CMPT-825
Natural Language Processing

Anoop Sarkar
http://www.cs.sfu.ca/∼anoop

October 8, 2010

Log Probability

Basics of Information Theory

Log Probability Arithmetic

- Practical problem with tiny $P(e)$ numbers: underflow
- One solution is to use log probabilities:

$$\begin{aligned}\log(P(e)) &= \log(p_1 \times p_2 \times \ldots \times p_n)\\ &= \log(p_1) + \log(p_2) + \ldots + \log(p_n)\end{aligned}$$

- Note that:

$$x = \exp(\log(x))$$

- Also more efficient: addition instead of multiplication

Log Probability Arithmetic

p	$\log(p)$
0.0	$-\infty$
0.1	-3.32
0.2	-2.32
0.3	-1.74
0.4	-1.32
0.5	-1.00
0.6	-0.74
0.7	-0.51
0.8	-0.32
0.9	-0.15
1.0	0.00

Log Probability Arithmetic

- So: $(0.5 \times 0.5 \times \ldots 0.5) = (0.5)^n$ might get too small but $(-1 - 1 - 1 - 1) = -n$ is manageable
- Another useful fact when writing code ($\log_2$ is *log to the base 2*):

$$\log_2(x) = \frac{\log_{10}(x)}{\log_{10}(2)}$$

Log Probability Arithmetic

- Adding probabilities is expensive to compute:
  $logadd(x, y) = \log(\exp(x) + \exp(y))$
- A more efficient soln, let *big* be a large constant e.g. 10^{30}:

  function $logadd(x, y)$: # returns $\log(\exp(x) + \exp(y))$
  if $(y - x) > \log(big)$ return y
  elsif $(x - y) > \log(big)$ return x
  else return
  $$min(x, y) + \log(\exp(x - min(x, y)) + \exp(y - min(x, y)))$$
  endif
- There is a more efficient way of computing
  $\log(\exp(x - min(x, y)) + \exp(y - min(x, y)))$

Log Probability Arithmetic

function $logadd(x, y)$:
    if $(y - x) > \log(big)$ return y
    elsif $(x - y) > \log(big)$ return x
    elsif $(x \geq y)$ return $x + \log(1 + \exp(y - x))$
        $\#$ note that $max(x, y) = x$ and $y - x \leq 0$
    else return $y + \log(\exp(x - y) + 1)$
        $\#$ note that $max(x, y) = y$ and $x - y \leq 0$
    endif

Also, in ANSI C, `log1p` efficiently computes $\log(1 + x)$

`http://www.ling.ohio-state.edu/~jansche/src/logadd.c`

Log Probability

Basics of Information Theory

Information Theory

- Information theory is the use of probability theory to quantify and measure "information".
- Consider the task of efficiently sending a message. Sender Alice wants to send several messages to Receiver Bob. Alice wants to do this as efficiently as possible.
- Let's say that Alice is sending a message where the entire message is just one character a, e.g. $aaaa\ldots$. In this case we can save space by simply sending the length of the message and the single character.

Information Theory

- Now let's say that Alice is sending a completely random signal to Bob. If it is random then we cannot exploit anything in the message to compress it any further.
- The *upper bound* on the number of bits it takes to transmit some infinite set of messages is what is called entropy.
- This formulation of entropy by Claude Shannon was adapted from thermodynamics, converting information into a quantity that can be measured.
- Information theory is built around this notion of message compression as a way to evaluate the amount of information.

Entropy

- Consider a probability distribution p
- Entropy of p is:

$$H(p) = -\sum_{x \in \mathcal{E}} p(x) \log_2 p(x)$$

- Any base can be used for the log, but base 2 means that entropy is measured in bits.
- Entropy answers the question: What is the upper bound on the number of bits needed to transmit messages from event space $\mathcal{E}$, where $p(x)$ defines the probability of observing x.

Entropy

- Alice wants to bet on a horse race. She has to send a message to her bookie Bob to tell him which horse to bet on.
- There are 8 horses. One encoding scheme for the messages is to use a number for each horse. So in bits this would be $001, 010, \ldots$
  (lower bound on message length $=$ 3 bits in this encoding scheme)
- Can we do better?

Entropy

Horse 1	$\frac{1}{2}$	Horse 5	$\frac{1}{64}$
Horse 2	$\frac{1}{4}$	Horse 6	$\frac{1}{64}$
Horse 3	$\frac{1}{8}$	Horse 7	$\frac{1}{64}$
Horse 4	$\frac{1}{16}$	Horse 8	$\frac{1}{64}$

- If we know how likely we are to bet on each horse, say based on the horse's probability of winning, then we can do better.
- Let p be the probability distribution given in the table above. The entropy of p is $H(p)$

Entropy

$$
\begin{aligned}
H(p) = & \\
= & -\sum_{i=1}^{8} p(i) \log_2 \; p(i) \\
= & -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{4}\log_2\frac{1}{4} + \frac{1}{8}\log_2\frac{1}{8} + \frac{1}{16}\log_2\frac{1}{16} + 4(\frac{1}{64}\log_2\frac{1}{64})\right) \\
= & -\left(\frac{1}{2}\times -1 + \frac{1}{4}\times -2 + \frac{1}{8}\times -3 + \frac{1}{16}\times -4 + 4(\frac{1}{64}\times -6)\right) \\
= & -\left(-\frac{1}{2} - \frac{1}{2} - \frac{3}{8} - \frac{1}{4} - \frac{3}{8}\right) \\
= & \; 2 \; bits
\end{aligned}
$$

- What is the entropy when the horses are equally likely to win?

$$H(uniform\ distribution) = -8(\frac{1}{8}\times -3) = 3\ bits$$

Entropy

- e.g., most likely horse gets code 0, next most likely gets 10, and then $110, 1110, \ldots$
  many possible coding schemes, this is a simple code to illustrate number of bits needed for a large number of messages ...
- Assume there are 320 messages (one for each race): code 0 occurs 160 times, code 10 occurs 80 times, code 110 occurs 40 times, code 1110 occurs 20 times, code 11110 occurs 5 times.
- Total number of bits for all messages: 160*len(0) + 80*len(10) + 40*len(110) + 20*len(1110) + 5*len(11110)
- Number of bits: $160*1 + 80*2 + 40*3 + 20*4 + 5*5 = 545$
- Total number of bits per message (per race): $\frac{545}{320} \approx 1.7$ bits (always less than 2 bits)

Perplexity

- The value $2^{H(p)}$ is called the **perplexity** of a distribution p
- Perplexity is the weighted average number of choices a random variable has to make.
- Choosing between 8 equally likely horses (H=3) is $2^3 = 8$.
- Choosing between the biased horses from before (H=2) is $2^2 = 4$.

Relative Entropy

- In real life, we cannot know for sure the exact winning probability for each horse.
- Let's say q is the estimate and p is the true probability (say we got q by observing previous races with these horses)
- We define the *distance* between q and p as the **relative entropy**: written as $D(q\|p)$

$$D(q\|p) = -\sum_{x \in \mathcal{E}} q(x) \log_2 \frac{p(x)}{q(x)}$$

- Note that

$$D(q\|p) = E_{q(x)}\left[\log_2 \frac{p(x)}{q(x)}\right]$$

- The relative entropy is also called the *Kullback-Leibler divergence*.

Cross Entropy and Relative Entropy

- The **relative entropy** can be written as the sum of two terms:

$$
\begin{aligned}
D(q\|p) &= -\sum_{x\in\mathcal{E}} q(x)\log_2 \frac{p(x)}{q(x)} \\
&= -\sum_x q(x)\log_2 p(x) + \sum_x q(x)\log_2 q(x)
\end{aligned}
$$

- We know that $H(q) = -\sum_x q(x)\log_2 q(x)$
- Similarly define $H_q(p) = -\sum_x q(x)\log_2 p(x)$

$$
D(q\|p) = H_q(p) - H(q)
$$

- The term $H_q(p)$ is called the **cross entropy**.

Cross Entropy and Relative Entropy

- The **relative entropy** between p and q can be written as the sum of two terms:

$$\begin{array}{lll} \textbf{relative entropy}(q,p) & = \textbf{cross entropy}(q,p) & - \textbf{entropy}(q) \\ D(q\|p) & = H_q(p) & - H(q) \end{array}$$

- $H_q(p) \geq H(q)$ always.
- $D(q\|p) \geq 0$ always, and $D(q\|p) = 0$ iff $q = p$
- $D(q\|p)$ is not a true distance:
  - It is asymmetric: $D(q\|p) \neq D(p\|q)$,
  - It does not obey the triangle inequality: $D(p\|r) \not\leq D(p\|q) + D(q\|r)$
- Pinsker's inequality (sup is the lowest upper bound):

$$\sqrt{\frac{D(q\|p)}{2}} \geq \sup\{|q(x) - p(x)|\}$$

Conditional Entropy and Mutual Information

- *Entropy* of a random variable X:

$$H(X) = -\sum_{x \in \mathcal{E}} p(x) \log_2 p(x)$$

- *Conditional Entropy* between two random variables X and Y:

$$H(X \mid Y) = -\sum_{x,y \in \mathcal{E}} p(x, y) \log_2 p(x \mid y)$$

- *Mutual Information* between two random variables X and Y:

$$I(X; Y) = D(p(x, y) \| p(x)p(y)) = \sum_{x} \sum_{y} p(x, y) \log_2 \frac{p(x, y)}{p(x)p(y)}$$