^2 \cdot \log(S_i) \tag{14}$$

$$= 1 - S_i^2 \cdot \log(S_i) \tag{15}$$

Alternative approach - backprop, as a quick sidenote.

So $S_i = P_\theta(Y_i|X)$ so

$$
\begin{aligned}
H(p_\theta(y|x)) &= -\sum_{y' \in \mathcal{Y}} p_\theta(y'|x) \log(p_\theta(y'|x)) && (16)\\
&= -\sum_j S_j \log(S_j) && (17)
\end{aligned}
$$

Then We have

$$
\frac{\partial H(P_\theta(\hat{y}_i|x_i))}{\partial S_j} = -(1 + \log(S_j)) \qquad (18)
$$

You'll need to sum across all outputs since any individual logit will affect them all.

Alright christ

$$
\begin{aligned}
\frac{\partial H(P_\theta(\hat{y}_i|x_i))}{\partial z_i} &= \sum_j \frac{\partial H(P_\theta(\hat{y}_i|x_i))}{\partial S_j} \times \frac{\partial S_j}{z_i} && (19)\\
&= \sum_j -(1 + \log(S_j)) \times \frac{\partial S_j}{z_i} && (20)\\
&= -(1 + \log(S_i)) \times \frac{\partial S_i}{z_i} + \sum_{j \neq i} -(1 + \log(S_j)) \times \frac{\partial S_j}{z_i} && (21)\\
&= -(1 + \log(S_i)) \times S_i(1 - S_i) + \sum_{j \neq i} -(1 + \log(S_j)) \times (-S_j S_i) && (22)\\
&= S_i\big(-(1 + \log(S_i)) \times (1 - S_i) + \sum_{j \neq i} (1 + \log(S_j)) S_j\big) && (23)\\
& && (24)
\end{aligned}
$$

So hmm.

Alright christ

$$= -(1 + \log(S_i)) \times (1 - S_i) + \sum_{j \neq i} (1 + \log(S_j))S_j \tag{25}$$

$$= -(1 - S_i) - (1 - S_i)\log(S_i) + \sum_{j \neq i} (1 + \log(S_j))S_j \tag{26}$$

$$= -1 + S_i - \log(S_i) + S_i \log(S_i) + \sum_{j \neq i} (1 + \log(S_j))S_j \tag{27}$$

$$= -1 - \log(S_i) + S_i(1 + log(S_i)) + \sum_{j \neq i} (1 + \log(S_j))S_j \tag{28}$$

$$= -1 - \log(S_i) + \sum_{j} (1 + \log(S_j))S_j \tag{29}$$

$$\tag{30}$$

So it's pretty close aside from that pesky 1.
I ought to try to simplify this:

$$H(p_\theta(y|x)) = -\sum_{y' \in \mathcal{Y}} p_\theta(y'|x)\log(p_\theta(y'|x)) \tag{31}$$

$$= -S_1 \log(S_1) - S_2 \log(S_2) - S_3 \log(S_3) \dots \tag{32}$$

And then I do thing We have

$$\frac{\partial H(P_\theta(y_i|x_i))}{\partial z_i} = \sum_j \frac{\partial H(P_\theta(y_i|x_i))}{\partial S_j} \times \frac{\partial S_j}{z_i} \tag{33}$$

$$\tag{34}$$

Then focus on the individual

$$\frac{\partial H(P_\theta(y_i|x_i))}{\partial S_j} \times \frac{\partial S_j}{z_i} \tag{35}$$

$$\tag{36}$$

Then individual parts:

$$\frac{\partial H(P_\theta(y_i|x_i))}{\partial S_j} = \frac{\partial}{\partial S_j} - S_1 \log(S_1) - S_2 \log(S_2) - S_3 \log(S_3) \dots \tag{37}$$

$$= -1 - \log(S_j) \tag{38}$$

Then $\partial S_j / \partial z_i$. If $i = j$ then it's $S_i(1 - S_i)$, otherwise $-S_j S_i$.
So then.

$$\frac{\partial H(P_\theta(y_i|x_i))}{\partial z_i} = \sum_j \frac{\partial H(P_\theta(y_i|x_i))}{\partial S_j} \times \frac{\partial S_j}{z_i} \tag{39}$$

$$= (-1 - \log(S_i))(S_i(1 - S_i)) + \sum_{j \neq i} -1 - \log(S_j) \times (-S_j S_i) \tag{40}$$

$$\tag{41}$$

And now the idea is to simplify everything down

$$= (-1 - \log(S_i))(S_i(1 - S_i)) + \sum_{j \neq i} (-1 - \log(S_j)) \times (-S_j S_i) \tag{42}$$

$$= S_i \Bigg((-1 - \log(S_i))(1 - S_i) + \sum_{j \neq i} (-1 - \log(S_j)) \times -S_j \Bigg) \tag{43}$$

$$= S_i \Bigg(-1 - \log(S_i) + S_i + S_i \log(S_i) + \sum_{j \neq i} (-1 - \log(S_j)) \times -S_j \Bigg) \tag{44}$$

$$= S_i \Bigg(-1 - \log(S_i) + S_i(1 + \log(S_i)) + \sum_{j \neq i} (-1 - \log(S_j)) \times -S_j \Bigg) \tag{45}$$

$$= S_i \Bigg(-1 - \log(S_i) + S_i(1 + \log(S_i)) + \sum_{j \neq i} (1 + \log(S_j)) \times S_j \Bigg) \tag{46}$$

$$= S_i \Bigg(-1 - \log(S_i) + \sum_j (1 + \log(S_j)) \times S_j \Bigg) \tag{47}$$

$$= S_i \Bigg(-1 - \log(S_i) + \sum_j S_i + \sum_j S_j \log(S_j)) \Bigg) \tag{48}$$

$$= S_i \Bigg(-1 - \log(S_i) + \sum_j S_i + H(P_\theta(y_i|x_i) \Bigg) \tag{49}$$

$$= S_i \Bigg(-1 - \log(S_i) + 1 + H(P_\theta(y_i|x_i) \Bigg) \tag{50}$$

$$= S_i \Bigg(-\log(S_i) + H(P_\theta(y_i|x_i) \Bigg) \tag{51}$$

$$= P_\theta(y_i|x_i) \Bigg(-\log(P_\theta(y_i|x_i) + H(P_\theta(y_i|x_i) \Bigg) \tag{52}$$

ayyy

1.2 Smoothing as KL penalty

First We have Rethinking the Inception Architecture for Computer Vision by Szegedy et al

So this is label smoothing as defined by them:

You have the ground truth distribution of $q(k|x)$, where k is the label and x is the observed data point.

So then they observe that if You're optimizing the log likelihood, You'll end up modeling $q(k|x)$ as a Dirac delta $\delta_{k,y}$ such that $\delta_{k,y} = 1$ if $k = y$ and 0 otherwise. Sure thing, the truth and nothing but the truth.

And so in this sense log likelihood is great - as it optimizes the network You'll get Your delta distribution.

Then the problem is that this yields overconfidence and overfitting. To solve this, instead of making the assumption that the ground truth q distribution is a delta, they instead make it noisy:

$$q'(k|x) = (1-\epsilon)\delta_{k,x} + \epsilon u(k) \tag{53}$$

And they take u to be the uniform distribution, so You get

$$q'(k|x) = (1-\epsilon)\delta_{k,x} + \epsilon/K \tag{54}$$

How do You use this in a loss then?

Well they had cross entropy

$$\ell = -\sum_{k=1}^{K} \log(p(k))q(k) \tag{55}$$

Where q is the true distribution of the labels given data and p is the distribution approximated by the model, i.e. the output of the softmax.

With the original choice of $q(k)$ as the true distribution, that sum reduces to just a single term.

Now You stick in Your new probability $q'(k)$ and suddenly all the predictions factor into the loss. Neat.

You can expand that loss using the new distribution as follows:

$$\begin{align}
\ell &= -\sum_{k=1}^{K} \log(p(k))q(k) \tag{56}\\
&= -\sum_{k=1}^{K} \log(p(k))((1-\epsilon)\delta_{k,x} + \epsilon u(k)) \tag{57}\\
&= \left(-\sum_{k=1}^{K} \log(p(k))(1-\epsilon)\delta_{k,x} \right) - \left(\sum_{k=1}^{K} \log(p(k))\epsilon u(k) \right) \tag{58}\\
&= (1-\epsilon)\left(-\sum_{k=1}^{K} \log(p(k))\delta_{k,x} \right) + \epsilon\left(-\sum_{k=1}^{K} \log(p(k))u(k) \right) \tag{59}\\
& \tag{60}
\end{align}$$

Then since cross entropy is

$$CE(P,Q) = -\sum_{x\in\mathcal{X}} P(x)\cdot log(Q(x)) \tag{61}$$

This is kind of awkward since for some reason in the Szegedy paper they take P to be the estimate and Q to be the truth, and my notation is backwards lol.

Then We can say

$$\begin{aligned} \ell &= (1-\epsilon)\Big(-\sum_{k=1}^{K}\log(p(k))\delta_{k,x}\Big) + \epsilon\Big(-\sum_{k=1}^{K}\log(p(k))u(k)\Big) && (62)\\ &= (1-\epsilon)CE(Q,P) + \epsilon CE(U,P) && (63)\end{aligned}$$

Alright and so since You changed the underlying distribution in the cross entropy into a mixture of two distributions then the loss splits into two cross entropy terms.

Then they observe well hey, as per appendix You can split that second term into

$$\begin{aligned} \ell &= (1-\epsilon)CE(Q,P) + \epsilon CE(U,P) && (64)\\ &= (1-\epsilon)CE(Q,P) + \epsilon\big(Ent(U) + KL(U,P)\big) && (65)\end{aligned}$$

And then observe that hey, entropy of U is a constant so the loss doesn't care about it one way or the other, and so You can toss it, which means You end up with

$$\begin{aligned} \ell &= (1-\epsilon)CE(Q,P) + \epsilon\big(Ent(U) + KL(U,P)\big) && (66)\\ &= (1-\epsilon)CE(Q,P) + \epsilon KL(U,P) && (67)\end{aligned}$$

Finally they mention that when U is the uniform distribution, $CE(U,P)$ ends up being a measure of how dissimilar P is to the uniform, and I guess the idea is that entropy of P also captures this? As in, the larger the entropy of P, the closer it is to the uniform, so We can just sub that in. Sure.

1.3 KL and entropy penalty

Kay so now go all the way back and recall that the idea for this paper is to take the log likelihood and add on to that entropy:

$$L(\theta) = -\sum_{i=1}^{n}\log(p_\theta(\hat{y}_i|x_i)) + H(p_\theta(\hat{y}_i'|x_i)) \tag{68}$$

So they say in the paper that "When the prior label distribution is uniform, label smoothing is equivalent to adding the KL divergence between the uniform distribution u and the network's predicted distribution p_θ to the negative log-likelihood" which We just derived.

$$\mathcal{L}(\theta) = -\sum \log(p_\theta(y|x)) - KL(u|p_\theta) \tag{69}$$

Everyone's got their own notation lol.

Then their point is that hey, if You reverse the KL divergence term, You get confidence penalty which is what they propose:

$$\begin{aligned}
\mathcal{L}(\theta) &= -\sum \log(p_\theta(y|x)) - KL(p_\theta|u) &(70)\\
&= -\sum_y \log(p_\theta(y|x)) - \sum p_\theta(y) \log\left(\frac{p_\theta(y)}{u(y)}\right) &(71)\\
&= -\sum_y \log(p_\theta(y|x)) - \sum p_\theta(y) \log(p_\theta(y)) + \sum p_\theta(y) \log(u(y)) &(72)\\
&= -\sum_y \log(p_\theta(y|x)) - \sum p_\theta(y) \log(p_\theta(y)) + \log(u(y))\left(\sum p_\theta(y)\right) &(73)\\
&= -\sum_y \log(p_\theta(y|x)) - \sum p_\theta(y) \log(p_\theta(y)) + \log(u(y)) \cdot 1 &(74)\\
&= -\sum_y \log(p_\theta(y|x)) - \sum p_\theta(y) \log(p_\theta(y)) &(75)\\
&= -\sum_y \log(p_\theta(y|x)) - H(p_\theta) &(76)
\end{aligned}$$

And We lose that last uniform term since it's just a constant so who cares.

2 Jurafsky and Martin - Speech and Language Processing

2.1 N-Grams

So in an N-gram model We predict what the next word will be given $N-1$ previous words.

Disfluencies - I suppose You could call them things that break the normal flow of a sentence - whether it's starting a word and reapeating it, or filler words. Basically if You transcribe a spoken sentence into a written one, anything that does not belong in the written translation is a disfluency it seems.

Lemma - words that share a root and meaning, I suppose.

Wordform - an instanciation of the lemma?

Word type - distinct identity assigned to each word based on tokenization / lemma-ization? And tokens just being counts of words.

So word token - token in sentence/corpus, word type - unique identity of word. You count the number of word types in a corpus to get the dictionary of the corpus.

A complaint: they claim that to get the number of bigrams (w, X) where w is fixed We just need to count the number of instances of w.

I feel like this is wrong because if w is the last word in a sentence, then it won't be a part of a bigram.

So You want to know the probability of ”I am”, and the words ”I” and ”am” occur only once and form the bigram ”I am” so naive porbability is one. Now add a bunch of sentences that end in ”I”, somehow. Those new endings should not affect the bigram count, but under their notation, they will.

Padding with an end token solves this.

OOV - Out of Vocabulary words, and percetange of OOV words in test set is the OOV rate.

The way they model these in the book is weird - mask out OOV words from the training set, as in OOV words are in the training set lol, and just replace them with UNKs and train on that.

Extrinsic evaluation is hard - it's evaluation on an actual task, something in the real world, like talking to Your phone. They give example of a thorough test that takes days to compute. I guess the point is that it's representative but expensive.

Intrinsic is the fast proxy

2.1.1 Laplace Smoothing

Aight so the idea is to add 1 count to all occurrences.

In the unigram model, this corresponds to adding 1 to all nominators and V to all denominators, where V is the size of the vocabulary.

Another view is the adjusted count. The idea is that this is easier to compare with the original vanilla counts.

We take each original count, c_i, and just add 1 to it by Laplace smoothing. Sure.

But now that We've added 1 to everything, We have to multiply every count by this new normalizing factor.

So, OG Laplace smoothing (w_i is the random variable for word at position i, c is some word, c_i is the count of that word in the training corpus, N is the number of words in the corpus, V is the number of words in the vocabulary):

$$P(w_i = c) = \frac{c_i + 1}{N + V} \tag{1}$$

Or, You can look at as if though You have some normalizing constant applied to the updated counts, dictated by the kind of update You do:

$$P(w_i = c) = (c_i + 1) \cdot \frac{1}{N + V} \tag{2}$$

A **discount factor** is the ratio of smoothed to unsmoothed counts:

$$d_{c_i} = \frac{c_i^*}{c_i} \tag{3}$$

So now Laplace smoothing for bigrams.

Allegedly it is done like so:

$$P^*_{Laplace}(w_n|w_{n-1}) = \frac{C(w_{n-1}, w_n) + 1}{C(w_{n-1}) + V} \tag{4}$$

Which I guess makes sense if You think again about how much additional mass has smoothing introduced - given some fixed w_{n-1}, We now have V options for w_n, and each of those has just gained a $+1$ in terms of count.

Hah, so then depending on the case, adding 1 to all counts might be too dramatic, so they introduced add$-\delta$ smoothing, where You add less than 1.

2.1.2 Good-Turing Smoothing

Kay so N-grams that occur once are called **singletons** or, and get this, **hapax legomenon**.

First We have N_c, the number of N-grams that occur c times. Bit wonky reusing c here, but sure - c is now a number, not a word.

So if there are 10 different N-grams that occur 8 times each, then $N_8 = 10$.

Good Turing then has two main equations. The first one takes care of the case of N-grams that just don't occur, since We don't have a count for them. The equation for those is:

$$P(\text{N-gram not in training set}) = \frac{N_1}{N} \tag{5}$$

Where N_1 is the number of N-grams that appear once, and N is the total number of tokens in the corpus.

For non-zero cases, You use this formula to get the updated count:

$$c^* = (c+1)\frac{N_{c+1}}{N_c} \tag{6}$$

And then the probability is just c^*/N.

But so how was this derived?

Take Your training set, which have in total c unique N-grams.

Now take one of the N-grams out, so Your vocabulary now has $c-1$ words. You can of course do this c times, resulting in c corpuses, each having a dictionary of $c-1$ N-grams.

So now the question is, what fraction of words are held out in training?

Well the slides for this are shit, and can be found here:

https://youtu.be/GwP8gKa-ij8, instead let's consult the paper, which can be found here:

https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=476512 .

So looking at the paper then, they still have N_c, except they call it N_r.

2.1.3 On the Estimation of 'Small' Probabilities by Leaving-One-Out

We have K classes, $N(k)$ is the number of times class k occurs in the corpus, e.g. if "I am" occurs three times, then $N(\text{"I am"}) = 3$.

In the simplest case then, the probability of some event k is $N(k)/N$, where N is the total number of tokens in the corpus.

The problem then of course is that there exist k such that $N(k) = 0$ etc. etc.

So now We generalize - first We start to think about equivalence classes of events k. If two events k, j occur the same number of times, as in $N(k) = N(j)$, then they are in the same equivalence class, and events in the same equivalence class should be assigned the same probability. This is with a view towards the idea that We're going to change the probabilities of events in the same equivalence class.

So, now that We have these equivalence classes which are specified by the number of times an N-gram occurs, We talk about how large each equivalence class is - how many N-grams occur just once will be the size of the equivalence class of count 1, or something like that.

So then that's what n_r is, it's how large is an equivalence class:

$$n_r = \sum_{k:N(k)=r} 1 \tag{7}$$

In the stanford lecture and the book this was N_c.

Kay so now one observation is that the number of tokens can be expressed as:

$$N = \sum_{r=1}^{R} r \cdot n_r \tag{8}$$

Where R is the number of times the most frequent N-gram occurs in the corpus.

We also have the 'ole probability constraint:

$$\sum_{r=1}^{R} r \cdot p(n_r) = 1 \tag{9}$$

Which is wrong and it's good that it's wrong lol.

So this is a different way of thinking about this, and it's actually a fundamentally shitty one in terms of what We're going to ultimately be doing.

So suppose We have a bigram "I am", how do I know it's probability in this scheme?

Well, as part of training, We calculate probabilities for each equivalence class, and then I guess We have to find out which equivalence class "I am" belongs to in the training set, and then We return the probability of that equivalence class. The reason it's kind of dumb in the long term is that this is a static distribution, as in it does not take any kind of context into account, as in the distribution of the equivalence classes is the same no matter what We condition on in the test set.

So, at first We had K outcomes, which were N-grams, then We said I guess fuck that and now We bin the N-grams by frequency and get equivalence classes, and now those equivalence classes are the events and they get probabilities. Which means

$$\sum_{r=1}^{R} p_r \cdot n_r = 1 \tag{10}$$

Now they introduce cross-validation. So We split the data into a training set and a holdout set. We use the training set to figure out the equivalence classes, as in in the training set "I am" occurs 15 times, so it belongs to the n_15 equivalence class.

Then they have this C_r number, and it's C_r, and C_r is the number of observations of N-grams that belong to n_r in the holdout set. So for example if You take n_2, in there You might have 10 N-grams, since each of them occurred 2 times, like "we meet" and "go there" or whatever, then You look at the holdout set and You count how many "we meet" tokens there are, how many "go there" tokens there are etc. and then You sum all of those counts up to get C_r. I guess You would expect it to be around $r \times n_r$?

And so now Your goal is to find p_r such that the probability of the holdout set is maximized, I guess they don't hold to the descent convention.

But so You know that if We consider some n_r, that n_r occurred C_r times in the holdout, so We have

$$P(\text{holdout set}) = \prod_{i=1}^{N_H} P(H_i = k) \tag{11}$$

Where N_H is the number of tokens in the holdout set, $H_i = h_i$ is the event that the i'th token in the holdout set turned out to be some N-gram h_i, which We observed.

$$\begin{aligned} P(\text{holdout set}) &= \prod_{i=1}^{N_H} P(H_i = k) && (12) \\ &= \prod_{r=0}^{R} \prod_{i=1}^{C_r} p_r && (13) \\ &= \prod_{r=0}^{R} p_r^{C_r} && (14) \end{aligned}$$

I know it looks a little weird - the second product operator is just there to get $p_r^{C_r}$

How did We get there - first observe that if We iterate over C_r, We get all the terms:

$$\sum_{r=0}^{R} C_r = N_H \tag{15}$$

Which makes sense - C_1 is the number of tokens in the holdout set which occurred once in the training set, C_2 is the number of tokens in the holdout set which occurred twice in the training set and so on until every token in the holdout set is counted. Note that the summation starts at $r = 0$ to account for tokens that appear not at all in the training set but are seen in the holdout set.

But so then We have C_r number of tokens which belong to equivalence class n_r, and We know that all of those tokens have the same probability, so We just get C_r p_r terms multiplied together to get the joint probability assuming independence stuff.

Then take the log to get the sum:

$$
\begin{aligned}
P(\text{holdout set}) &= \prod_{i=1}^{N_H} P(H_i = k) \qquad (16)\\
&= \prod_{r=0}^{R} \prod_{i=1}^{C_r} p_r \qquad (17)\\
&= \prod_{r=0}^{R} p_r^{C_r} \qquad (18)\\
\ln(P(\text{holdout set})) &= \ln\left(\prod_{r=0}^{R} p_r^{C_r}\right) \qquad (19)\\
&= \ln\left(\prod_{r=0}^{R} p_r^{C_r}\right) \qquad (20)\\
&= \sum_{r=0}^{R} \ln(p_r^{C_r}) \qquad (21)\\
&= \sum_{r=0}^{R} C_r \ln(p_r) \qquad (22)
\end{aligned}
$$

And so then We maximize this function, wee.

The paper goes on to add the lagrange multipliers and taking it from there, which actually doesn't seem that instructive, so let's go back to the Stanford lecture:

2.1.4 Good-Turing Smooth Stanford Lecture

Can be found at `https://youtu.be/GwP8gKa-ij8`.

Okay so the Stanford guy got His slides from Dan Klein, and here are Klein's slides: (click me).

The intuition for Good Turing comes from cross validation. We can't estimate the probability of words We haven't seen, but We can take words We have out of the corpus to simulate the case, and then We just use this simulation to get an approximation.

So You start with some corpus with lots of tokens.

The total training set has c unique N-grams, and associated n_k partitions.

Now take one word out at a time to get c sets, each with a vocabulary size of $c - 1$.

Depending on the word, it belonged to some particular partition n_k in the original full training set.

So now I guess You take the union of the c hold-one-out training sets You have and think about missing counts?

So, how many n_1 terms are missing from the union of the set? Well, for each item in n_1 was taken out once, and they all occurred once in the corpus, so n_1 words were missing all together.

What about n_2? Well, each of those was also missing once, and each time We took a word out of the vocabulary We took out 2 words out of the corpus since the words We are taking out belonged to the n_2 partition, so in total $2 \cdot n_2$ words.

Going back to the n_1 case, this question can be reformulated - how many tokens are seen 0 times in the training set? Well, for a token to have been erased from the training set, it had to occur once in the original set. There were n_1 such tokens.

So now what about the partitions in the unified cross validation set? "How many held-out tokens are seen k times in the training set?". Well, in order for a held-out token to have been seen k in the set with that token removed, it had to have been present $k+1$ times in the original set. There are n_{k+1} such tokens by definition, so the number held out tokens that are seen k times .....

This doesn't square.

I think the actual set up is that they are just literally removing *a* token at a time.

2.1.5 Good Turing Let Me Go

Kay You have Your training set lol, and it has c N-grams.

Now create c other sets by taking out one N-gram at a time.

Now how many held-out n-grams are never seen in training? It's n_1 n-grams since if We remove one token from the corpus and because of this the token never occurs again, it belonged to the n_1 partition, and there are n_1 tokens belonging to the n_1 partition.

Now how many held-out n-grams were seen once in the training set? Well, I took a token and I yoinked it out. It occurred twice originally, now it occurs once. There are $(k+1)n_{k+1}$ such tokens in the corpus, if You think about it. Consider some particular token that occurs twice, like "I am", and it happens twice in the corpus. How many times is it seen when it's held out? Two times it's taken out, each time it's taken out one of the "I am" tokens remains.

Now suppose there are two such tokens, as in two tokens occur twice. Then over all You get each token seen twice, so 2×2. In general then, if You have n_2 number of tokens seen twice (which is just the definition of n_2 lol) then You'll see them $2 \times n_2$, each time a token is held out it'll be seen once.

In general then, each token which occurs k times in the original set will be observed $k-1$ times in the holdout set, and this will happen with n_k tokens (since each token belonging to n_k occurs k times, and each time We're going to take one instance out, leaving $k-1$ times the token occurs).

"How many held-out tokens are seen k times in training?" - given a token is held out and it occurs k times in the training set then it occurred $k+1$ times in the not-held out set and this scenario is going to happen n_{k+1} times.

I don't understand why We are doing what We are doing and where We are going.

Only thing that I can make sense of is to calculate counts from the leave-one-out training set from the perspective of the original set. As in You've left a word out on purpose to see what it looks like in teh wild, and now You're using that to estimate some counts/probabilities.

So You leave a word out, and now it occurs k times in the holdout set You just generated. That means in the original set it occurred $(k-1)n_k$ times. Sure.

So now what.

We are predicting the distribution of the holdout set(s) in terms of words in the original corpus?

Kay so if We leave a word out and generate a bunch of corpora, and We take those new corpora to model the real world, We expect that in there the words that will occur with frequency k, as in words that belong to the partition n_k in our original dataset occurred $(k+1)n_{k+1}$ times.

And then since We are looking at words that occurred k times, there are ah fuck

Okay so We will expect n_k words to be of frequency k in the future datasets as an absolute statement.

And then You get the expected count for each individual words out of the n_k words that will have frequency k by dividing by n_k. What a mess.

2.1.6 Okay so here is a Good-Turing Smoothing derivation

Start with some corpus with N tokens in it. The tokens may be n-grams for instance.

Call the complete corpus of all the tokens set C.

Now generate holdout corpora by taking one of the N tokens out at a time, which will yield N holdout sets, each with $N-1$ tokens in them.

Now think about the complete corpus - in that corpus, there may be 10 tokens that only occur once, as in if You removed that particular token, then that type of token does not appear in the corpus again, for example if "I am" is in there as an n-gram just once, then removing that token mean "I am" is no longer in the set at all anywhere else.

You can count the number of tokens that occur only once, and that will yield a number. Call that number n_1. Likewise, there will be a bunch of tokens that occur twice, like if "to go" is in the corpora twice. Count the number of tokens that appear twice (which means You are counting *types* of tokens), and You'll get n_2, the number of tokens in the corpus that occur twice. So on for n_k, the number of tokens that occur k times.

The idea then is that in the real world We will need to figure out what to do with tokens We've never seen before.

We obviously can't just simply reason about the unseen token - We've never seen it before, know nothing of it's potential frequency.

So the idea then is to take the complete corpus and drop out one token at a time to generate those holdout corpora, and come up for probabilities in the complete set in terms of tokens.

First are the tokens that won't appear at all in the holdout when held out - there will be n_1 of those, since for a token to be unseen after removal, it had to have had occurred just once in the complete corpus.

Now as a probability, We have N total tokens, and out of that total N, n_1 occurred just once. So to get a probability, We do n_1/KN.

Now for the general case, if We observe some token w times in a holdout corpus, then it had to have had occurred $w + 1$ times in the complete corpus. There are then by definition n_{w+1} such tokens in the complete corpus.

To get the probability of some token which belongs to the n_w partition, We just do

$$\frac{(w+1)n_{w+1}}{N} \tag{23}$$

Since out of all N tokens, $(w + 1)n_{w+1}$ will occur with frequency w in the holdout sets, which try to keep in mind that the holdout sets are trying to model our goal distribution.

Since that equation there gives You the fraction of tokens that will occur n_w times, then the expected count can be derived as follows: first, We will have n_w types of tokens that occur w times, as in "I am" may occur w times, "me too" can occur w times and so on.

We want probabilities for types of tokens in the complete corpus in terms of counts of tokens in the holdout corpora.

Anyway to get counts We just multiply the fraction We expect to have frequency w:

$$\frac{(w+1)n_{w+1}}{N} \times N = (w+1)n_{w+1} \tag{24}$$

But then there are n_w such tokens, by kind of definition, so We get

$$\frac{(w+1)n_{w+1}}{n_w} \tag{25}$$

As expected new count of tokens that occur w times lol.

I think the notation here is shit. You need pseudocounts. Which is what these are. This new expected count is a psuedocount, which it kind of has to be since We carved out space for tokens with frequency 0, so even though the no-shit-Sherlock answer to "how often do You expect a token which occurs w times to occur?" is w, now it's this thing above due to our meddling, hence a pseudocount. Or at least it's a pseudocount from the perspective of the holdout set.

2.1.7 One Last Good Time, I Swear

Cornell lecture here (click me).

Simulate the problem ahead of time by doing what precisely.

Well, holding stuff out, but how to do it such that it's representative?

Well it only makes sense to think about the distribution of something given that it's been held out, otherwise it's kinda the same as empirical observation.

I am struggling to envision what sample space We are dividing up here.

We hold out one token, then group the resulting holdout sets by the partitioning of the held out tokens, and look at the resulting distribution.

Okay there are two main parts here I think.

First is the partition assumption.

Second is the idea that if You observe a holdout set in the wild, that the counts of that holdout set are approximately equal to the complete set.

So, let n_r be frequency of frequencies in the complete set, let $\hat{n}_r$ be f.o.f. in the holdout set.

Now hold out a partition.

If the partition is missing, then it was n_1 in the complete set. n_1/N is the fraction of tokens, in terms of complete-set terms.

If, given that a token of this partition has been held out, it occurs k times in the holdout set, then it occurred $k+1$ times in the original set, and there are $(k+1)\times n_{k+1}$ such tokens in the original set. As a fraction it's $(k+1)\times n_{k+1}/N$.

It's bad intuition. The idea is then to say sure, We have come this far, but now that We have observed the actual holdout set, even though the "source" probability might be $(k+1)\times n_{k+1}/N$, We will now take this and divide it among the n_k token types that We have observed.

Rather, call $\hat{n}_k$ the number of items that have occurred k times in the holdout set, where We got to the holdout set by holding out one token from each type that belonged to n_{k+1} in the complete set. Basically $\hat{n}_k \approx n_{k+1}$.

So then the final equation is given by

$$\frac{(k+1)\times n_{k+1}}{N\times \hat{n}_k} \tag{26}$$

And then You say okay well that's approximately

$$\frac{(k+1)\times n_{k+1}}{N\times n_k} \tag{27}$$

Is the best I have.

2.1.8 Issues in Good-Turing

Jeez the more I stare at Good-Turing the shittier it seems.

So for one, it's not all that principled. It starts with this neat "hold one out" idea, and then just makes assumptions until You get a distribution out.

It's also ill defined since the adjusted count for the most frequent partition depends on an even more frequent partition, which does not exist.

And then to solve this they just throw all caution to the wind and fit a log-linear model and use that to fill in blanks, plus some random hacks like "oh of frequency is above 5 don't even worry lol.

2.1.9 Backoff and Interpolation

Backoff is using (n-1)-gram counts if n-gram counts are zero.

Interpolation is taking weighted sums of various n-gram probabilities such that the weights sum to 1.

You can an also have conditioned interpolation, where the weight factors change depending on context.

2.1.10 Katz Backoff

First equation is:

$$P_{\text{Katz}}(w_n|w_{n-N+1}^{n-1} = P^*(w_n|w_{n-N+1}^{n-1}) \tag{28}$$

if the n-gram count of the N-gram in question is non-zero, and otherwise We back off to

$$P_{\text{Katz}}(w_n|w_{n-N+1}^{n-1}) = \alpha(w_{n-N+1}^{n-1})P_{\text{katz}}(w_n|w_{n-N+2}^{n-1}) \tag{29}$$

So lots of little bits going on here.

First is P^*. The reason this is some new distribution and not just straight forward P which would result from n-gram counts, is that We are giving mass to things that P assigns zero mass to. That's kind of the whole point - We encountered an n-gram with zero count, which our naive approximation would give 0 mass to, so We use something else that gives non-zero mass, which means We have to take mass away from other stuff to keep everything summing to 1.

Next is the backoff case - to see this a little clearer maybe let's write

$$P_{\text{Katz}}(w|w_b^a) = \alpha(w_b^a)P_{\text{katz}}(w|w_{b+1}^a) \tag{30}$$

Basically just dropping a context word once a turn. Wild.

3 Working through what's already there

$$\sum_{\boldsymbol{c}} \Big(\mathrm{KL}\left(\tilde{p}(\cdot \mid \boldsymbol{c}) \,||\, p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c})\right) + \gamma(\boldsymbol{c}) \cdot \mathrm{KL}(u \,||\, p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c}) \Big) \tag{1}$$

So c is the context, I suppose N-gram wise.

Recall

$$\begin{aligned} KL(P, Q) &= \sum_{x \in \mathcal{X}} P(x) \cdot \big(\log(P(X)) - \log(Q(X)) \big) \qquad (2) \\ &= \sum_{x \in \mathcal{X}} P(x) \cdot \log \left(\frac{P(X)}{Q(X)} \right) \qquad (3) \end{aligned}$$

So the first divergence is between $\tilde{p}(\cdot \mid \boldsymbol{c})$ and $p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c})$, where I can only presume the first distribution is the true distribution and the second is the model distribution.

And then of course the true distribution just puts all the mass on the actually observed label so You just get negative log likelihood.

Then We have

$$\gamma(\boldsymbol{c}) = \frac{\lambda}{\#(\boldsymbol{c})} \tag{4}$$

Okay so this is a weird mix of stuff He's up to here.

He starts with the end.

The equation is label smoothing with that $\gamma(\boldsymbol{c})$ weighting factor, and He picks it such that minimum for the equation yields additive smoothing.

So anyway then He goes on to say well let's look at the probability of a word given a context:

$$p_{\boldsymbol{\theta}}(w|\boldsymbol{c}) \propto \frac{\#(\boldsymbol{c} \circ w)}{\#(\boldsymbol{c})} + \frac{\lambda}{\#(\boldsymbol{c})} \tag{5}$$

I kind of want to derive that though.

And derive it I did - see appendix.

4 Jelinek-Mercer

Idea is to interpolate between N-gram models.

First We have the standard MLE estimate for unigram:

$$p_{mle}(w_i) = \frac{c(w_i)}{\sum_{w_i} c(w_i)} \tag{1}$$

Or I guess to keep the notation consistent:

$$p_{mle}(w_i) = \frac{\#(w_i)}{\sum_{w_i} \#(w_i)} \tag{2}$$

Then for the interpolated bigram estimate We have

$$p_{inter}(w_i|w_{i-1}) = \lambda p_{mle}(w_i|w_{i-1}) + (1-\lambda)p_{mle}(w_i) \tag{3}$$

So a convex combination of the two maximum likelihood estimates.

And then recursively for an n-gram model:

$$p_{inter}(w_i|w_{i-n+1}) = \lambda_{w_{i-n+1}^{i-1}} p_{mle}(w_i|w_{i-n+1}^{i-1}) + (1-\lambda_{w_{i-n+1}^{i-1}})p_{inter}(w_i|w_{i-n+2}^{i-1}) \tag{4}$$

It's so nice and simple lol.

So first We have $\lambda_{w_{i-n+1}^{i-1}}$, so this lambda is conditioned on the previous $n-1$ words, so We can distinguish between $n^{|V|}$ different λ terms, oof.

But so given one of those λ terms We take the convex combination of the maximum likelihood estimate taking all the context into account, plus the interpolated probability with the context window shrunk by 1.

The recursion stops at either the unigram model, which can be smoothed using additive smoothing or something, or You can stop at a 0-th order model, which is just the uniform distribution over words.

And then how choose the λ terms and what does it all mean.

Well depending on the λ You weigh different n-gram models differently. I suppose the ones with the most frequency for this particular w_i should get more weight, since they'll be based on more data.

Either way, too many λ terms to tune. Bucket by

$$\sum_{w_i} c(w_{i-n+1}^{i}) \tag{5}$$

That doesn't really illuminate it for me so this seems better.

The paper says partitioning is done in accordance to $c(w_{i-n+1}^{i-1})$, which is just the number of times that that context has appeared. Contexts that have appeared the same number of times get the same λ. Aight.

So now what.

4.1 So now what

Well in ze previous derivation We started with a loss and went boiled it down to an expression in terms of counts.

Now We have counts, and I suppose We are looking to go the other way.

I guess I should try doing that on the existing examples to get a handle on things:

4.2 From count to log likelihood

Okay so We start with the n-gram formulation:

$$P(w_i) = \frac{\#(w_i)}{\sum_{w_i} \#(w_i)} \tag{6}$$

Then suppose that

$$-\lambda = \sum_{w_i} \#(w_i) \tag{7}$$

Why is this right?

Well, in both previous derivations, We used that

$$P(w_i) = \frac{\circ}{-\lambda} \tag{8}$$

Where $\circ$ is unknown for the time being. I guess the idea is that at some point that langrange multiplier will be on the left side of the equation, multiplying the desired probability, and We'll have to divide by it to get it outta there.

Well, is the $\circ$ unknown? Kinda feels known lol. If You have the n-gram formulation, then You have $\circ$.

So it's really

$$P(w_i) = \frac{\#(w_i)}{-\lambda} \tag{9}$$

Amazing progress.

Aight so then We do know that λ will be on it's own and that the whole thing will be equal to zero so

$$P(w_i) = \frac{\#(w_i)}{-\lambda} \tag{10}$$

$$\frac{\#(w_i)}{P(w_i)} + \lambda = 0 \tag{11}$$

$$\tag{12}$$

And then You know that λ will be be multiplying a simple constraint but does that actually give You anything?

You take the integral, You get

$$\#(w_i)\ln(P(w_i)) + \lambda P(w_i) \tag{13}$$

and then You know You got one of these for each derivative so then You sum and split out the constraint for clarity

$$\left(\sum_{i=1}^{m} \#(w_i)\ln(P(w_i))\right) + \lambda\left(\sum_{w_i} P(w_i) - 1\right) \tag{14}$$

Then I guess You divide by the count of a particular occurence to get individual occurences and look for $ln(P(w_i))$ terms for that KL between truth and model.

4.3 Additive Smoothing

I think I'll omit the context

$$P(w_i|\boldsymbol{c}) = \frac{\#(w_i,\boldsymbol{c}) + \alpha}{\sum_{i=1}^{|V|}\#(w_i,\boldsymbol{c}) + |V|\alpha} \tag{15}$$

Where $|V|$ is the size of the vocabulary.

So then let

$$-\lambda = \sum_{i=1}^{|V|}\#(w_i,\boldsymbol{c}) + |V|\alpha \tag{16}$$

So then

$$P(w_i|\boldsymbol{c}) = \frac{\#(w_i,\boldsymbol{c}) + \alpha}{\sum_{i=1}^{|V|}\#(w_i,\boldsymbol{c}) + |V|\alpha} \tag{17}$$

$$P(w_i|\boldsymbol{c}) = \frac{\#(w_i,\boldsymbol{c}) + \alpha}{-\lambda} \tag{18}$$

$$\frac{\#(w_i,\boldsymbol{c}) + \alpha}{P(w_i|\boldsymbol{c})} + \lambda = 0 \tag{19}$$

$$\tag{20}$$

Then by the integral We get

$$\left(\sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c}) + \alpha) \ln(P(w_i|\boldsymbol{c}) \right) + \lambda \left(\sum_{i=1}^{|V|} P(w_i|\boldsymbol{c}) \right) \tag{21}$$

Then We can ignore the constraint

$$\left(\sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c}) + \alpha) \ln(P(w_i|\boldsymbol{c}) \right) + \lambda \left(\sum_{i=1}^{|V|} P(w_i|\boldsymbol{c}) \right) \tag{22}$$

$$\left(\sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c}) + \alpha) \ln(P(w_i|\boldsymbol{c}) \right) \tag{23}$$

$$\left(\sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c})) \ln(P(w_i|\boldsymbol{c}) + \alpha \ln(P(w_i|\boldsymbol{c}) \right) \tag{24}$$

And that is the total log probability. Then to focus on a particular word We focus on a particular i:

$$\left(\sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c})) \ln(P(w_i|\boldsymbol{c}) + \alpha \ln(P(w_i|\boldsymbol{c}) \right) \tag{25}$$

$$(\#(w_i, \boldsymbol{c})) \ln(P(w_i|\boldsymbol{c}) + \alpha \ln(P(w_i|\boldsymbol{c}) \tag{26}$$

If that is the log likelihood then that count term is counting duplicates so

$$(\#(w_i, \boldsymbol{c})) \ln(P(w_i|\boldsymbol{c}) + \alpha \ln(P(w_i|\boldsymbol{c}) \tag{27}$$

$$\ln(P(w_i|\boldsymbol{c}) + \frac{\alpha}{(\#(w_i, \boldsymbol{c}))} \ln(P(w_i|\boldsymbol{c}) \tag{28}$$

$$\tag{29}$$

Then add a constant which I guess We have floating around lol

$$\ln(P(w_i|\boldsymbol{c}) + \frac{\alpha}{(\#(w_i, \boldsymbol{c}))} \ln(P(w_i|\boldsymbol{c}) \tag{30}$$

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\alpha}{(\#(w_i, \boldsymbol{c}))} \ln\left(\frac{1}{P(w_i|\boldsymbol{c})} \right) \tag{31}$$

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\alpha}{(\#(w_i, \boldsymbol{c}))} \ln\left(\frac{1}{P(w_i|\boldsymbol{c})} \right) + \ln\left(\frac{\alpha}{(\#(w_i, \boldsymbol{c}))} \right) \tag{32}$$

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\alpha}{(\#(w_i, \boldsymbol{c}))} \ln\left(\frac{\alpha/(\#(w_i, \boldsymbol{c}))}{P(w_i|\boldsymbol{c})} \right) \tag{33}$$

So now the first term is then of course cross entropy or KL w.r.t. original distribution and the second, well, the second is dodgier.

Depending on the value of α it's the uniform distribution?

What went wrong - did something go wrong?

Yeah, there should be a summation across all labels in the loss for any particular label.

Okay so here

$$\left(\sum_{i=1}^{|V|}(\#(w_i,\boldsymbol{c})+\alpha)\ln(P(w_i|\boldsymbol{c})\right)+\lambda\left(\sum_{i=1}^{|V|}P(w_i|\boldsymbol{c})\right) \tag{34}$$

Hmm. I feel like this is where We should have the fact that there is the same number of all-word terms for each word.

I guess the information is in there, still, it's just that We lost what α was? 'cause it wasn't just random.

$$\sum_{y_i\in\mathcal{Y}}\#(y_i,\boldsymbol{c})+\#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)=\lambda \tag{35}$$

Is what it was, and

$$-\lambda=\sum_{i=1}^{|V|}\#(w_i,\boldsymbol{c})+|V|\alpha \tag{36}$$

Whoop I think I found the error.

So We had

$$\left(\sum_{i=1}^{|V|}(\#(w_i,\boldsymbol{c}))\ln(P(w_i|\boldsymbol{c})+\alpha\ln(P(w_i|\boldsymbol{c})\right) \tag{37}$$

Okay so that was not the mistake.

So what happens if We set

$$\gamma(\boldsymbol{c})=\frac{|V|\alpha}{\#(\boldsymbol{c})} \tag{38}$$

$$\gamma(\boldsymbol{c})\#(\boldsymbol{c})=|V|\alpha \tag{39}$$

$$\frac{\gamma(\boldsymbol{c})\#(\boldsymbol{c})}{|V|}=\alpha \tag{40}$$

Plugging that in

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\alpha}{(\#(w_i,\boldsymbol{c}))}\ln\left(\frac{\alpha/(\#(w_i,\boldsymbol{c}))}{P(w_i|\boldsymbol{c})}\right) \tag{41}$$

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\frac{\gamma(\boldsymbol{c})\#(\boldsymbol{c})}{|V|}}{(\#(w_i,\boldsymbol{c}))}\ln\left(\frac{\frac{\gamma(\boldsymbol{c})\#(\boldsymbol{c})}{|V|}/(\#(w_i,\boldsymbol{c}))}{P(w_i|\boldsymbol{c})}\right) \tag{42}$$

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\gamma(\boldsymbol{c})\#(\boldsymbol{c})}{|V|(\#(w_i,\boldsymbol{c}))}\ln\left(\frac{\gamma(\boldsymbol{c})\#(\boldsymbol{c})/|V|(\#(w_i,\boldsymbol{c}))}{P(w_i|c)}\right) \tag{43}$$

Okay so I think something definitely gone wrong since I don't think I'm wait a god damn minute

I'm getting the uniform distribution.... Over data items as opposed to labels?

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\alpha}{(\#(w_i,\boldsymbol{c}))}\ln\left(\frac{\alpha/(\#(w_i,\boldsymbol{c}))}{P(w_i|c)}\right) \tag{44}$$

Okay now let's think about summing this over all data items. Each data item..

This is weird because it's only focusing on the correct label.

Can We scrounge together a uniform distribution if We look at the summation over all data items?

Yeah okay

If We sum over all samples of a particular word, We'll get

Okay so the idea at the edge of my mind is that the reason each term is normalized by $(\#(w_i, \boldsymbol{c})$ is so that when We add together duplicates, they'll get the same share of the uniform distribution, regardless of how many duplicates there are.

That checks out.

Maybe fix it here? Instead of doing that

$$\ln(P(w_i|\boldsymbol{c}) + \frac{\alpha}{(\#(w_i,\boldsymbol{c}))}\ln(P(w_i|\boldsymbol{c}) \tag{45}$$

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\alpha}{(\#(w_i,\boldsymbol{c}))}\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right) \tag{46}$$

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\alpha}{(\#(w_i,\boldsymbol{c}))}\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right) + \ln\left(\frac{\alpha}{(\#(w_i,\boldsymbol{c}))}\right) \tag{47}$$

$$\ln(P(w_i|\boldsymbol{c}) - \frac{\alpha}{(\#(w_i,\boldsymbol{c}))}\ln\left(\frac{\alpha/(\#(w_i,\boldsymbol{c}))}{P(w_i|\boldsymbol{c})}\right) \tag{48}$$

Instead do

$$\ln(P(w_i|\boldsymbol{c}) + \frac{\alpha}{(\#(w_i, \boldsymbol{c}))} \ln(P(w_i|\boldsymbol{c}) \tag{49}$$

$$\ln(P(w_i|\boldsymbol{c}) + \alpha\frac{1}{(\#(w_i, \boldsymbol{c}))} \ln(P(w_i|\boldsymbol{c}) \tag{50}$$

$$\ln(P(w_i|\boldsymbol{c}) - \alpha\frac{1}{(\#(w_i, \boldsymbol{c}))} \ln(\frac{1}{P(w_i|\boldsymbol{c}}) \tag{51}$$

$$\ln(P(w_i|\boldsymbol{c}) - \alpha\frac{1}{(\#(w_i, \boldsymbol{c}))} \ln(\frac{1}{P(w_i|\boldsymbol{c}}) + \alpha\frac{1}{(\#(w_i, \boldsymbol{c}))} \ln(1/\#(w_i, \boldsymbol{c})) \tag{52}$$

$$\ln(P(w_i|\boldsymbol{c}) - \alpha\frac{1}{(\#(w_i, \boldsymbol{c}))} \ln\left(\frac{1/(\#(w_i, \boldsymbol{c}))}{P(w_i|\boldsymbol{c}}\right) \tag{53}$$

Sorta, but it doesn't address missing labels. Not a fan.

4.3.1 Mistakes

Alright so is this line correct:

$$\left(\sum_{i=1}^{|V|}(\#(w_i, \boldsymbol{c}) + \alpha)\ln(P(w_i|\boldsymbol{c})\right) + \lambda\left(\sum_{i=1}^{|V|} P(w_i|\boldsymbol{c}) - 1\right) \tag{54}$$

In the original derivation it was

$$\left(\sum_{y_i \in \mathcal{Y}}(\#(y_i, \boldsymbol{c})) \cdot \ln\left(\frac{1}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right)\right) \tag{55}$$

$$+ \#(\boldsymbol{c})\gamma(\boldsymbol{c})\left(\sum_{y_k \in \mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c}))\right) \tag{56}$$

$$+ \left(\lambda \sum_{y_l \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y_l|\boldsymbol{c}) - 1\right) \tag{57}$$

The signs are wonky, because I messed up the λ sign.

$$\left(\sum_{i=1}^{|V|} -(\#(w_i, \boldsymbol{c}) + \alpha)\ln(P(w_i|\boldsymbol{c})\right) + \lambda\left(\sum_{i=1}^{|V|} P(w_i|\boldsymbol{c}) - 1\right) \tag{58}$$

$$\left(\sum_{i=1}^{|V|}(\#(w_i, \boldsymbol{c}) + \alpha)\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right)\right) + \lambda\left(\sum_{i=1}^{|V|} P(w_i|\boldsymbol{c}) - 1\right) \tag{59}$$

Is what it would be if it were correct, I think lol.
Then drop the constraint 'cause who cares

$$\left(\sum_{i=1}^{|V|}(\#(w_i,\boldsymbol{c})+\alpha)\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right)\right) \tag{60}$$

$$\left(\sum_{i=1}^{|V|}\#(w_i,\boldsymbol{c})\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right)\right)+\left(\sum_{i=1}^{|V|}\alpha\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right)\right) \tag{61}$$

And now I suppose We try to do splitting by datapoint?

One datapoint gets one cross entropy term, which is okay.

Then We need to think about α lol. Are there any constraint for what it should be set as? It just has to be above zero, so We can introduce any constant We'd like, for example

$$\alpha=\frac{1}{|V|}\#(\boldsymbol{c})\beta \tag{62}$$

Then

$$\sum_{i=1}^{|V|}\alpha\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right)\Bigg) \tag{63}$$

$$\sum_{i=1}^{|V|}\frac{1}{|V|}\#(\boldsymbol{c})\beta\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right)\Bigg) \tag{64}$$

$$\tag{65}$$

And now You need to think carefully about how to distribute this. You also need to add a constant lol The constant is:

$$\sum_{i=1}^{|V|}\frac{1}{|V|}\#(\boldsymbol{c})\beta\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right)\Bigg) \tag{66}$$

$$\left(\sum_{i=1}^{|V|}-\frac{1}{|V|}\#(\boldsymbol{c})\beta\ln(P(w_i|\boldsymbol{c}))\right)+\sum_{i=1}^{|V|}\frac{1}{|V|}\#(\boldsymbol{c})\beta\ln\left(\frac{1}{|V|}\right) \tag{67}$$

$$\left(\sum_{i=1}^{|V|}-\frac{1}{|V|}\#(\boldsymbol{c})\beta\ln\left(\frac{1/|V|}{P(w_i|\boldsymbol{c})}\right)\right) \tag{68}$$

$$\tag{69}$$

And so now You use the count term to distribute this across all data items, split out the β and for each data item then We get

$$\ln\left(\frac{1}{P(w_i|\boldsymbol{c})}\right) + \beta\sum_{i=1}^{|V|}\frac{1}{|V|}\ln\left(\frac{1/|V|}{P(w_i|\boldsymbol{c})}\right) \tag{70}$$

$$\tag{71}$$

I think that's that.

4.4 Deriving it I suppose

So, We had

$$p_{inter}(w_i|w_{i-n+1}) = \lambda_{w_{i-n+1}^{i-1}}p_{mle}(w_i|w_{i-n+1}^{i-1}) + (1-\lambda_{w_{i-n+1}^{i-1}})p_{inter}(w_i|w_{i-n+2}^{i-1}) \tag{72}$$

I guess I should just start with an interpolated bigram and see how I get on. I mean, it's just a sum of two MLE estimates:

$$p_{inter}(w_i|w_{i-1}) = \alpha p_{mle}(w_i|w_{i-1}) + (1-\alpha)p_{mle}(w_i) \tag{73}$$

I feel like You can't just make the claim that KL divergence works here, but it's worth a shot. The objective then is:

$$\sum_{n=1}^{2}\alpha_n \text{KL}(\tilde{p}(\cdot|\boldsymbol{c}_{<n}) \,||\, p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{c}_{<2})) \tag{74}$$

Which kind of makes You go "hmm".

I suppose the trivial thing to do would be to just train separate n-gram models and ensamble them. I guess they are not looking for that though lol.

So for the ground truth We take the unigram and bigram observed samples, and in both cases We try to make our distribution more like those. Aight.

Does that make sense to do? On the one hand We have count of contexts over count of instances, on the other yeah just some model. It might have access to the previous 2 tokens okay

So given an n gram model, it the divergences between short n-gram counts that act as smoothing here I suppose. Aight.

So then to peel away the KL for a particular observed token $(w_j, \boldsymbol{c}_{<n})$:

$$\alpha_n \text{KL}(\tilde{p}(\cdot|\boldsymbol{c}_{<n}) \,||\, p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{c}_{<2})) \tag{75}$$

$$\alpha\sum_{i=1}^{|V|}\tilde{p}(w_i|\boldsymbol{c}_{<n})\ln\left(\frac{\tilde{p}(w_i|\boldsymbol{c}_{<n})}{p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}_{<2})}\right) \tag{76}$$

$$\alpha\ln\left(\frac{1}{p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}_{<2})}\right) \tag{77}$$

So the point again is for a particular observed token, only one head word matches, the rest have probability zero.

Summing over the different n-grams and the entire corpus then yields:

$$\sum_{n=1}^{2} \alpha_n \mathrm{KL}(\tilde{p}(\cdot|\boldsymbol{c}_{<n}) \mid\mid p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{c}_{<2})) \tag{78}$$

$$= \sum_{n=1}^{2} \sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c}_{<n}) \alpha_n \mathrm{KL}(\tilde{p}(\cdot|\boldsymbol{c}_{<n}) \mid\mid p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{c}_{<2})) \tag{79}$$

$$= \sum_{n=1}^{2} \sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c}_{<n}) \alpha_n \ln\left(\frac{1}{p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}_{<2})}\right) \tag{80}$$

Okay so We sum over both n-gram cases, and then sum over all words in the vocabulary. That's fine.

Then for each word in the vocabulary We'll have the number of times it occurs with the appropriate length context. Does that make sense? Well as We sum over all the words, We'll go over each token once this way, so that squares.

Then the appropriate α for the n count and We're good, I think.

Should the indexing be simplified? Seems like maybe so, or at least I think it makes sense to move the outer sum in.

$$= \sum_{n=1}^{2} \sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c}_{<n}) \alpha_n \ln\left(\frac{1}{p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}_{<2})}\right) \tag{81}$$

$$= \sum_{i=1}^{|V|} \sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n}) \alpha_n \ln\left(\frac{1}{p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}_{<2})}\right) \tag{82}$$

So then I suppose it's on to Lagrange

$$\mathcal{L} = \left(\sum_{i=1}^{|V|} \sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n}) \alpha_n \ln\left(\frac{1}{p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}_{<2})}\right)\right) + \lambda\left(\sum_{j=1}^{|V|} p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}{<}\, 2) - 1\right) \tag{83}$$

$$\mathcal{L} = \left(\sum_{i=1}^{|V|} \sum_{n=1}^{2} -(\#(w_i, \boldsymbol{c}_{<n}) \alpha_n \ln(p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}_{<2}))\right) + \lambda\left(\sum_{j=1}^{|V|} p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}{<}\, 2) - 1\right) \tag{84}$$

Then derivative w.r.t. some $p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2})$

$$\sum_{n=1}^{2} -(\#(w_i, \boldsymbol{c}_{<n})\alpha_n \frac{1}{p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2})} + \lambda = 0 \tag{85}$$

$$\sum_{n=1}^{2} -(\#(w_i, \boldsymbol{c}_{<n})\alpha_n \frac{1}{p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2})} = -\lambda \tag{86}$$

$$\frac{\sum_{n=1}^{2} -(\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2})} = -\lambda \tag{87}$$

$$\frac{p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2})}{\sum_{n=1}^{2} -(\#(w_i, \boldsymbol{c}_{<n})\alpha_n} = -\frac{1}{\lambda} \tag{88}$$

$$p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2}) = -\frac{\sum_{n=1}^{2} -(\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{\lambda} \tag{89}$$

Then back to the constraint:

$$\sum_{i=1}^{|V|} p_{\boldsymbol{\theta}}(w_i|\boldsymbol{c}_{<2}) = 1 \tag{90}$$

$$\sum_{i=1}^{|V|} -\frac{\sum_{n=1}^{2} -(\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{\lambda} = 1 \tag{91}$$

$$-\frac{\sum_{i=1}^{|V|}\sum_{n=1}^{2} -(\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{\lambda} = 1 \tag{92}$$

$$\sum_{i=1}^{|V|}\sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n = \lambda \tag{93}$$

Then back again:

$$p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2}) = \frac{\sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{\lambda} \tag{94}$$

$$p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2}) = \frac{\sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{\sum_{i=1}^{|V|}\sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n} \tag{95}$$

And then it's just simplification I suppose.

$$\begin{aligned} p_{\boldsymbol{\theta}}(w_m|\boldsymbol{c}_{<2}) &= \frac{\sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{\sum_{i=1}^{|V|}\sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n} && (96) \\ &= \frac{\sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{\sum_{n=1}^{2}\sum_{i=1}^{|V|} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n} && (97) \\ &= \frac{\sum_{n=1}^{2} (\#(w_i, \boldsymbol{c}_{<n})\alpha_n}{\sum_{n=1}^{2} (\#(\boldsymbol{c}_{<n})\alpha_n} && (98) \end{aligned}$$

5 Appendix

5.1 Entropy

Oooookay.

This is a simple one - the entropy of a random variable is the expected value of information gained by observing a random variable outcome. How much info do We gain on average by an instance of the variable.

$$Ent(X) = -\sum_{x \in \mathcal{X}} p(x) \cdot log(p(x)) \tag{1}$$

5.2 Cross Entropy

The difference here is that I think We are measuring the amount of information We gain given that events are in reality generated by some distribution p, but We are modeling the events with the distribution q.

Or, according to A Probabilistic Perspective, if We model some distribution p with our own version q, it's the number of bits We'll need on average to represent an event. It's strictly positive is the interesting thing, as in if Your model deviates at all from the truth, You goofed.

$$CE(P, Q) = -\sum_{x \in \mathcal{X}} P(x) \cdot log(Q(x)) \tag{2}$$

Now here comes a connection with KL divergence:

5.3 KL Divergence

Measures the difference in information between two distributions, with p assumed to be the true one.

$$\begin{align} KL(P, Q) &= \sum_{x \in \mathcal{X}} P(x) \cdot \big(\log(P(X)) - \log(Q(X)) \big) \tag{3} \\ &= \sum_{x \in \mathcal{X}} P(x) \cdot \log\left(\frac{P(X)}{Q(X)}\right) \tag{4} \end{align}$$

So on average how many more bits do You need to encode an event drawn from distribution P if You are modelling the distribution with Q.

The connection with cross entropy seems pretty clear - cross entropy measures the total number of bits needed to encode events from P given Q, and KL measures the *additional* bits needed.

And so cross entropy in terms of KL divergence also makes sense - cross entropy keeps track of the number of bits needed to represent event from P using Q, and that is of course just the amount of information needed to represent event from P, which is just the entropy of P, plus the additional bits We need due to shitty encoding, which is precisely the KL divergence, so

$$CE(P,Q) = Ent(P) + KL(P,Q) \tag{5}$$

5.4 Perplexity

5.5 As a measure of model confusion

A.k.a. how perplexed is the model.

So We take a sentence from the test set, calculate it's probability which is probably a product of n-gram terms, then take the log, and the normalize the log probability by dividing through by N. Makes sense, don't want longer sentences to be unlikelier just due to length.

And then... We take the exponent to get rid of the log? Seems ass backwards.

$$\begin{aligned} e^{\frac{1}{N}\cdot\sum_{i=1}^{N}\ln(P(w_i))} &= \left(e^{\sum_{i=1}^{N}\ln(P(w_i))}\right)^{1/N} \tag{6}\\ &= \left(\prod_{i=1}^{N} e^{ln(P(w_i))}\right)^{1/N} \tag{7}\\ &= \left(\prod_{i=1}^{N} P(w_i)\right)^{1/N} \tag{8} \end{aligned}$$

And so now that We have this normalized probability which I suppose We took a detour into logs for the intuition, now We take the inverse of this.

Inverse because the larger the probability, the less the model should be confused. I guess You could also report the certainty of the model lol.

Want to know the confusion? Take the base You calculated entropy with and exponentiate it with that base - this gives You the number of equally likely options the model had to consider. It's neat.

5.6 Lagrange Multipliers

Here We go again.

There is only one real core piece of intuition here and it's about the direction of the gradient - suppose You have $x \in \mathcal{X}$ as Your range of possible values, and You're looking to maximize $f(x)$ such that $g(x) = 0$, so g is the function expressing the constraint.

Simple constraint is good old fashioned $x_1^2 + x_2^2 = r$ for a circle in 2D.

So first You'll only be considering values of x that satisfy $g(x) = 0$, which will narrow down Your options.

Now take the gradient of the constraint - which means the constraint has to be differentiable, which means You can kind of imagine it's shape. Introducing some kind of $= c$ in the constraint just locks down the value of $g(x) = c$, as in We just have to focus on some level curve. Then of course the gradient is perpendicular to the level curve.

Then all You have to do is find x such that $f'(x) = \lambda g'(x)$ and $g(x) = 0$, skipping a lot of stuff here.

If the gradients aren't parallel, then You can take a step such that You stay on the level curve of g and increase f.

5.7 Unigram MLE

Find the original resource here.

Let S (for sentence) be the random variable of size n and let $\mathbf{s}$ be a vector of size n which contains actual observations, $\mathbf{s}_i \in \{w_1 \dots w_m\}$, so each observed word can be on of m words.

By the product rule then

$$\begin{aligned} P(S = \mathbf{s}) &= P(S_1 = \mathbf{s}_1, S_2 = \mathbf{s}_2 \dots S_n = \mathbf{s}_n) && (9) \\ &= P(S_1 = \mathbf{s}_1) \times P(S_2 = \mathbf{s}_2 | S_1 = \mathbf{s}_1) \times P(S_3 = \mathbf{s}_3 | S_1 = \mathbf{s}_1, S_2 = \mathbf{s}_2)... && (10) \end{aligned}$$

Then by the unigram assumption

$$\begin{aligned} P(S = \mathbf{s}) &= P(S_1 = \mathbf{s}_1) \times P(S_2 = \mathbf{s}_2 | S_1 = \mathbf{s}_1) \times P(S_3 = \mathbf{s}_3 | S_1 = \mathbf{s}_1, S_2 = \mathbf{s}_2)... && (11) \\ &= P(S_1 = \mathbf{s}_1) \times P(S_2 = \mathbf{s}_2) \times P(S_3 = \mathbf{s}_3) \dots \times P(S_n = \mathbf{s}_n) && (12) \end{aligned}$$

Now of course these become position independent and $P(S_i = \mathbf{s}_i) = P(S_j = \mathbf{s}_j)$ if $\mathbf{s}_i = \mathbf{s}_j$. Let $P(\mathbf{s}_i) = P(w_k)$, then We can group the terms as follows:

$$\begin{aligned} P(S = \mathbf{s}) &= P(S_1 = \mathbf{s}_1) \times P(S_2 = \mathbf{s}_2) \times P(S_3 = \mathbf{s}_3) \dots \times P(S_n = \mathbf{s}_n) && (13) \\ &= \prod_{i=1}^{m} P(w_i)^{\# w_i} && (14) \end{aligned}$$

Where

$$\# w_i = \sum_{l=1}^{n} \{1 | \mathbf{s}_l = w_i\} \tag{15}$$

So just the number of times w_i occurs in the sentence.

But so okay We have a likelihood, and then for the log likelihood:

$$P(S = \mathbf{s}) = \prod_{i=1}^{m} P(w_i)^{\#w_i} \tag{16}$$

$$\ln(P(S = \mathbf{s})) = \ln\left(\prod_{i=1}^{m} P(w_i)^{\#w_i}\right) \tag{17}$$

$$= \sum_{i=1}^{m} (\#w_i) \ln(P(w_i)) \tag{18}$$

$$\tag{19}$$

And since ln is monotonous and We just care about the maximum, We can use take the log.

Then We use Lagrange multipliers along with the constraint that

$$\sum_{i=1}^{m} P(w_i) = 1 \tag{20}$$

$$\sum_{i=1}^{m} P(w_i) - 1 = 0 \tag{21}$$

So then We are looking for the max of

$$\mathcal{L} = \sum_{i=1}^{m} (\#w_i) \ln(P(w_i)) + \lambda\Bigg(\sum_{i=1}^{m} P(w_i) - 1\Bigg) \tag{22}$$

So then We know

$$\frac{\partial \mathcal{L}}{\partial P(w_i)} = \frac{\partial}{\partial P(w_i)} \sum_{j=1}^{m} (\#w_j) \ln(P(w_j)) + \lambda\Bigg(\sum_{k=1}^{m} P(w_k) - 1\Bigg) = 0 \tag{23}$$

$$\frac{\#w_i}{P(w_i)} + \lambda = 0 \tag{24}$$

$$P(w_i) = -\frac{\#w_i}{\lambda} \tag{25}$$

And then the derivatives. The count is a constant w.r.t. the probability assigned - should really think of the probability as P_θ or something.

We are also taking the derivative w.r.t. a particular unigram - what value should the probability of this unigram be set to to maximize probability etc.

So We have an expression which maximizes **L**, now We use the constraint

$$\sum_{i=1}^{m} P(w_i) = 1 \tag{26}$$

$$\sum_{i=1}^{m} -\frac{\#w_i}{\lambda} = 1 \tag{27}$$

$$\frac{1}{\lambda}\sum_{i=1}^{m} -\#w_i = 1 \tag{28}$$

$$\lambda = -\sum_{i=1}^{m} \#w_i \tag{29}$$

and finally plugging this back in:

$$P(w_i) = -\frac{\#w_i}{\lambda} \tag{30}$$

$$P(w_i) = -\frac{\#w_i}{-\sum_{i=1}^{m} \#w_i} \tag{31}$$

$$P(w_i) = \frac{\#w_i}{\sum_{i=1}^{m} \#w_i} \tag{32}$$

Is the MLE. Wee.

5.8 Bigram MLE

Okay so weird:

$$p(\mathbf{w}) = \prod_{i=1}^{n} p(w_i)^{s(w_i)} \prod_{i=1}^{n}\prod_{j=1}^{n} p(w_j|w_i)^{c(w_i,w_j)} \tag{33}$$

So what in tarnation is that

Okay so first We are going through this word by word, so We are taking each words contributions to bigrams.

The index i is shared, and $s(w_i) = 1$ if w_i is the first word in a bigram.

I mean I can probably derive this mess.

You'll have a product of a bunch of bigram probabilities, and We can further decompose those bigram probabilities into unigram probabilities and conditional probabilities by the product rule.

Actually, I feel like this isn't even necessary as a derivation - to see that bigram MLE for some (w_i, w_j) is

$$\frac{c(w_i, w_j)}{\sum_{j=1}^{n} c(w_i, w_j)} \tag{34}$$

Just observe that this is really a bunch of unigram problems, since You are just looking for ways to distribute the mass of the second word given the first word has happened.

5.9 Unigram MLE with Uniform distribution KL

Kay so now We go all the way back:

$$\sum_{\boldsymbol{c}} \Big(\mathrm{KL}\left(\tilde{p}(\cdot \mid \boldsymbol{c}) \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c})\right) + \gamma(\boldsymbol{c}) \cdot \mathrm{KL}(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c})) \Big) \tag{35}$$

Okay so We are summing over contexts predicting labels, equivalent to any old thing really since it's predicting label given data.

$$\mathrm{KL}(P(\hat{y}|\boldsymbol{c}), P_\theta(\hat{y}|\boldsymbol{c}) \tag{36}$$

$$\sum_{y \in \mathcal{Y}} P(\hat{y}|\boldsymbol{c}) \cdot \ln\left(\frac{1}{P_\theta(\hat{y}|\boldsymbol{c})}\right) \tag{37}$$

$$-\sum_{y \in \mathcal{Y}} P(\hat{y}|\boldsymbol{c}) \cdot \ln(P_\theta(\hat{y}|\boldsymbol{c})) \tag{38}$$

$$-\ln(P_\theta(\hat{y} = y_{true}|\boldsymbol{c})) \tag{39}$$

Where P is the true probability and P_θ is the estimated probability.

Basically We have negative log likelihood and the only probability that gets optimized (though due to unity constraints they all do I suppose) is the probability of the true label.

I guess We again multiply by minus in order to turn this into a maximization problem and We arrive back at Unigram MLE.

Now for the smoothing term:

$$\mathrm{KL}(u, P_\theta(\hat{y}|\boldsymbol{c}) \tag{40}$$

$$\sum_{y \in \mathcal{Y}} \frac{1}{|\mathcal{Y}|} \cdot \ln\left(\frac{1/|\mathcal{Y}}{P_\theta(\hat{y})}\right) \tag{41}$$

$$\frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} \ln(1/|\mathcal{Y}|) - \ln(P_\theta(\hat{y})) \tag{42}$$

$$\left(\frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} \ln(1/|\mathcal{Y}|)\right) + \frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} - \ln(P_\theta(\hat{y})) \tag{43}$$

$$-\frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} \ln(P_\theta(\hat{y})) \tag{44}$$

$$\tag{45}$$

And this is all correct, as far as I can tell. Again We multiply by minus to turn this into a maximization problem. Though I guess We don't need to.

Anyway then Your loss is

$$\mathcal{L} = \ln(P_\theta(\hat{y} = y_{true}|\boldsymbol{c})) + \bigg(\frac{1}{|\mathcal{Y}|}\sum_{y\in\mathcal{Y}} \ln(P_\theta(\hat{y}|\boldsymbol{c}))\bigg) + \lambda\bigg(\sum_{y\in\mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) - 1\bigg) \tag{46}$$

Except for one bit - multiples of the same context. If We are going to sum over possible contexts instead of instances of contexts, We need to account for the possibility of duplicate occurences, i.e. if "I am well" occurs more than once, and if the summation only counts that occurrence once, then it's undercounting.

And I think the count term applies to all the terms in the sum, since if You think about it, the way We arrived at the above equations is that We made the assumption that the "true" distribution was a mixture of a dirac delta and a univariate distribution, and this assumption will apply to every term in the summation equally, hence counting. This is difficult to follow.

$$\mathcal{L} = \tag{47}$$

$$(\#(\boldsymbol{c}, y_{true}))\bigg(\ln(p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})) + \bigg(\frac{1}{|\mathcal{Y}|}\sum_{y\in\mathcal{Y}} \ln(P_\theta(\hat{y}|\boldsymbol{c}))\bigg)\bigg) \tag{48}$$

$$+ \lambda\bigg(\sum_{y\in\mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) - 1\bigg) \tag{49}$$

And now I suppose just the derivative w.r.t. $p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})$

$$\frac{\partial}{\partial p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})}(\#(\boldsymbol{c}, y_{true}))\bigg(\ln(p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})) + \bigg(\frac{1}{|\mathcal{Y}|}\sum_{y\in\mathcal{Y}} \ln(P_\theta(\hat{y}|\boldsymbol{c}))\bigg)\bigg) \tag{50}$$

$$(\#(\boldsymbol{c}, y_{true}))\bigg(\frac{1}{p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c}))} + \bigg(\frac{1}{p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})) \cdot |\mathcal{Y}|)}\bigg)\bigg) \tag{51}$$

$$(\#(\boldsymbol{c}, y_{true}))\bigg(\frac{1 + |\mathcal{Y}|}{p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})) \cdot |\mathcal{Y}|)}\bigg) \tag{52}$$

$$\frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})) \cdot |\mathcal{Y}|} \tag{53}$$

And then the other term:

$$\frac{\partial}{\partial p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})} + \lambda\bigg(\sum_{y\in\mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) - 1\bigg) = \lambda \tag{54}$$

lol

So then in total We get

$$\frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c})) \cdot |\mathcal{Y}|} + \lambda = 0 \tag{55}$$

$$(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|) = -p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c}) \cdot |\mathcal{Y}| \cdot \lambda \tag{56}$$

$$\frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{-|\mathcal{Y}| \cdot \lambda} = p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c}) \tag{57}$$

Alright.
Then We plug that into the constraint:

$$\sum_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) = 1 \tag{58}$$

$$\sum_{y \in \mathcal{Y}} \frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{-|\mathcal{Y}| \cdot \lambda} = 1 \tag{59}$$

$$\sum_{y \in \mathcal{Y}} \frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{-|\mathcal{Y}|} = \lambda \tag{60}$$

oof.

$$p_{\boldsymbol{\theta}}(\hat{y} = y_{true}|\boldsymbol{c}) = \frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{-|\mathcal{Y}| \cdot \lambda} \tag{61}$$

$$= \frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{-|\mathcal{Y}| \cdot \sum_{y \in \mathcal{Y}} \frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{-|\mathcal{Y}|}} \tag{62}$$

$$= \frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{\sum_{y \in \mathcal{Y}} (\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)} \tag{63}$$

$$= \frac{(\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)}{\sum_{y \in \mathcal{Y}} (\#(\boldsymbol{c}, y_{true})) \cdot (1 + |\mathcal{Y}|)} \tag{64}$$

$$= \frac{(\#\boldsymbol{c}, y_{true})}{\sum_{y \in \mathcal{Y}} \#(\boldsymbol{c}, y)} \tag{65}$$

$$\tag{66}$$

Uhh.
Something's gone wrong here. If I take the KL divergence of just the univariate part, surely I'll be able to get the Univariate distribution?

5.9.1 Sanity Check

$$\sum_{\boldsymbol{c}} \Big(\mathrm{KL}\left(\tilde{p}(\cdot \mid \boldsymbol{c}) \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c})\right) + \gamma(\boldsymbol{c}) \cdot \mathrm{KL}(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c}) \Big) \tag{67}$$

And again, KL divergence is:

$$\mathrm{KL}(P, Q) = \sum_{x \in \mathcal{X}} P(x) \cdot \ln\left(\frac{P(x)}{Q(x)}\right) \tag{68}$$

So the red term is

$$\begin{align}
\sum_{\boldsymbol{c}} \gamma(\boldsymbol{c}) \cdot \mathrm{KL}(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c}) &= \gamma(\boldsymbol{c}) \cdot \sum u(x) \cdot \ln\left(\frac{u(x)}{p_{\boldsymbol{\theta}}(x|\boldsymbol{c})}\right) \tag{69}\\
&= \gamma(\boldsymbol{c}) \cdot \sum u(x) \cdot \ln(u(x)) - u(x) \cdot \ln(p_{\boldsymbol{\theta}}(x|\boldsymbol{c})) \tag{70}\\
&= \gamma(\boldsymbol{c}) \cdot \sum -\big(- u(x) \cdot \ln(u(x))\big) - u(x) \cdot \ln(p_{\boldsymbol{\theta}}(x|\boldsymbol{c})) \tag{71}\\
&= \gamma(\boldsymbol{c}) \cdot \sum -Ent(u) + Cross(u, p_{\boldsymbol{\theta}}) \tag{72}
\end{align}$$

Where *Ent* is the entropy and *Cross* is the cross entropy. Sure, this all makes sense.

So then what function are We trying to find the extrema of?

$$\mathcal{L} = \left(\sum_{\boldsymbol{c}} \gamma(\boldsymbol{c}) \cdot \sum_{y \in \mathcal{Y}} u(y) \cdot \ln(u(y)) - u(y) \cdot \ln(p_{\boldsymbol{\theta}}(y|\boldsymbol{c}))\right) + \lambda\left(\sum_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) - 1\right) \tag{73}$$

This is as general as it gets right? Literally just taking the objective and adding a Lagrange multiplier to the equation.

Well not quite. The interpretation here is that We are summing over unique contexts, right?

So in the original smoothing paper they just go over each data item and try to optimize the KL divergence between the observed distribution (which is just discrete) and the model distribution. Well, I suppose the first set up is the cross entropy formulation which selects just one model distribution term to analyze at a time, and then they introduce the change to the assumed true distribution by converting it to a mixture of a dirac delta and a uniform noise distribution.

If instead of summing over data items You want to sum over contexts, You'll need to account for duplicates:

$$\begin{align}
&\left(\sum_{\boldsymbol{c}} (\#(y, \boldsymbol{c}) \gamma(\boldsymbol{c}) \cdot \sum_{y \in \mathcal{Y}} u(y) \cdot \ln(u(y)) - u(y) \cdot \ln(p_{\boldsymbol{\theta}}(y|\boldsymbol{c}))\right) \tag{74}\\
&+ \lambda\left(\sum_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) - 1\right) \tag{75}
\end{align}$$

Big boi lol. $\#(y, \boldsymbol{c})$ is just the count of times that a particular label y was observed with a given context $\boldsymbol{c}$.

$$\left(\sum_{y \in \mathcal{Y}} (\#(y, \boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot \big(u(y) \cdot \ln(u(y)) - u(y) \cdot \ln(p_{\boldsymbol{\theta}}(y|\boldsymbol{c})) \big) \right) \tag{76}$$

$$+ \lambda \left(\sum_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) - 1 \right) \tag{77}$$

But the indexing is all wrong.

We have a bunch of contexts. Sure.

For each unique context, We have the number of instances that that context occurred with each individual label, sure.

For each of those occurrences of a label and a context, We get a KL term. That KL term is the same for all cases, right? Yep.

So this all reduces down to

$$(\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot \left(\sum_{y \in \mathcal{Y}} u(y) \cdot \ln(u(y)) - u(y) \cdot \ln(p_{\boldsymbol{\theta}}(y|\boldsymbol{c})) \right) + \lambda \left(\sum_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) - 1 \right) \tag{78}$$

And then piecewise derivative:

$$\frac{\partial}{\partial p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})} (\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot \sum_{y \in \mathcal{Y}} u(y) \cdot \ln(u(y)) - u(y) \cdot \ln(p_{\boldsymbol{\theta}}(y|\boldsymbol{c})) \tag{79}$$

$$= (\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot -u(y) \frac{1}{p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})} \tag{80}$$

Similarly

$$\frac{\partial}{\partial p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})} \lambda \left(\sum_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) - 1 \right) \tag{81}$$

$$= \lambda \tag{82}$$

So then We in total get $|\mathcal{Y}|$ equations which read:

$$\forall y_j \in \mathcal{Y} : (\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot -u(y)\frac{1}{p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})} + \lambda = 0 \tag{83}$$

$$(\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot -u(y)\frac{1}{p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})} = -\lambda \tag{84}$$

$$(\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y)\frac{1}{p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})} = \lambda \tag{85}$$

$$\frac{1}{p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})} = \frac{\lambda}{(\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y)} \tag{86}$$

$$p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c}) = \frac{(\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y)}{\lambda} \tag{87}$$

And We can plug that into the constraint yet again

$$\sum_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) = 1 \tag{88}$$

$$\sum_{y \in \mathcal{Y}} \frac{(\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y)}{\lambda} = 1 \tag{89}$$

$$\frac{\sum_{y \in \mathcal{Y}} (\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y)}{\lambda} = 1 \tag{90}$$

$$\sum_{y \in \mathcal{Y}} (\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y) = \lambda \tag{91}$$

And then back again

$$p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c}) = \frac{(\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y)}{\lambda} \tag{92}$$

$$p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c}) = \frac{(\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y)}{\sum_{y \in \mathcal{Y}} (\#(\boldsymbol{c})) \cdot \gamma(\boldsymbol{c}) \cdot u(y)} = \frac{1}{|\mathcal{Y}|} \tag{93}$$

We did it reddit, We calculated a trivial MLE solution lol

5.10 Unigram MLE with Uniform distribution KL but right lol

$$\sum_{\boldsymbol{c}} \Big(\mathrm{KL}\left(\tilde{p}(\cdot \mid \boldsymbol{c}) \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c})\right) + \gamma(\boldsymbol{c}) \cdot \mathrm{KL}(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c}) \Big) \tag{94}$$

$$\mathrm{KL}(p(\cdot|\boldsymbol{c}), p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{c}) = \sum_{y \in \mathcal{Y}} p(y|\boldsymbol{c}) \ln \left(\frac{p(y|\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y|\boldsymbol{c})} \right) \tag{95}$$

$$\sum_{\boldsymbol{c}} \gamma(\boldsymbol{c}) \cdot \mathrm{KL}(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c}) = \gamma(\boldsymbol{c}) \cdot \sum u(x) \cdot \ln\left(\frac{u(x)}{p_{\boldsymbol{\theta}}(x|\boldsymbol{c})}\right) \tag{96}$$

$$= \gamma(\boldsymbol{c}) \cdot \sum u(x) \cdot \ln(u(x)) - u(x) \cdot \ln(p_{\boldsymbol{\theta}}(x|\boldsymbol{c})) \tag{97}$$

$$= \gamma(\boldsymbol{c}) \cdot \sum -u(x) \cdot \ln(p_{\boldsymbol{\theta}}(x|\boldsymbol{c})) \tag{98}$$

We just drop the term that depends only on the uniform distribution since it's going to be 0 when We take the derivative anyway.

Now for the indexing.

We're conditioning on a unique context.

Need to sum over labels twice?

$$\sum_{y_i \in \mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \left(\sum_{y_j \in \mathcal{Y}} p(y_j|\boldsymbol{c}) \ln\left(\frac{p(y_j|\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})}\right) + \gamma(\boldsymbol{c}) \sum_{y_k \in \mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c}))\right) \tag{99}$$

Now I feel like some of those indexes can be unified.

The outermost counts the occurrences, so that's fine I think.

y_i is the "true" label - so if $y_j \neq y_i$, then $p(y_j|context) = 0$.

$$\sum_{y_i \in \mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \left(\ln\left(\frac{p(y_i|\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right) + \gamma(\boldsymbol{c}) \sum_{y_k \in \mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c}))\right) \tag{100}$$

Now for the third summation - it really depends on the context rather than any particular true label, so it can be just split out all together:

$$\left(\sum_{y_i \in \mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \left(\ln\left(\frac{p(y_i|\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right)\right)\right) + \#(\boldsymbol{c})\gamma(\boldsymbol{c}) \sum_{y_k \in \mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c})) \tag{101}$$

Yeet. Then in total We get

$$\left(\sum_{y_i \in \mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \ln\left(\frac{1}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right)\right) \tag{102}$$

$$+ \#(\boldsymbol{c})\gamma(\boldsymbol{c})\left(\sum_{y_k \in \mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c}))\right) \tag{103}$$

$$+ \left(\lambda \sum_{y_l \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y_l|\boldsymbol{c}) - 1 \right) \tag{104}$$

Then We take the derivative w.r.t. $p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})$, where y_m is just some label

$$\#(y_m, \boldsymbol{c}) \cdot \frac{1}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} \tag{105}$$

$$- \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m) \cdot \frac{1}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} \tag{106}$$

$$+ \lambda \tag{107}$$

$$= 0 \tag{108}$$

lol
We have $|\mathcal{Y}|$ of those expressions.
So then

$$\#(y_m, \boldsymbol{c}) \cdot \frac{1}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m) \cdot \frac{1}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} + \lambda = 0 \tag{109}$$

$$(\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)) \frac{1}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} + \lambda = 0 \tag{110}$$

$$(\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)) \frac{1}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} = -\lambda \tag{111}$$

$$\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m) = -\lambda \cdot p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c}) \tag{112}$$

$$\frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)}{-\lambda} = p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c}) \tag{113}$$

$$\tag{114}$$

And then

$$\sum_{y \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y|\boldsymbol{c}) = 1 \tag{115}$$

$$\sum_{y \in \mathcal{Y}} \frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)}{-\lambda} = 1 \tag{116}$$

$$\sum_{y \in \mathcal{Y}} \#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m) = -\lambda \tag{117}$$

$$\tag{118}$$

And then

$$\frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)}{-\lambda} = p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c}) \tag{119}$$

$$\frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)}{\sum_{y \in \mathcal{Y}} \#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)} = p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c}) \tag{120}$$

So does this simplify. Sorta?

$$\frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)}{\sum_{y\in\mathcal{Y}} \#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m)} \tag{121}$$

$$\frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y)}{\sum_{y\in\mathcal{Y}} \#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y)} \tag{122}$$

$$\frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y)}{-|\mathcal{Y}|\#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y) + \sum_{y\in\mathcal{Y}} \#(y_m, \boldsymbol{c})} \tag{123}$$

$$\frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot (1/|\mathcal{Y}|)}{-|\mathcal{Y}|\#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot (1/|\mathcal{Y}|) + \sum_{y\in\mathcal{Y}} \#(y_m, \boldsymbol{c})} \tag{124}$$

$$\frac{\#(y_m, \boldsymbol{c}) - \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot (1/|\mathcal{Y}|)}{-\#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) + \sum_{y\in\mathcal{Y}} \#(y_m, \boldsymbol{c})} \tag{125}$$

$$\frac{\#(y_m, \boldsymbol{c}) - \delta \cdot (1/|\mathcal{Y}|)}{-\delta + \sum_{y\in\mathcal{Y}} \#(y_m, \boldsymbol{c})} \tag{126}$$

5.10.1 Mistakes

$$\sum_{\boldsymbol{c}} \Big(\mathrm{KL}\left(\tilde{p}(\cdot \mid \boldsymbol{c}) \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c})\right) + \gamma(\boldsymbol{c}) \cdot \mathrm{KL}(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c})\Big) \tag{127}$$

$$\mathrm{KL}(p(\cdot|\boldsymbol{c}), p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{c}) = \sum_{y\in\mathcal{Y}} p(y|\boldsymbol{c}) \ln\left(\frac{p(y|\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y|\boldsymbol{c})}\right) \tag{128}$$

$$\sum_{\boldsymbol{c}} \gamma(\boldsymbol{c}) \cdot \mathrm{KL}(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{c}) = \gamma(\boldsymbol{c}) \cdot \sum u(x) \cdot \ln\left(\frac{u(x)}{p_{\boldsymbol{\theta}}(x|\boldsymbol{c})}\right) \tag{129}$$

$$= \gamma(\boldsymbol{c}) \cdot \sum u(x) \cdot \ln(u(x)) - u(x) \cdot \ln(p_{\boldsymbol{\theta}}(x|\boldsymbol{c})) \tag{130}$$

$$= \gamma(\boldsymbol{c}) \cdot \sum_{y\in\mathcal{Y}} -u(y) \cdot \ln(p_{\boldsymbol{\theta}}(y|\boldsymbol{c})) \tag{131}$$

We just drop the term that depends only on the uniform distribution since it's going to be 0 when We take the derivative anyway.

Now for the indexing.

We're conditioning on a unique context.

Need to sum over labels twice?

$$\sum_{y_i\in\mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \left(\sum_{y_j\in\mathcal{Y}} p(y_j|\boldsymbol{c}) \ln\left(\frac{p(y_j|\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_j|\boldsymbol{c})}\right) + \gamma(\boldsymbol{c}) \sum_{y_k\in\mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c}))\right) \tag{132}$$

Now I feel like some of those indexes can be unified.
The outermost counts the occurrences, so that's fine I think.
y_i is the "true" label - so if $y_j \neq y_i$, then $p(y_j|context) = 0$.

$$\sum_{y_i \in \mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \left(\ln\left(\frac{p(y_i|\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right) + \gamma(\boldsymbol{c}) \sum_{y_k \in \mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c})) \right) \tag{133}$$

Now for the third summation - it really depends on the context rather than any particular true label, so it can be just split out all together:

$$\left(\sum_{y_i \in \mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \left(\ln\left(\frac{p(y_i|\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right)\right)\right) + \#(\boldsymbol{c})\gamma(\boldsymbol{c}) \sum_{y_k \in \mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c})) \tag{134}$$

Yeet. Then in total We get

$$\left(\sum_{y_i \in \mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \ln\left(\frac{1}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right)\right) \tag{135}$$

$$+ \#(\boldsymbol{c})\gamma(\boldsymbol{c}) \left(\sum_{y_k \in \mathcal{Y}} -u(y_k) \cdot \ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c}))\right) \tag{136}$$

$$+ \left(\lambda \sum_{y_l \in \mathcal{Y}} p_{\boldsymbol{\theta}}(y_l|\boldsymbol{c}) - 1 \right) \tag{137}$$

Then We take the derivative w.r.t. $p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})$, where y_m is just some label

$$\#(y_m, \boldsymbol{c}) \cdot \frac{1}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} \tag{138}$$

$$- \#(\boldsymbol{c}) \cdot \gamma(\boldsymbol{c}) \cdot u(y_m) \cdot \frac{1}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} \tag{139}$$

$$+ \lambda \tag{140}$$

$$= 0 \tag{141}$$

And this is where things went wrong. Instead take the derivative piece by piece:

$$\frac{\partial}{\partial p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} \left(\sum_{y_i \in \mathcal{Y}} (\#(y_i, \boldsymbol{c})) \cdot \ln\left(\frac{1}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right)\right) \tag{142}$$

$$\tag{143}$$

Let's just do

$$\frac{d}{dx}\ln\left(\frac{1}{x}\right) = \frac{d}{du}\ln(u)\cdot\frac{d}{dx}\frac{1}{x}, \qquad u = \frac{1}{x} \tag{144}$$

$$= \frac{1}{u}\cdot\frac{-1}{x^2} \tag{145}$$

$$= \frac{1}{x^{-1}}\cdot\frac{-1}{x^2} \tag{146}$$

$$= -x^{-1} \tag{147}$$

So then back again

$$\frac{\partial}{\partial p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})}\sum_{y_i\in\mathcal{Y}}(\#(y_i,\boldsymbol{c}))\cdot\ln\left(\frac{1}{p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c})}\right) = \frac{-\#(y_m,\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} \tag{148}$$

Next We have

$$\frac{\partial}{\partial p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})}\#(\boldsymbol{c})\gamma(\boldsymbol{c})\Bigg(\sum_{y_k\in\mathcal{Y}} -u(y_k)\cdot\ln(p_{\boldsymbol{\theta}}(y_k|\boldsymbol{c}))\Bigg) = -\frac{\#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} \tag{149}$$

$$\tag{150}$$

and the last one's just λ lol
All together then:

$$\frac{-\#(y_m,\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} - \frac{\#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} + \lambda = 0 \tag{151}$$

$$\frac{-\#(y_m,\boldsymbol{c})}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} - \frac{\#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c})} = -\lambda \tag{152}$$

$$-\#(y_m,\boldsymbol{c}) - \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y) = -\lambda p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c}) \tag{153}$$

$$\frac{\#(y_m,\boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\lambda} = p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c}) \tag{154}$$

Then back again

$$\sum_{y_i\in\mathcal{Y}} p_{\boldsymbol{\theta}}(y_i|\boldsymbol{c}) = 1 \tag{155}$$

$$\sum_{y_i\in\mathcal{Y}}\frac{\#(y_i,\boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\lambda} = 1 \tag{156}$$

$$\frac{\sum_{y_i\in\mathcal{Y}}\#(y_i,\boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\lambda} = 1 \tag{157}$$

$$\sum_{y_i\in\mathcal{Y}}\#(y_i,\boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y) = \lambda \tag{158}$$

Then back again

$$\frac{\#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\lambda} = p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c}) \tag{159}$$

$$\frac{\#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\sum_{y_i \in \mathcal{Y}} \#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)} = p_{\boldsymbol{\theta}}(y_m|\boldsymbol{c}) \tag{160}$$

And then You try to simplify I suppose

$$\frac{\#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\sum_{y_i \in \mathcal{Y}} \#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)} \tag{161}$$

$$\frac{\#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\#(\boldsymbol{c}) + \sum_{y_i \in \mathcal{Y}} \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)} \tag{162}$$

$$\frac{\#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\#(\boldsymbol{c}) + |\mathcal{Y}|\#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)} \tag{163}$$

$$\frac{\#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\#(\boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})} \tag{164}$$

Then letting $\gamma(\boldsymbol{c}) = \alpha/\#(\boldsymbol{c})$.

$$\frac{\#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\#(\boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})} = \frac{\#(y_i, \boldsymbol{c}) + \alpha u(y)}{\#(\boldsymbol{c}) + \alpha} \tag{165}$$

$$= \frac{\#(y_i, \boldsymbol{c}) + \frac{\alpha}{|\mathcal{Y}|}}{\#(\boldsymbol{c}) + \alpha} \tag{166}$$

Das ist correct, I think. To recover the standard formulation for Laplace smoothing let $\gamma(\boldsymbol{c}) = \frac{|\mathcal{Y}|\alpha}{\#(\boldsymbol{c})}$.

$$\frac{\#(y_i, \boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})u(y)}{\#(\boldsymbol{c}) + \#(\boldsymbol{c})\gamma(\boldsymbol{c})} = \frac{\#(y_i, \boldsymbol{c}) + |\mathcal{Y}|\alpha u(y)}{\#(\boldsymbol{c}) + |\mathcal{Y}|\alpha} \tag{167}$$

$$= \frac{\#(y_i, \boldsymbol{c}) + \alpha}{\#(\boldsymbol{c}) + |\mathcal{Y}|\alpha} \tag{168}$$