Foundations of Natural Language Processing
Lecture 6b
Language Models: Evaluation II

Alex Lascarides

School of informatics

Recap

How do we evaluate Language Models?

- **Language models** tell us $P(\vec{w}) = P(w_1 \ldots w_n)$.

- We can't evaluate an LM with accuracy metrics.

- Entropy, however, measures confidence in the model's predictions of a random variable X:
$$H(X) = -\sum_{x \in X} Pr(x) \log_2(x)$$

Now: Evaluation II

- What entropy 'means'

- Details on how to use entropy to evaluate LMs.

Entropy as y/n questions (hence log_2)

How many yes-no questions (bits) do we need to find out the outcome?

- Uniform distribution with 2^n outcomes: n yes-no questions.

Entropy as encoding sequences

- Assume that we want to encode a sequence of events X.
- Each event is encoded by a sequence of bits, we want to use as few bits as possible.
- For example
  - Coin flip: heads $=$ 0, tails $=$ 1
  - 4 equally likely events: a $=$ 00, b $=$ 01, c $=$ 10, d $=$ 11
  - 3 events, one more likely than others: a $=$ 0, b $=$ 10, c $=$ 11
  - Morse code: *e* has shorter code than *q*
- Average number of bits needed to encode $X \geq$ entropy of X

The Entropy of English

- Given the start of a text, can we guess the next word?
- For humans, the measured entropy is only about 1.3.
  - Meaning: on average, given the preceding context, a human would need only 1.3 y/n questions to determine the next word.
  - This is an upper bound on the true entropy, which we can never know (because we don't know the true probability distribution).
- But what about N-gram models?

Coping with not knowing true probs: Cross-entropy

- Our LM *estimates* the probability of word sequences.

- A good model assigns high probability to sequences that actually have high probability (and low probability to others).

- Put another way, our model should have low uncertainty (entropy) about which word comes next.

- **Cross entropy** measures how close $\hat{P}$ is to true P:

  $H(P, \hat{P}) = \sum_x -P(x)\ \log_2 \hat{P}(x)$

- Note that cross-entropy $\geq$ entropy: our model's uncertainty can be no less than the true uncertainty.

- But still dont know $P(x)$...

Coping with Estimates: Compute per word cross-entropy

- For $w_1 \ldots w_n$ with large n, per-word cross-entropy is well approximated by:

$$H_M(w_1 \ldots w_n) = -\frac{1}{n}\log_2 P_M(w_1 \ldots w_n)$$

- This is just the average negative log prob our model assigns to each word in the sequence. (i.e., normalized for sequence length).

- Lower cross-entropy $\Rightarrow$ model is better at predicting next word.

Cross-entropy example

Using a bigram model from Moby Dick, compute per-word cross-entropy of I spent three years before the mast (here, without using end-of sentence padding):

$$
\begin{aligned}
& -\tfrac{1}{7}(\quad \lg_2(P(\textit{I})) + \lg_2(P(\textit{spent}|\textit{I})) + lg_2(P(\textit{three}|\textit{spent})) + \lg_2(P(\textit{years}|\textit{three})) \\
& \qquad + \lg_2(P(\textit{before}|\textit{years})) + \lg_2(P(\textit{the}|\textit{before})) + \lg_2(P(\textit{mast}|\textit{the})) \quad) \\
= \; & -\tfrac{1}{7}(\quad -6.9381 - 11.0546 - 3.1699 - 4.2362 - 5.0 - 2.4426 - 8.4246 \quad) \\
= \; & -\tfrac{1}{7}(\quad 41.2660 \quad) \\
\approx \; & \quad 6
\end{aligned}
$$

- Per-word cross-entropy of the *unigram* model is about 11.

- So, unigram model has about 5 bits more uncertainty per word then bigram model. But, what does that mean?

Data compression

- If we designed an optimal code based on our bigram model, we could encode the entire sentence in about 42 bits. 6*7

- A code based on our unigram model would require about 77 bits. 11*7

- ASCII uses an average of 24 bits per word (168 bits total)!

- So better language models can also give us better data compression: as elaborated by the field of **information theory**.

Perplexity

- LM performance is often reported as **perplexity** rather than cross-entropy.
- Perplexity is simply $2^{\text{cross-entropy}}$
- The average branching factor at each decision point, if our distribution were uniform.
- So, 6 bits cross-entropy means our model perplexity is $2^6 = 64$: equivalent uncertainty to a uniform distribution over 64 outcomes.

Perplexity looks different in J&M 3^{rd} edition because they don't introduce cross-entropy; I'll accept either answers!

Interpreting these measures

I measure the cross-entropy of my LM on some corpus as 5.2.
Is that good?

Interpreting these measures

I measure the cross-entropy of my LM on some corpus as 5.2.
Is that good?

- No way to tell! Cross-entropy depends on both the model and the corpus.
  - Some language is simply more predictable (e.g. casual speech vs academic writing).
  - So lower cross-entropy could mean the corpus is "easy", or the model is good.

- We can only compare different models on the same corpus.

- Should we measure on training data or held-out data? Why?

Summary

- LMs can be evaluated using per word cross entropy.
- Intuitively, this is a measure of:
  - The LMs confidence in its predictions about the next word (averaged over the sequence).
  - The extent to which it has compressed the data necessary for making those predictions.
- But this measure is informative only when comparing the per word cross entropy of two different LMs
  - We don't have meaningful upper bounds.