Estimation, Evaluations and Hold Outs

Today we will look at additional techniques for estimating probability distributions. Remember that the general approach is to find distributions that maximize the likelihood (or the expectation) of a corpus that we hope characterizes the new data. Often, this corpus is simply the training data itself, although today we look at technique that attempts to better model new data by "holding back" some data when doing the original training.

1. Evaluation Using a Development Corpus

Given we have different techniques for estimating probability functions, how can we know which one is best. We know if we calculate the probability (or log probability) of the training corpus, the MLE will appear to be the best. Thus, we need to have a second corpus, often called the **development corpus**, which we can use to evaluate our results. Once we use the development corpus to select the best approximation, then we are ready for the actual evaluation on new data (the **test corpus**). For instance, let us assume we have the different models for the ABC corpus from last time, and we collect another smaller development corpus, which is

C B C C A A B

Since we are evaluating a trigram model, lets recast this corpus as a set showing the element and its context (starting at 3 element so we don't have to use any start symbols in order to keep the example short):

<C|CB> <C|BC> <A|CC> <A|CA><B|AA>

We can now calculate the likelihood of this development corpus for the different values of lambda:

λ	Likelihood of development corpus	Log Likelihood
1	.014	-1.84
.5	.018	-1.74
.31	.019	-1.71
.01	.002	-2.62

Table 1: Evaluating different λ's on a development corpus

So we see that, assuming the development corpus is representative of new data (which is unlikely given its small size), then λ=.31 is the best of these values.

2. Held Out Estimator Technique

This idea of using an additional corpus can be extended to getting better estimates of seen events as well. For this technique, we will call this additional corpus the **held out data** for in essence it is a subset of the training data that is withheld from the initial estimation procedure. We could count how many times each observation occurred in both a training set and the held-out data, and use these number to explore how often observations that occur r times in the training data occur in the development data. This would not only give us an estimate of how often observations not in the training data occur in subsequent data but also how often observations that occur r times occur in subsequent data.

To set this up, we first divide our training data into two corpora: T, still called the training data, and HO, called the held out data. For each vocabulary item o, we can compute two counts: $Count_T(o)$ is the number of times o occurs in the training data, and $Count_{HO}(o)$ is how many times it occurs in the development data. Now we gather together all the types that occur the same number of times in the training corpus: let Class(r) be the set of all observation types that occurred exactly r times in the training corpus. If we uses the standard MLE technique, we would estimate the probability of each item in Class(r) as having a probability r/N. Rather than use this, however, we look at how many times elements in each class occur in HO. Specifically, we count how many times any of these items occur in the development corpus:

$$ClassCnt(r) = \sum_{o \text{ in } Class(r)} Count_{HO}(o)$$

Now we can compute the average number of times an observation that occurred r times in the training corpus occurs in the development corpus:

$$AverageCnt(r) = ClassCnt(r)/ |Class(r)|$$

We now use this "adjusted count" for producing the probability estimate

$P_{HO}(o) = AverageCnt(r) / N_{HO}$, where $Count_T(o) = r$, and N_{HO} is the size of the held out data.

The most clear case where this helps is seen by looking at the elements that do not occur in the training corpus. Let's use the trigram example above from last lecture. There were 14 trigrams that did not occur in the training corpus. Let's assume only 6 of these occur in HO with size 40 (it doesn't matter if the same word occurred three times or three words occurred once). So we have

$$ClassCnt(0) = 6$$

Thus the average number of times we can expect one of these unseen bigrams to appear in the held out data is

$$AverageCnt(0) = 6/40 = .15$$

Thus, the held-out probability estimate for any one of these, say, AAA, is

$$P_{HO}(AAA) = .15/40 = .00375$$

An Example

Lets estimate the joint probability distribution for triples. The training corpus will be the same ABD corpus as last time:

Training Data: T = {A B C A B B C C A C B C A A C B C C B C}

and the following is the held-out data

Held Out Data: HO = {C B C C B A A C B C A B C}.

The sizes of the two sets differ, and $N_T = 18$ and $N_{HO} = 11$.

We can count the triples in each corpus and compute the following table. We would get the analysis shown in table 2.

r	Members of Class(r)	Size of Class(r)	Class-Cnt(r)	AverageCnt(r)	P_{HO}(w) for w in class(r)
3	{CBC}	1	2	2	.18
2	{BCA, BCC, ACB}	3	3	1	.09
1	{AAC, ABB, ABC, BBC, CAA, CAB, CAC, CCA, CCB}	9	3	.333	.03
0	{AAA, AAB, ABA, ACA, ACC, BAA, BAB, BAC, BBA, BBB, BCB, CBA, CBB, CCC}	14	3	.214	.019

Table 2: Re- estimating using held-out data

Using the held out data to revise the original counts, we have an empirical measure of the degree of uncertainty in the data and can adjust the probabilities accordingly. By treating all n-grams that appear R times as a set, we avoid random fluctuations dramatically changing the probability estimates based on the held out data. The final column gives the probability estimate of each triple:

e.g., P(AAA) = .019

Note that the probability estimated to elements that appear once in the corpus (.03) is not that different from the probability assigned to elements that did not appear (.019). This is appropriate since with such rare events, whether it happens once or not at all is not that significant.

To compute the conditional probabilities P(z | y z), we need to produce a value for P(y z). We can't estimate this independently of the triple joint distribution otherwise we risk not producing a probability distribution. But we can compute the joint sitribution for pairs from the triple mode by observing that

P(y z) = $\sum_i$ P(y z v_i), where v_i ranges over the unigram vocabulary.

For example,

$$P(B\ C) = P(B\ C\ A) + P(B\ C\ B) + P(B\ C\ C)$$
$$= .09 + .019 + .09 \quad \text{(taking values from Table 2)}$$
$$= .199$$

Thus $P(A \mid B\ C) = .09 / .199 \approx .45$ and $P(B \mid C\ B) = .019/.199 \approx .096$.

Let's compare this distribution on the development corpus that we used to evaluate the add-λ estimates. Table 3 gives the conditional probabilities used in the development corpus.

	$P_{HO}(x\ y)$	$P_{HO}(z \mid x\ y)$
$P(C \mid C\ B)$	.18+.019+.019 = .218	.18/.218 ≈ .83
$P(C \mid B\ C)$	.09+.09+.019 =.199	.09/.199 ≈ .45
$P(A \mid C\ C)$	.03+.03+.019 = .079	.03/.079 ≈ .38
$P(A \mid C\ A)$	.03+.03+.03 = .09	.03/.09 ≈ .333
$P(B \mid A\ A)$	.03+.019+.019 = .068	.019/.068 ≈ .28

Thus the likelihood of the development corpus in this case is

.83*.45*.38*.333*.28 ≈ .013

This is a bit worse than some of the add-λ estimates, we can't attach much significant to this because of the ridiculously small corpora being used. The significance of this depends on whether the development corpus is representative of new data, which given its small size, is unlikely.

3. Different Approaches to Handling Held Out Data

As mentioned above, one disadvantage of the held out method appears to be that we need to reduce our training corpora in order to produce the held out data. So the smaller initial training set could hurt us.

One way around this is two do the estimate with several different sets for held-out data . Say we divide the corpus into two parts: A and B. First we initially train on A and use B as the held out data, and then we create a new model by training on B and using A as the held-out data. We then have two estimates, P_{m1} and P_{m2} and can combine them to make a new estimate P_m using

$$P_m(o) = .5 * P_{m1}(o) + .5\ P_{m2}(o)$$

Of course, we need not divide the corpus exactly into half. Experience has shown that, assuming the corpus is large enough, it is better to use more data for the initial estimate, saving only about 10% for the held-out data. This suggests a generalization of the above strategy. We divide the initial corpus into ten parts, and build ten estimates each one using a different part as the held out data. The final probability distribution would then be constructed using the average over the ten estimations obtained.

Cross-Validation

Note we can use the same technique for test data as well for evaluation. Say we have one corpus and need to select part of it as the test data. Once again, typically people choose test data to be 10% of the size of the training data. Rather than doing this once, we could do a more extensive evaluation of a technique by do ten experiments, each one using a different subset of the corpora as the test set and the rest as the training. This is called the cross-validation technique and is an effective way to perform more extensive testing of a technique without requiring more data.

Of course, if we were evaluating a technique that requires held-out data, we'd need to combine both approaches. In this case, we'd select one subset as test data, one as held-out data and train on the rest. There are now a hundred combinations that perform different experiments using the same corpora. The final evaluation result would be the average of the individual experiments performed.

5. Interpolation Methods

Another method for dealing with unseen events is to use a combination of probabilistic models. This especially useful when we are trying to estimate conditional probabilities and may construct a series of estimates of distributions using different contexts. For instance, to estimate the probability of a word given the context of the previous two words, we might use a linear combination of trigram, bigram and unigram models. In particular,

$$P_{li}(w_3 \mid w_1\ w_2) = \lambda_1 P_1(w_3) + \lambda_2 P_2(w_3 \mid w_2) + \lambda_3 P_3(w_3 \mid w_1\ w_2)$$

Where P_1, P_2 and P_3 are the unigram, bigram and trigram estimates respectively. To guarantee that P_{li} is a probability distribution, we must require each λ_i be between 0 and 1, and $\sum_i \lambda_i = 1$. (You proved a version of this in assignment 1)

For example, using the same ABC corpus (repeated here for convenience) and the standard MLE technique.

A B C A B B C C A C B C A A C B C C B C

we can use MLE to estimate the bigram and unigram probabilities. The trigram probabilities were estimated in the last lecture.

Unigram	Count	MLE
A	5	.25
B	6	.3
C	9	.45

Table 1: The Unigram estimates

Bigram: x_{i-1} x_i	Pair Count	$P_{MLE}(x_i \mid x_{i-1})$
A A	1	.2
A B	2	.4
A C	2	.4
B A	0	0
B B	1	.167
B C	5	.833
C A	3	.375
C B	3	.375
C C	2	.25

Table 2: The "bigram" conditional probability

We could set the weighting factors by hand. Lets say $\lambda_1 = .05$, $\lambda_2 = .1$ and $\lambda_3 = .85$. With these weights, the linear interpolated probability function would assign the following probability to the sequence A A A.

$P_{li}(A \mid A\ A)$
$= .05 * P1(A) + .1 * P2(A \mid A) + .85 * P3(A \mid A\ A)$
$= .05 * .25 + .1 * .2 + .85 * 0$
$= .0125 + .02 = .0325$

By assigning some probability to unseen elements, this method obviously takes some probability from the ones that were seen. For instance, the most common trigram CBC is the only trigram beginning with CB. Thus $P_{MLE}(C \mid CB) = 1.0$. With linear interpolation we get

$P_{li}(C \mid CB)$
$= .85 * P3(C \mid B\ C) + .15 * P2(C \mid B) + .05 * P1(C)$
$= .85 * 1 + .1 * .833 + .05 * .45$
$= .85 + .083 + .0225$
$= .9558$

This estimate is slightly lower but still in the right ballpark because the trigram, bigram and unigram estimates all found the combination likely.

Table 3 looks at how this model works on our development corpus.

Element	$P_{MLE}(z \mid x\ y)$	$P_{MLE}(z \mid y)$	$P_{MLE}(z)$	Linear Combination
$P_{LI}(C \mid CB)$	1	.833	.45	.9558
$P_{LI}(C \mid BC)$	.5	.25	.45	.4725
$P_{LI}(A \mid CC)$	.5	.375	.25	.475
$P_{LI}(A \mid CA)$	1	.2	.25	.8825
$P_{LI}(B \mid AA)$	0	.4	.3	.055

Table 3: Calculating the likelihood of the developm ent corpus

Thus the likelihood of the development corpus with this estimate is .01. We could experiment with different weights and see what combination works best in practice. We could also use a learning procedure to determine good values for these parameters using an iterative procedure over a held out dataset.

6. Back-Off Methods

The final approach is similar to linear interpolation but combines the estimates differently. . For example, if we are interested in estimating $P(W_i \mid W_{i-2}\ W_{i-1})$, we use the MLE estimate for the trigram if we feel we have a good enough estimate for the trigram. By "good enough" we mean that it occurred more than some number k in the training data. If it is not good enough, then we use the bigram estimate discounted by some factor β. In other words, our initial estimates (before normalization) would be

$$P_{BO}(W_i \mid W_{i-2}\ W_{i-1}) = \text{if } C(W_{i-2}\ W_{i-1}\ W_I) > k \text{ then } P_{MLE}(W_i \mid W_{i-2}\ W_{i-1})$$
$$\text{else } \beta_2 * P'_{BO}(W_i \mid W_{i-1})$$

We would compute the backoff estimate of the bigram in the same way, backing off to the unigram estimate if necessary. We would then compute the backoff probability distribution by normalizing in the usual way.

Note to use this method for a general n-gram model, we need to set *k*, the minimum number of observations to make us believe we have a good estimate, and a series of normalizing factors $\beta_{n-1} \ldots \beta_1$ for each possible backoff from an n-gram to an n-1-gram. More sophisticated models can be developed that would discount the MLE estimates above and encode the remaining probability mass in the $\beta_\square$'s, thus eliminating the need for renormalization. Back-off models were suggested by Katz, and his particular method including techniques for discounting is often called the **Katz Back-off Model**. There are more details in the text.