Estimating Probability Distributions

Readings: Manning and Schutze, Section 6.2
Jurafsky & Martin, Section 6.3

One of the central problems we face in using probability models for NLP is obtaining the actual distributions. The true distributions are invariably not known, and we must estimate them for training data. This chapter explores the basic concepts and techniques for estimating probability distributions from data.

Let us start with a simple problem. Say we have a button and every time we push it a color is shown. So we press it six times and we see

R R B G R B

We want to construct a probabilistic model of this process, in other words, define a probabilistic model that characterizes what the result of the next element in the sequence is. The first thing we need to do is decide on the form of the model. We believe that this data is generated by a stochastic process with a sequence of random variable X_i that generates a sequence $o_1, \ldots, o_2$. Lets say also that we know that the possible values, i.e., the vocabulary **V** (i.e., sample space) for X is {R, G, B, Y}, for red, green blue and yellow. We are interested in defining a good probability model for the i'th (i.e., next) element in the sequence given the previous values. Ideally this would be a conditional probability function of form

$P(X_i \mid X_{1,I-1})$ (i.e., $X_i \mid X_1, X_2, \ldots, X_{i-1}$))

But as the context grows, we would need a new probability function for each value of I, which would be a very complex model to estimate. Rather, we assume that this is a stationary stochastic process, so that each position i has the same probability distribution function. To enable this, we must fix the context that we can look back to by making independence assumptions. For instance, the strongest assumption we can make is that each X_i is independent (called the unigram model), and thus each value would be independent of the other values and we would need to estimate a simple distribution P(X) (= $P(X_x$ for all i). A stronger model, often called a bigram model, would say that the output of step i is dependent on step i-1, and thus distribution we use at each stage is the conditional probability $P(X_i|X_{i-1})$. In general, an n-gram model would use a the conditional probability $P(X_i \mid X_{1,n-1})$.

To explore estimation techniques, let's assume the simplest case–there is a single distribution X that we need to estimate. Given this, we can view the sequence as 6 independent samples and can observe that R occurred 3 times, B twice, and G once. We will typically summarize such situations using the function *Cnt* as follows:

Cnt(R) = 3
Cnt(B)=2
Cnt(G)=1

What makes a good probability distribution for a given set of data?

What criteria should we use to decide on a probability distribution for X? In the abstract, we want a distribution that most accurately predicts what the next value will be. But since we can't see the next value, we can develop an evaluation measure based on this idea. The vasic measure of goodness that is used considers what the distribution says about the likelihood of the observed data. Presumably, a distribution that assigns a higher probability to the observed data is a better model that one that assigns a lower probability. To be concrete, consider the uniform distribution, which would assign an equal probability of .25 (i.e., 1/4) to each of the four values. Intuitively, this doesn't seem a good model for the observed data, since the counts do not look very uniform. With this distribution, the probability of observing the 6 values would be $.25^6 = .00024$ (which we will write as 2.4e-4). Table 1 compares this with two other possible distributions. The second is the uniform distribution again except that it only assigns probability to values that were observed in the corpus.

Distribution	P(R)	P(B)	P(G)	P(Y)	Prob of Sequence: R R B G R B
U1: Uniform	.25	.25	.25	.25	2.4 e –4
U2: Uniform over observed values	.33	.33	.33	0	1.6 e –3
MLE (Ratio based on counts)	.5 (=3/6)	.33 (=2/6)	.167 (=1/6)	0	2.3 d -3

The third one, called the **maximum likelihood estimate** (**MLE**), simply uses the counts from the corpus to estimate the distribution:

$$P_{MLE}(X=v_v) = Cnt(v_v) / N, \text{ where } v_v \in V$$

Here v_v ranges over the vocabulary, and N is the size of the corpus. The MLE has some nice formal properties. It can be proven that the MLE assigns the highest probability to the overall corpus of any possible probability distribution (see box 1). Also, an important theorem of probability called the **law of large numbers** shows that the more data you use for the MLE estimate, the better the estimate actually gets (more formally, the true distribution is the limit of the estimates as the dataset size goes to infinity). But the MLE has serious problems in practice, as we shall see soon. Even so, the techniques used in MLE estimation are the basis for developing better estimation schemes, so let's consider MLE estimation in a bit more detail.

MLE Estimates for more Complex Distributions

For a simple unigram distribution, the MLE estimate simple sets the probability of each value to the proportion of times that value occurs in the data. What if we want to estimate a joint distribution, say $P(X_{i-1}, X_i)$. To estimate this distribution, we view the dataset as a series of pairs. Typically, when we do this we add special values at the beginning of the sequence for when i would be 0 or less. This value will be written as □. Viewed as a series of pairs, the data above would be

<□, R> <R, R> <R,B> <B,G> <G,R> <R,B>

Now we can estimate the joint distribution simply by counting as before:

Cnt(<□, R>) = 1
Cnt(<R, R>) = 1

Box 1: Proving that the MLE assigns the highest probability to the corpus
Consider the situation in which we flip a coin three times and get H H T. We can view this as a sequence of observations $O_{1,3}$ and we would like to find the probability distribution p that makes this observed sequence most likely. Since we assume that each coin flip is independent of the others, we know that $P(O_{1,n}) = P(o_1)*\ldots* P(o_n)$. If we assume P(H) = m, then P(T) = (1 –m), so the probability of the sequence H H T would be $m*m*(1-m) = m^2 – m^3$. Remembering our basic calculus, we can find the maximum of this function by differentiating it to obtain $2m – 3\ m^2$ and setting this to 0. The one non-zero solution for this is m = 2/3. This says that we maximize the probability of the sequence by setting P(H) = 2/3 (which is of course exactly the MLE). One can prove in general that the MLE always has this property of maximizing the probability (or

Cnt(<R,B>) =2
Cnt(<B,G>) =1
Cnt(<G,R>) =1

Thus, P(□, R) = 1/6, P(R, R) = 1/6, P(R, B) = 1/3, and so on. Note this is very close to the uniform distribution over observed values– which reflects the fact that we really don't have enough training data to get good estimates, This is not surprising because the vocabulary in this case, the set of all pairs, contains 20 items, and we only have six observations.

To see how to estimate the conditional probability distribution, $P(X_i \mid X_{i-1})$, we could expand it with its definition

$$P(X_i \mid X_{i-1}) = P(X_{i-1}, X_i) / P(X_{i-1})$$
$$= (Cnt(<X_{i-1},X_i>)/N) / (Cnt(X_{i-1})/N)$$
$$= Cnt(<X_{i-1},X_i>)/ (Cnt(X_{i-1})$$

Thus P(R | □) = 1/1 = 1, P(R | R) = 1/3, P(B | R) = 2/3. and so on.

Why the MLE doesn't work well

While MLE is guaranteed to maximizes the probability of an observed data, we are actually interested in finding estimators that perform well on *new* data. A serious problem arises from this perspective because the MLE assigns a zero probability to elements that have not been observed in the corpus. This means it will assign a zero probability to any sequence containing a previously unseen element. For instance, given the above example, let's assume that the appearance of the value Y is relatively rare, say 1 in 50. Even so, the probability that a 6 element sequence contains the element Y will be 12%. That means that 12% of possible new sequences would be assigned a probability of 0 (clearly ridiculous since the sequence just occurred!). For instance, the sequence R R R Y R R, consisting of 5 of the most common value and one yellow would get an estimate of 0

from the MLE and distribution U2, while the uniform distribution would assign it 2.4 e –4, the same value it applies to any sequence of length 6.

You might think we could cure this by making sure w have enough data, but Zipf's law indicates that we'll always have rare words not matter how large the corpora. Furthermore, as we try to use more accurate probability models that make less independence assumptions, the amount of training data required would be truly staggering. For example, to estimate a probability model for on word bigrams given a small English vocabulary of 20,000 words, the number of possible bigrams would be $20{,}000^2 = 400{,}000{,}000$, much larger than most corpora.

A second problem is that the MLE doesn't distinguish between different levels of certainty. For instance, say we draw balls from an urn, observe and replace. We know there are ten different colors of balls. If we observed 10 red balls out of 40 draws, we have some confidence that the probability of drawing a red ball is 1/4. If we draw only four balls, and one is red, we get the same probability estimate, but we should not be so confident. The MLE does not reflect any degree of uncertainty we might have based on the amount of evidence we have seen.

One way to address these problems is to incorporate some **prior knowledge** into our probability estimate (the practice often called Bayesian Statistics). For instance, if we have never drawn a ball from the urn yet, it would be best to assume that each ball is equally likely. This is the best distribution we could guess in the sense it would maximize our expected probability of an arbitrary sequence drawn from the urn. If you guessed some other distribution of the data, then you might get lucky on some sample sequences, but would lose overall on average. In this rest of this chapter we review some of the most common techniques for adding priors that produce better estimates.

Smoothing Using the Add-λ technique

One way to capture this property when we do have observations is to assign a small amount of probability to each possible observation at the start. We do this by adding some small number λ to the count of each outcome to get the estimation formula (sometimes called the **Lidstone's estimation**):

$$P_\lambda(x_i) = (Cnt(x_i) + \lambda) / (N + V* \lambda)$$

Where N is the size of the corpus (i.e., the number of tokens) and V is the size of the vocabulary (i.e., the number of types). We need to add the $V * \lambda$ to the denominator in order for the estimates to add to 1, making it a probability distribution.

When λ =1, this technique essentially starts by saying every possible event has occurred at least once, and it is called **Laplace estimation**. Let's see how this technique solves our two problems, assuming λ =1: Concerning the first problem, if we never see an token of a type T in a corpus of size N and vocabulary size |V|, the probability estimate of a token of T occurring will be 1/(N+|V|). No type will have a zero probability. Concerning the second, consider the example above involving four draws and forty draws. With ten red out of forty draws, the estimate is (10+1) /(40+10) = 11/50 = .22. So we have an estimate fairly close to the .25 the MLE estimate would give. With the four draws, the estimate is

(1+1)/(4+10) = 2/14 = 1.4. So although we saw a red ball 1/4 of the time in the four trials, our prior knowledge that there are ten balls makes our estimate stay close to the uniform distribution value of .1. This is good because one observation out of four is not very strong evidence.

One problem with this technique occurs when there are a large number of possible events that have not been seen, For instance, consider an example cited in Manning and Schutze. In a test of data trained on 22 million words from the AP wire, Church and Gale found a vocabulary of about 400,000 words, (yielding $1.6 * 10^{11}$ possible bigrams, far larger than the maximum number of different bigrams in the corpus (approx. 22,000,000)). The Laplace estimate would estimate that each unseen bigram would have a probability of

$$1/(N + |V|)$$

where N=22,000,000 and $V=1.6 * 10^{11}$, a very small number. But we know that there are at least B – N bigrams that are not seen. So the amount of probability mass assigned to unseen bigrams would be at least

$$(|V|-N) * 1/(N + |V|)$$
$$= (|V| – N)/(N + |V|)$$
$$= .9997$$

Thus, this model predicts that 99% of the new data will consist of bigrams not seen before!!!! One way to compensate for this is to make λ smaller. Another common value for λ is .5, and this is called **Expected Likelihood Estimation** (**ELE**). For an arbitrary value of λ, the probability of each unseen bigram would be

$$.5 /(N + |V| * .5)$$

In the test on 44 million words of English above, this estimate would predict that 23% of the bigrams would not have been seen before, a much better match with the data.

Empically Setting λ

<<do for RBYG example>>>Or we could pick another value for λ. In fact, if we had a second corpus, we could empirically estimate the probability of seeing a new bigram, and then use that number to work backwards to derive a value for λ that matches it.

Now let's estimate what an appropriate value for λ might be. Say we have another corpus consisting of 12 outcomes of which two are not found in the original corpus. Thus, we estimate that the probability of encountering a new bigram is 1/6 = .166. We know the probability estimate for a single unseen outcome from the original corpus is

$$\lambda / (18 + 27\lambda)$$

and there are 14 such outcomes, so we set

$$14 \lambda / (18 + 27\lambda) = 1/6$$

and solve for λ, obtaining the value .31. By setting λ to this value, we would produce a probability distribution that correctly predicts the expected number of unseen elements (as calculated from the second training corpus).

Using add-□ Techniques For More Conditional Probability Functions

We saw above how to compute the MLE estimate the probability conditional probability functions. But what is the equivalent technique for conditional probabilities. Consider the bigram case, say estimate

$$P(B \mid R) = P(B,R)/P(R)$$

One might think we can simply add □ to each of bigram and unigram counts. But this causes a problem. Consider the estimate for a bigram YG, where neither the unigram Y nor the bigram YG has been seen before. If we simply add □ to each, then the estimate for $P(G \mid Y) = (0 + □)/(0+□) = 1$, clearly a ridiculous value, especially since we'd also get the same value of 1 for $P(R \mid Y)$!!! The problem is that there are potentially |V| bigrams for each possible unigram, where V is the unigram vocabulary, in our case 4. So to get the right estimate, we need to add |V|*□ to the unigram counts. Thus, the add-l estimate for a conditional probability is given by

$$P_□(X \mid Y) = (Cnt(<X,Y>)+□)/(Cnt(Y)+|V| □ □)$$

We generalize similarly for higher-order n-grams. For example, consider the trigram estimates for a new language ABC with V={A, B, C}, and given the corpus

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

There are $3^3 = 27$ different possible trigram types, only 13 of which are observed outcomes (ABC, BCA, CAB, …,CBC) and thus 14 have not been seen. The raw counts are show in table one, organized by the "context" (i.e., the previous two letters).

Trigram	Count	Trigram	Count	Trigram	Count	Context	Count
AAA	0	AAB	0	AAC	1	AA	1
ABA	0	ABB	1	ABC	1	AB	2
ACA	0	ACB	2	ACC	0	AC	2
BAA	0	BAB	0	BAC	0	BA	0
BBA	0	BBB	0	BBC	1	BB	1
BCA	2	BCB	0	BCC	2	BC	4
CAA	1	CAB	1	CAC	1	CA	1
CBA	0	CBB	0	CBC	3	CB	3
CCA	1	CCB	1	CCC	0	CC	2

Table 1: The raw trigram counts

We can see two problems arise from the MLE estimation. First, the probability of some contexts (namely BA) is 0, so the conditional probability is undefined. Table 2 shows the counts that result from the add-□ technique. One way to look at the counts for the bigram pairs this is to observe that since every triple of form XYZ contains the bigram XY, and every bigram XY is in one triple, then $Count(X\ Y) = \sum_i Count(X\ Y\ v_i)$, where v_i ranges over the unigram vocabulary.

Trigram	Count	Trigram	Count	Trigram	Count	Context	Count
AAA	δ	AAB	δ	AAC	1+ δ	AA	1+ 3δ
ABA	δ	ABB	1+ δ	ABC	1+ δ	AB	2+ 3δ
ACA	δ	ACB	2+ δ	ACC	δ	AC	2+ 3δ
BAA	δ	BAB	δ	BAC	δ	BA	3δ
BBA	δ	BBB	δ	BBC	1+ δ	BB	1+ 3δ
BCA	2+ δ	BCB	δ	BCC	2+ δ	BC	4+ 3δ
CAA	1+ δ	CAB	1+ δ	CAC	1+ δ	CA	1+ 3δ
CBA	δ	CBB	δ	CBC	3+ δ	CB	3+ 3δ
CCA	1+ δ	CCB	1+ δ	CCC	δ	CC	2+ 3δ

Table 2: The smoothed counts using the add-δ Technique

Table three shows the conditional probabilities calculate for a sampling of trigrams using different values of δ.

Prob	δ = 1	δ = .5	δ = .31	δ = .01
P(A | AA)	.25	.2	.16	.01
P(C | AA)	.5	.6	.68	.98
P(C | BA)	.333	.333	.333	.333
P(A | BC)	.428	.45	.47	.499
P(B | BC)	.143	.2	.06	.002
P(C | BC)	.428	.45	.47	.499

Table 3: The Conditional Prob. Estimates for some different values of δ

Now we have this information, what does it tell us? Actually, just looking at the distributions doesn't answer this, for we don't know which is a better distribution for accounting for new data. We see as δ gets larger, the distribution becomes more uniform, as as it gets smaller, it approaches the MLE. Next time we will explore methods of empirically evaluating different distributions.