Statistical Inference:

n-gram Models over Sparse Data

(M&S Ch 6)

Overview

- Statistical Inference consists of taking some data (generated in accordance with some unknown probability distribution) and then making some inferences about this distribution.
- There are three issues to consider:
  - Dividing the training data into equivalence classes
  - Finding a good statistical estimator for each equivalence class
  - Combining multiple estimators

Forming Equivalence Classes I

- ***Classification Problem***: try to predict the target feature based on various classificatory features. ==> ***Reliability versus discrimination***
- ***Markov Assumption***: Only the prior local context affects the next entry: (n-1)th Markov Model or n-gram
- Size of the n-gram models versus number of parameters: we would like n to be large, but the number of parameters increases exponentially with n.
- There exist other ways to form equivalence classes of the history, but they require more complicated methods ==> will use n-grams here.

Statistical Estimators I: Overview

- ***Goal***: To derive a good probability estimate for the target feature based on observed data
- ***Running Example***: From n-gram data

$P(w_1,..,w_n)$ predict $P(w_{n+1}|w_1,..,w_n)$

- ***Solutions we will look at***:
  - Maximum Likelihood Estimation
  - Laplace's, Lidstone's and Jeffreys-Perks' Laws
  - Held Out Estimation
  - Cross-Validation
  - Good-Turing Estimation

Statistical Estimators II: *Maximum Likelihood Estimation*

- $P_{MLE}(w_1,..,w_n)=C(w_1,..,w_n)/N$, where $C(w_1,..,w_n)$ is the frequency of n-gram $w_1,..,w_n$
- $P_{MLE}(w_n|w_1,..,w_{n-1})= C(w_1,..,w_n)/C(w_1,..,w_{n-1})$

- This estimate is called ***Maximum Likelihood Estimate*** (MLE) because it is the choice of parameters that gives the highest probability to the training corpus.
- MLE is usually unsuitable for NLP because of the sparseness of the data ==> Use a ***Discounting*** or ***Smoothing*** technique.

Example

20 the green

1	the green, an	1	the green. His	1	the green room
1	the green areas	3	the green jungle	1	the green stud
1	the green bay	1	the green light	1	the green stuff
1	the green beneath	2	the green of	1	the green suite
1	the green demon	1	the green park	1	the green water
1	the green eyes	1	the green, past		

$$P(\mathit{jungle} \mid \mathit{the\ green}) = 3/20 = 0.15$$
$$P(\mathit{light} \mid \mathit{the\ green}) = 1/20 = 0.05$$
$$P(\mathit{sea} \mid \mathit{the\ green}) = 0/20 = 0$$

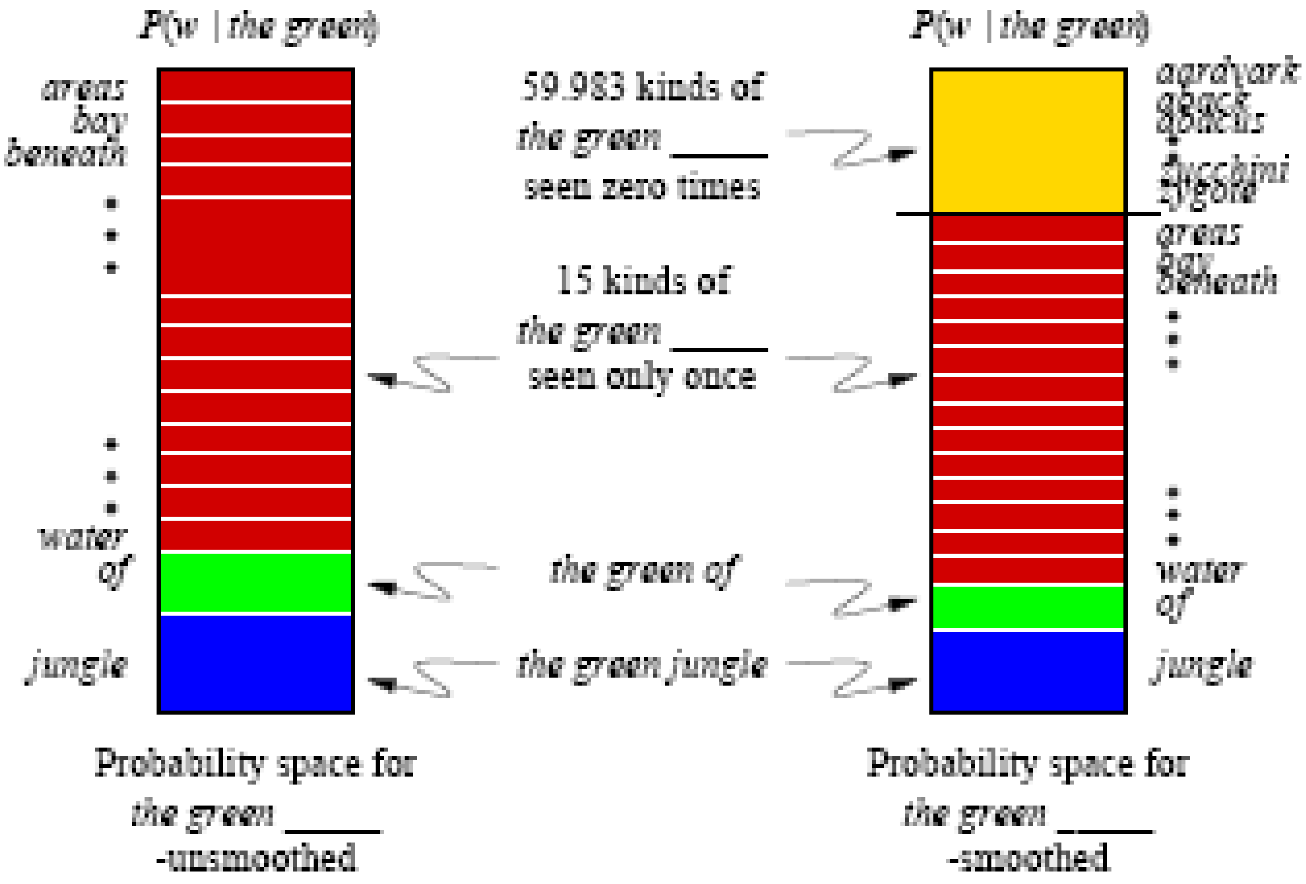
Table 4: A fragment of an ARPA-format bigram language model produced by the SLM Toolkit.

```
\data\
ngram 1=35770
ngram 2=444895

\1-grams:
-6.1177 <UNK>    0.0000
-99.0000 </s>   -4.7968
-99.0000 <s>    -0.6203
-1.5984 a       -0.5293
-5.6405 aa      -0.1468
-6.1177 aart    -0.1528
-6.1177 aase    -0.1524
-4.6863 ab      -0.2796
-6.1177 aba     -0.1528
-5.8166 aback   -0.1521
-5.1176 abalone -0.2733
-6.1177 abalones -0.1437
 ..........
-5.1634 zumwalt -0.1309
-5.6405 zuniga  -0.1463
-4.9415 zurich  -0.4013
-5.6405 zurkuhlen -0.1290
-6.1177 zwentendorf -0.1528
-6.1177 zwhalan -0.1509
-6.1177 zydeco  -0.1417
-6.1177 zz      -0.1526
-5.4187 zzzz    -0.6612

\2-grams:
-0.0000 </s> <s>
-2.8991 <s> </s>
-1.6152 <s> a
-5.3247 <s> aaron
-5.3247 <s> ab
-4.0924 <s> abc
-5.3247 <s> abdallah
 ..........
-5.3247 <s> zero
-5.3247 <s> zeros
-4.7315 <s> zinc
-4.7315 <s> zondervan
-5.3247 <s> zorinsky
-4.3076 <s> zzzz
-2.8960 a </s>
-2.7199 a a
-3.7359 a acra
-4.4540 a advance
-5.0472 a affecting
-4.0302 a affiliate
-5.0472 a after
-4.4540 a agreed
-4.0302 a agreement
 ..........
-2.6283 win because
-2.6283 win big
-1.7682 win but
-2.6283 win cars
-2.0351 win concessions
-2.6283 win confirmation
-2.6283 win congressional
 ..........
-0.8439 zurich </s>
-0.5840 zurich and
-1.7040 zurich based
-1.7040 zurich raised
-1.7040 zurich said
-0.8439 zurich switzerland
-1.7040 zurich to
-1.0050 zurkuhlen and
-1.0050 zurkuhlen of
-0.5279 zwentendorf austria
-0.5279 zwhalan an
-0.5279 zydeco a
-0.5279 zz top
-0.1069 zzzz best

\end\
```

Statistical Estimators III: Smoothing Techniques: *Laplace*

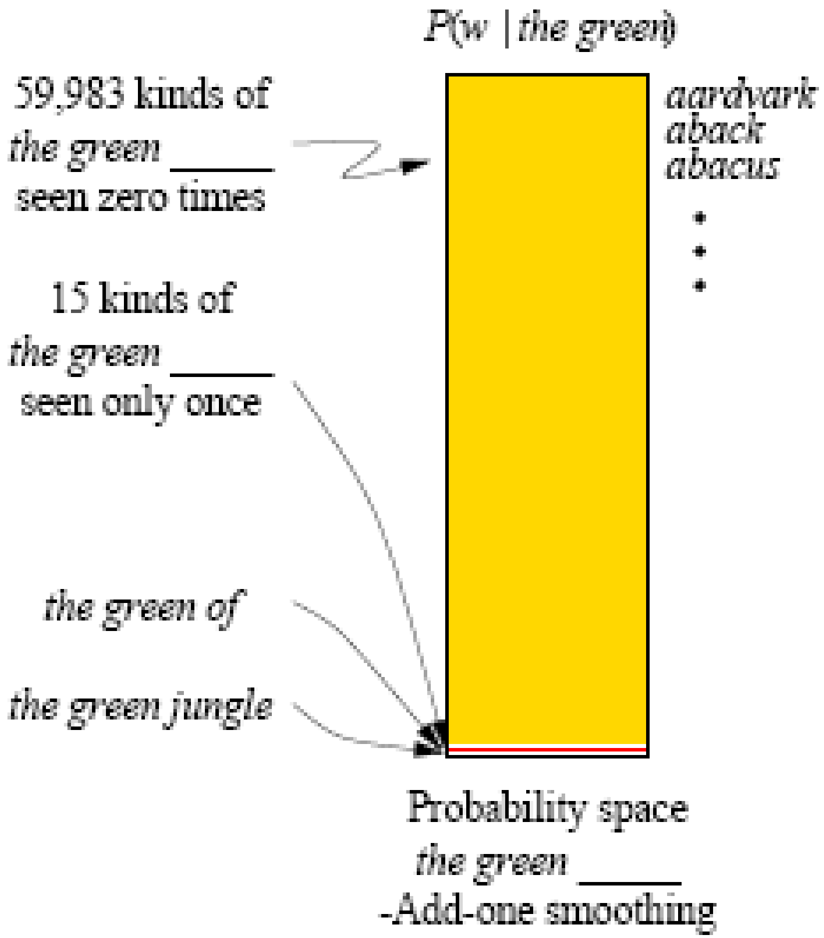
- $P_{LAP}(w_1,..,w_n)=(C(w_1,..,w_n)+1)/(N+B)$, where $C(w_1,..,w_n)$ is the frequency of n-gram $w_1,..,w_n$ and B is the number of bins training instances are divided into. ==> ***Adding One*** Process
- The idea is to give a little bit of the probability space to unseen events.
- However, in NLP applications that are very sparse, Laplace's Law actually gives far too much of the probability space to unseen events.

Example



P(w | the green)
areas
bay
beneath
water
of
jungle
59.983 kinds of
the green ____
seen zero times
15 kinds of
the green ____
seen only once
the green of
the green jungle
P(w | the green)
aardvark
abacus
zucchini
zygote
areas
bay
beneath
water
of
jungle
Probability space for
the green ____
-unsmoothed
Probability space for
the green ____
-smoothed

Example



P(w | the green)
59,983 kinds of
the green ______
seen zero times
15 kinds of
the green ______
seen only once
the green of
the green jungle
aardvark
aback
abacus
Probability space
the green ______
-Add-one smoothing

Statistical Estimators IV: Smoothing Techniques:*Lidstone and Jeffrey-Perks*

- Since the adding one process may be adding too much, we can add a smaller value λ.
- $P_{LID}(w_1,..,w_n)=(C(w_1,..,w_n)+\lambda)/(N+B\lambda)$, where $C(w_1,..,w_n)$ is the frequency of n-gram $w_1,..,w_n$ and B is the number of bins training instances are divided into, and $\lambda>0$. ==> ***Lidstone's Law***
- If $\lambda=1/2$, Lidstone's Law corresponds to the expectation of the likelihood and is called the ***Expected Likelihood Estimation*** (ELE) or the ***Jeffreys-Perks*** Law.

Statistical Estimators V, Robust Techniques: *Held Out Estimation*

- For each n-gram, $w_1,..,w_n$, we compute $C_1(w_1,..,w_n)$ and $C_2(w_1,..,w_n)$, the frequencies of $w_1,..,w_n$ in training and held out data, respectively.
- Let N_r be the number of bigrams with frequency r in the training text.
- Let T_r be the total number of times that all n-grams that appeared r times in the training text appeared in the held out data.
- An estimate for the probability of one of these n-gram is: $P_{ho}(w_1,..,w_n)= T_r/(N_rN)$
  where $C(w_1,..,w_n)= r$.

Statistical Estimators VI: Robust Techniques: *Cross-Validation*

- Held Out estimation is useful if there is a lot of data available. If not, it is useful to use each part of the data both as training data and held out data.
- ***Deleted Estimation*** [Jelinek & Mercer, 1985]: Let N_r^a be the number of n-grams occurring r times in the a^{th} part of the training data and T_r^{ab} be the total occurrences of those bigrams from part a in part b. $Pdel(w_1,..,w_n)= (T_r^{ab}+T_r^{ba})/N(N_r^a+ N_r^b)$ where $C(w_1,..,w_n) = r$.
- ***Leave-One-Out*** [Ney et al., 1997]

Statistical Estimators VI: Related Approach: *Good-Turing Estimator*

- If $C(w_1,..,w_n) = r > 0$, $P_{GT}(w_1,..,w_n) = r^*/N$ where $r^*=(r+1)N_r/r$
- If $C(w_1,..,w_n) = 0$, $P_{GT}(w_1,..,w_n) \approx N_1/(N_0 N)$
- ***Simple Good-Turing*** [Gale & Sampson, 1995]:
- Use a smoothed estimate of the expectation of N_r.
- As a smoothing curve, use $N_r=ar^b$ (with $b < -1$) and estimate a and b by simple linear regression on the logarithmic form of this equation:

  $\log N_r = \log a + b \log r$, if r is large.
- For low values of r, use the measured N_r directly.

Good-Turing Smoothing (example)

- In the Brown Corpus, suppose for n =2, N_2=4000 N_3=2400.
- Then 2* = 3 (2400/4000) = 1.8
- $P_{GT}(jungle|green)$ = 3*/207 = 2.2/207 = 0.01062

Good-Turing Smoothing (example)

- Probability mass left over for unseen events

$$
\begin{aligned}
&= 1 - \sum_{r=1}^{\infty} N_r (r^*/N) \\
&= 1 - 1/N \sum_{r=1}^{\infty} (r+1) N_{r+1} \\
&= 1 - 1/N(N - N_1) \text{ (because } \sum_{r=1}^{\infty} rN_r = N) \\
&= N_1/N
\end{aligned}
$$

Divide this among $N_0 = V^n - \sum_{r=1}^{\infty} N_r$ kinds of unseen events

$C(W_{1,n}) = r = 0$, then

$$P_{GT}(W_{1,n}) = \frac{N_1}{NN_0} = 0^*/N = r^*/N$$

- $P_{GT}(lantern \mid green) = 0^*/207 = N_1/207N$

Combining Estimators I: Overview

- If we have several models of how the history predicts what comes next, then we might wish to combine them in the hope of producing an even better model.
- Combination Methods Considered:
  - Simple Linear Interpolation
  - Katz’s Backing Off
  - General Linear Interpolation

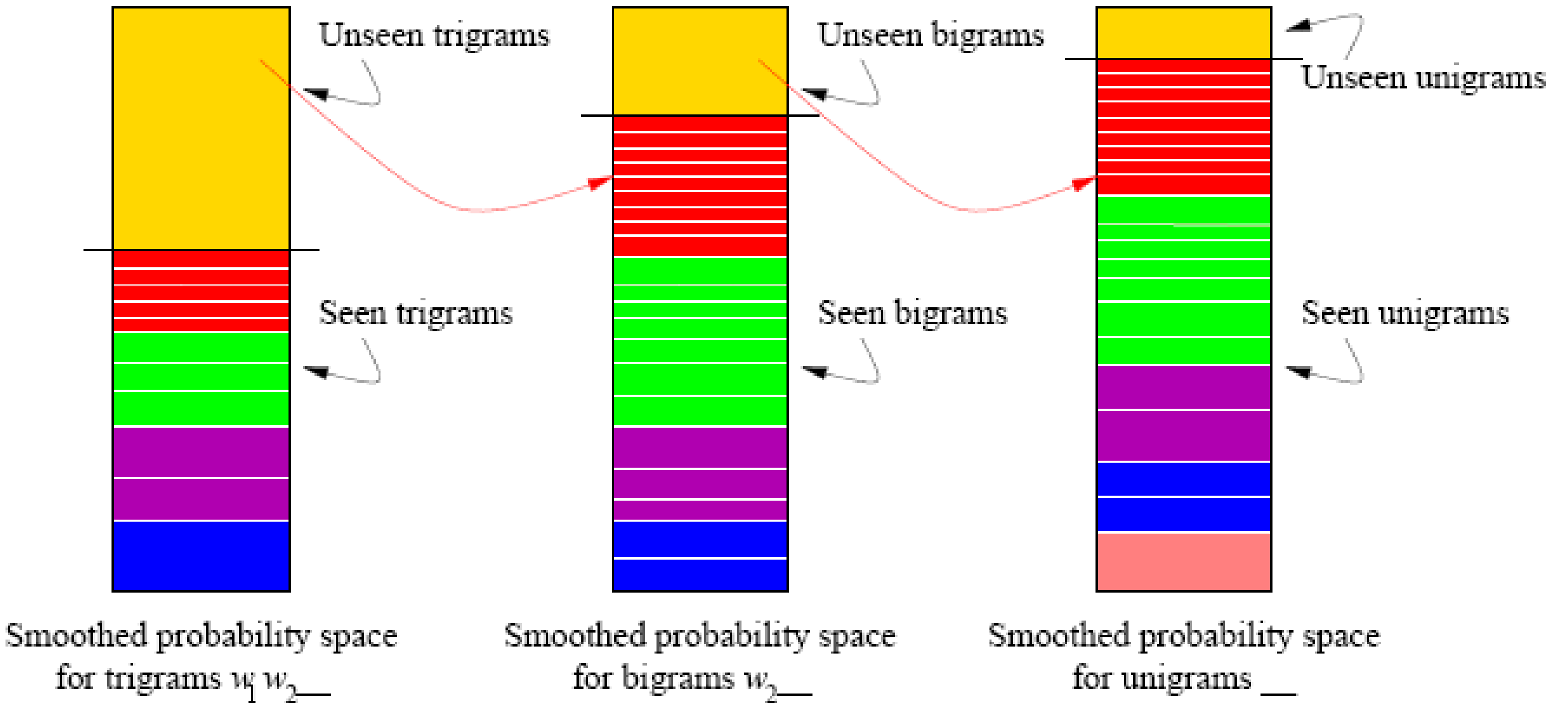
Combining Estimators II: Simple Linear Interpolation

- One way of solving the sparseness in a trigram model is to mix that model with bigram and unigram models that suffer less from data sparseness.
- This can be done by ***linear interpolation*** (also called ***finite mixture models***). When the functions being interpolated all use a subset of the conditioning information of the most discriminating function, this method is referred to as ***deleted interpolation***.
- $P_{li}(w_n|w_{n-2},w_{n-1})=\lambda_1 P_1(w_n)+ \lambda_2 P_2(w_n|w_{n-1})+ \lambda_3 P_3(w_n|w_{n-1},w_{n-2})$ where $0\leq\lambda_i \leq 1$ and $\Sigma_i \lambda_i =1$
- The weights can be set automatically using the Expectation-Maximization (EM) algorithm.

Combining Estimators II: Katz's Backing Off Model

- In back-off models, different models are consulted in order depending on their specificity.
- If the n-gram of concern has appeared more than k times, then an n-gram estimate is used but an amount of the MLE estimate gets discounted (it is reserved for unseen n-grams).
- If the n-gram occurred k times or less, then we will use an estimate from a shorter n-gram (back-off probability), normalized by the amount of probability remaining and the amount of data covered by this estimate.
- The process continues recursively.

Katz's Backing Off Model (3-grams)



$$P_{BO}(w_3 \mid w_1 w_2) = \begin{cases} P_S(w_3 \mid w_1 w_2) & \text{if } C(W_{1,3}) > k \\ \alpha(w_1 w_2) P_S(w_3 \mid w_2) & \text{if } C(W_{1,3}) \leq k \text{ and } C(W_{2,3}) > k \\ \alpha(w_2) P_S(w_3) & \text{otherwise} \end{cases}$$

Katz's Backing Off Model (2-grams)

- For bigrams:

$$P_{BO}(w_2|w_1) = \begin{cases} P_S(w_2|w_1) & \text{if } C(w_1 w_2) > k \\ \alpha(w_1) P_S(w_2) & \text{otherwise} \end{cases}$$

$$\alpha(w_1) = \frac{1 - \sum_{w_2 : C(w_1 w_2) > 0} P_S(w_2|w_1)}{1 - \sum_{w_2 : C(w_1 w_2) > 0} P_S(w_2)}$$

Combining Estimators II: General Linear Interpolation

- In simple linear interpolation, the weights were just a single number, but one can define a more general and powerful model where the weights are a function of the history.
- For k probability functions Pk, the general form for a linear interpolation model is:

$P_{li}(w|h) = \Sigma_i^k \lambda_i(h) P_i(w|h)$ where $0 \leq \lambda_i(h) \leq 1$ and $\Sigma_i \lambda_i(h) = 1$