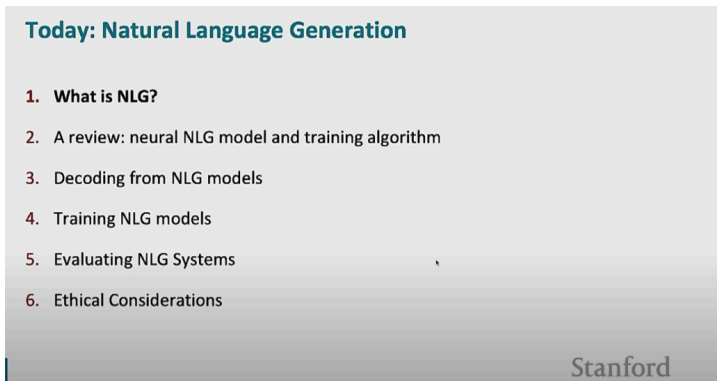
Notes

20 June 2024 22:26

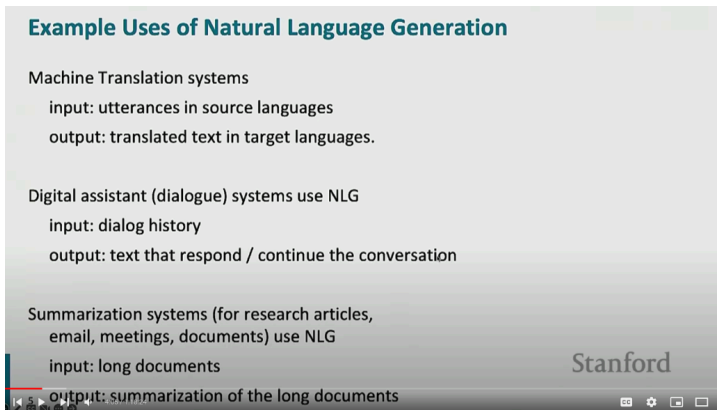
Outline:

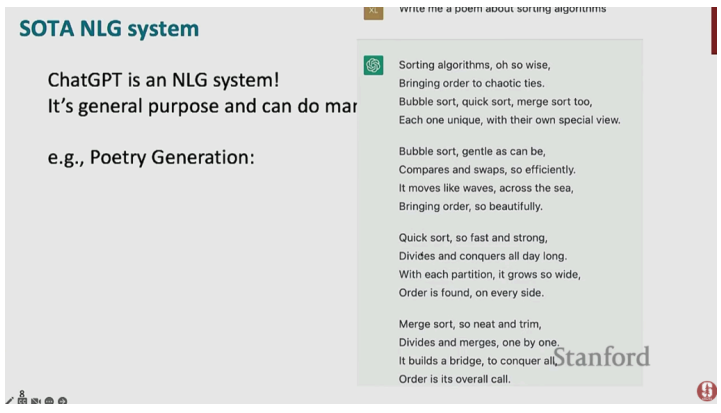


What is Natural language Generation?

- NLP = Natural Language Understanding + Natural Language Generation
- NLG deals with part of the system that generates the text output

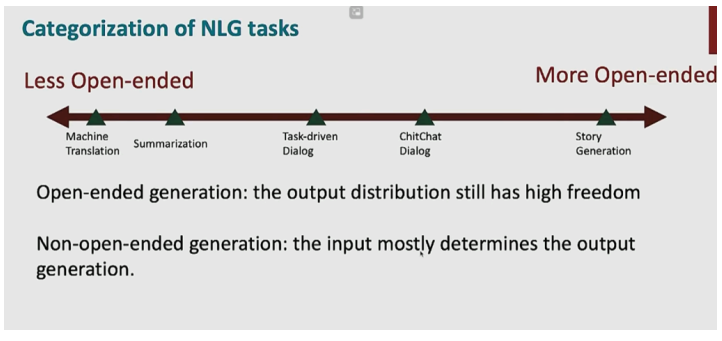
Examples of Natural Language Generation:





Categorization of NLG tasks:

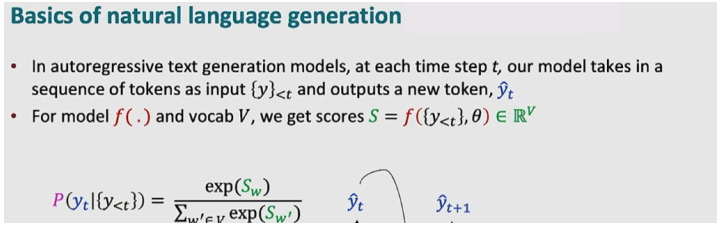
- Categories wrt. The openness of the task – how much dependency an NLG system has on the input to generate the output?
- e.g. Machine translation is closed end task – as it is almost fully dependent on the input
- Summarization is a bit more open ended than translation – as there is scope for creativity but semantics of inputs are still very much necessary.
- Chat-got or story generation are bit more open ended tasks as they are lot of possibilities to diverge from the input semantics

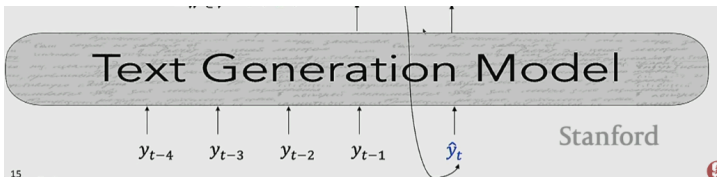


- More open ended high entropy

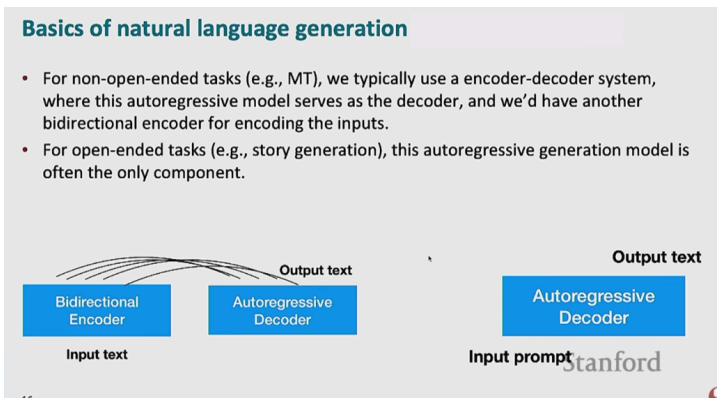
Review of NLG:

- Autoregressive generation: mostly used for open ended task

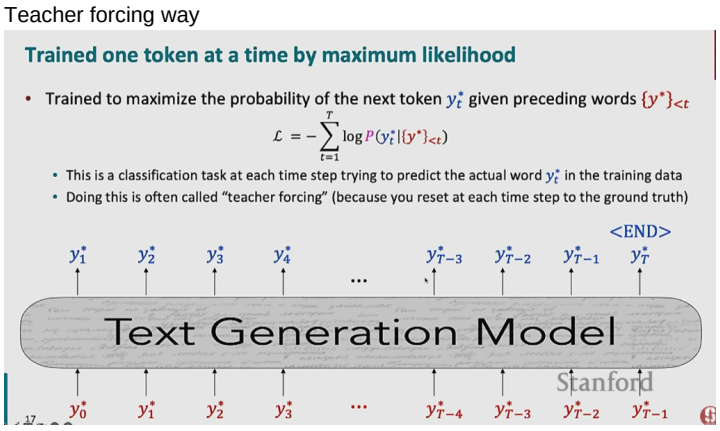




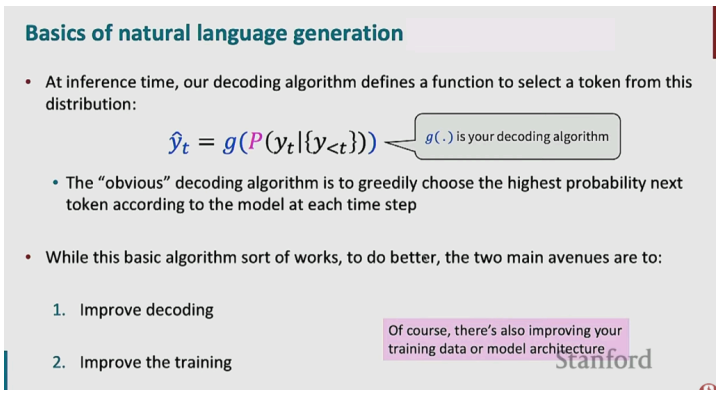
- Encoder-decoder tasks: mostly used by close ended tasks



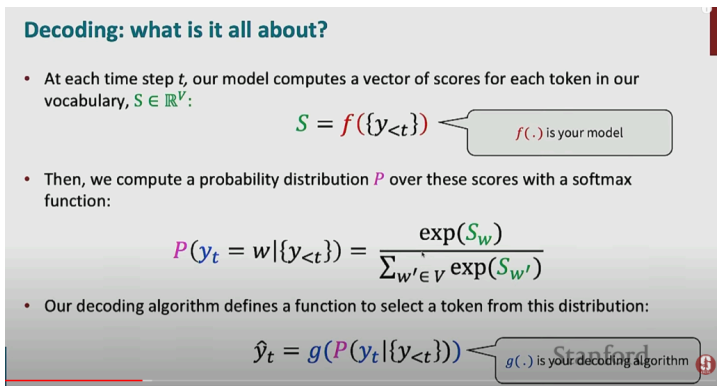
- Training of text generation models:
  - Teacher forcing way



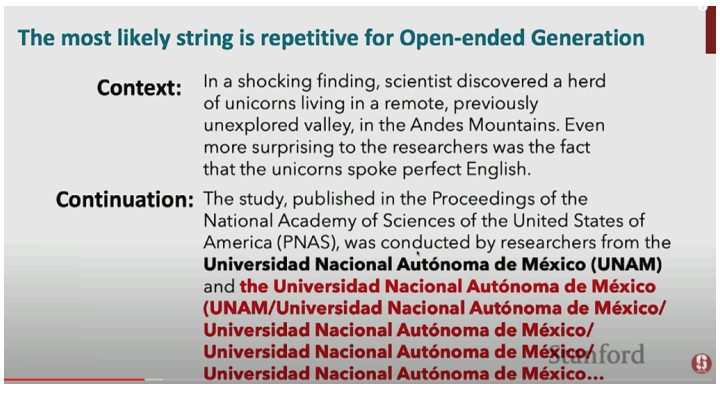
- Decoding:



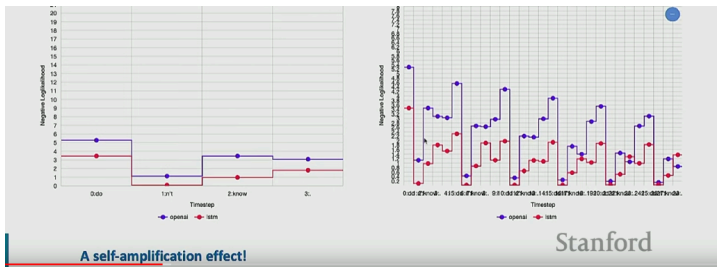
Decoding from NLG models:



- From Lecture 7 – recall greedy decoding and beam search. These both techniques directly or indirectly are trying to generate the token with highest probability.
  - Such decoding techniques are good for close ended tasks.
  - For open ended task, they end up producing the repetitive tokens

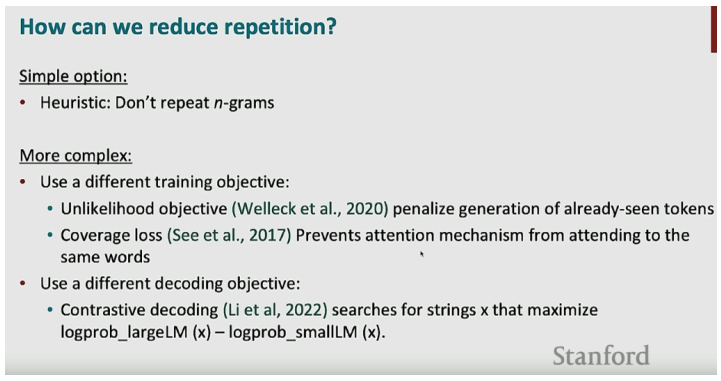


  - Why does this repetition happen?
    - Model generates the "best applicable" token at current moment without considering previous/subsequent tokens. As token gets repeated the negative log likelihood decreases /probability of that token increases.

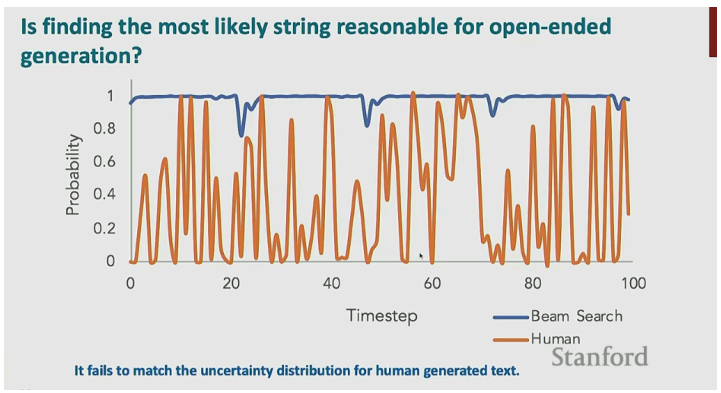


- How to reduce repetition?
  - Use n-grams blocking: e.g. if model has decoded I'm happy previously and now you see I'm being decoded the simply block "happy" being decoded. This is not always feasible.

  - Use different training objectives:
    - Unlikelihood loss: penalize the generation of already-seen token during training
    - Coverage loss: repetition happens because model has similar attention patterns over and over again. Coverage loss prevents the model attending to same words
  - Different decoding objective:
    - Contrastive decoding:



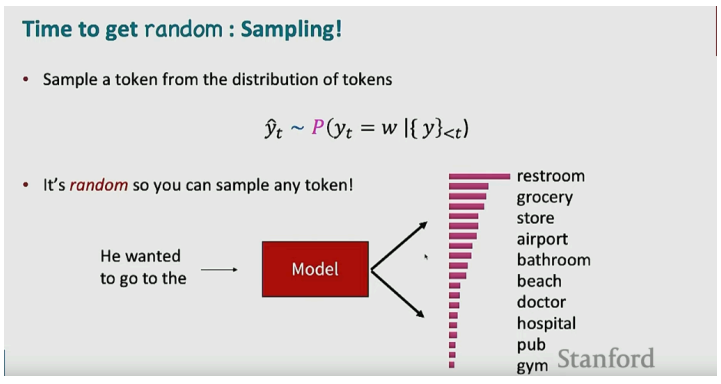
- Is greedy-type of decoding the best way for open ended tasks? ---- NO
  - Here you can see that human "generation" had varying probabilities across the time but the model was always sure about the next word.



  - If not greedy decoding then what???

(random) Sampling based decoding:

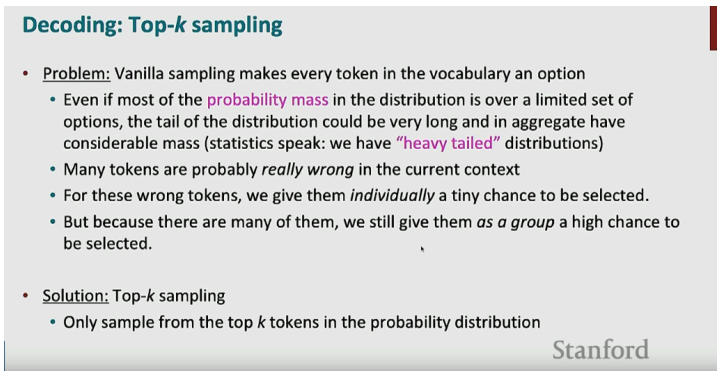
- Greedy decoding would select restroom here (assuming restroom has highest probability) but with random sampling now any of the options from vocab are open

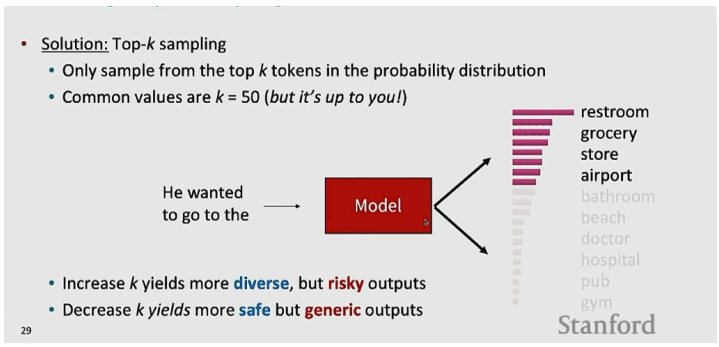


- Problems with Sampling: We never zero out any tokens so all the tokens are valid options!!!!! Even undesired/wrong words can be thrown out by the model. ---- Use top-K sampling

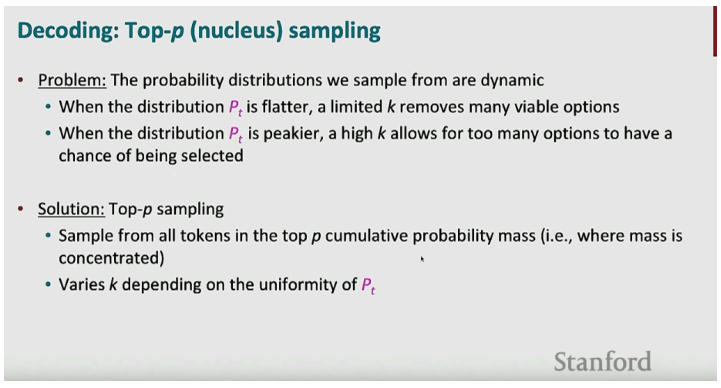
- Top-K sampling selects the next token randomly from top-K token only and neglects other tokens having very low prob. (cuts out the tail)

- K is small for close ende tasks and large for open ended tasks

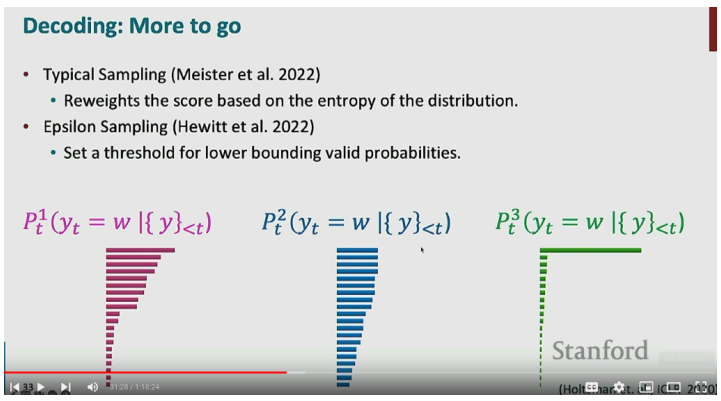




- Is top K sampling good? Nah, not really. It suffers from bad recall/precision depending upon the probability distribution produced by the model.
    - Deciding good k value is difficult. We can instead use probability information to decide K for each token...

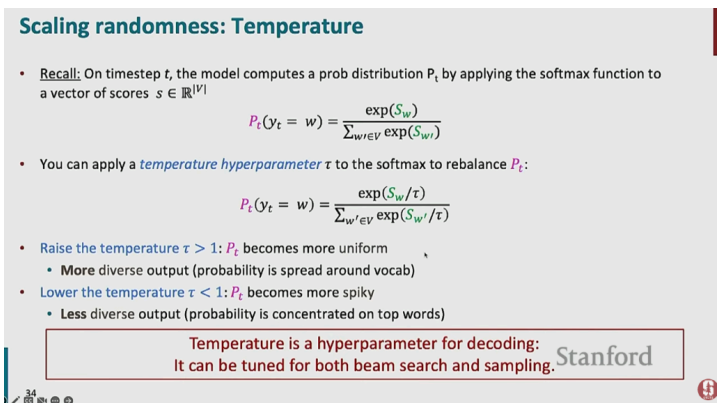


- Other decoding algorithms:
    - Typical sampling: try to generate the tokens with probability closer to the negative entropy of the data distribution.
    - Epsilon Sampling: limit the probabilities being assigned by the model, hence tokens with prob lower than the bound will not be an option to when choosing the next token

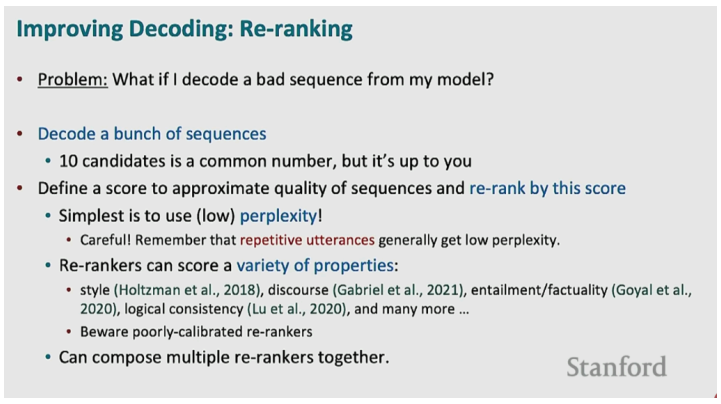


Scaling Randomness: Temperature Parameter

- Temperature parameter does not change the absolute probabilities values. It only scales the relative values I.e. difference in probability values might be more/less depending upon the temperature



Improving decoding by Reranking

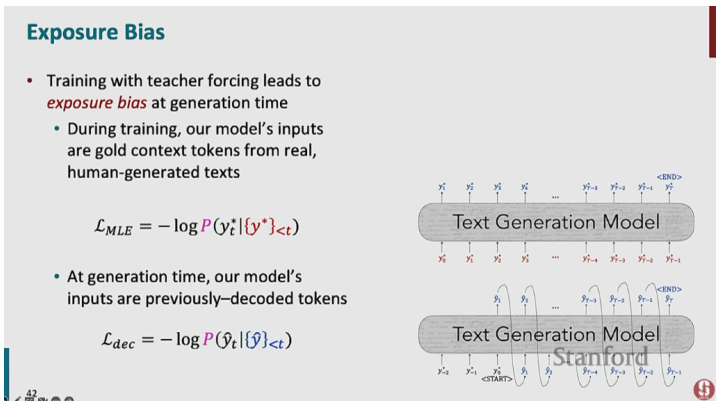


Training the NLG systems

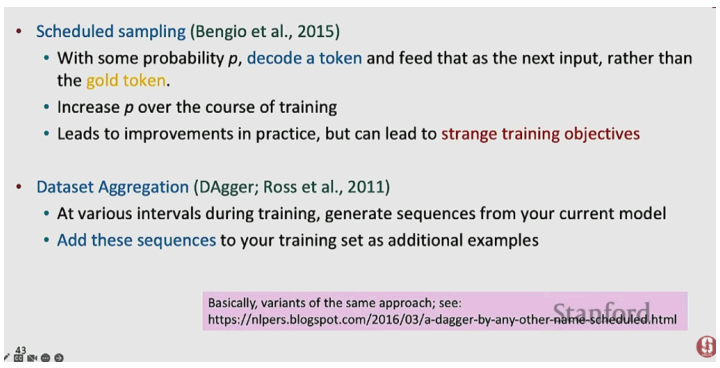
Repetition (during decoding) and Training methodology:

- Models learnt with techer forcing tend to focus on Mode of the distribution I.e. such models assign high probability to the most repeating string in the dataset. Why does this happen?
  --- due to exposure bias

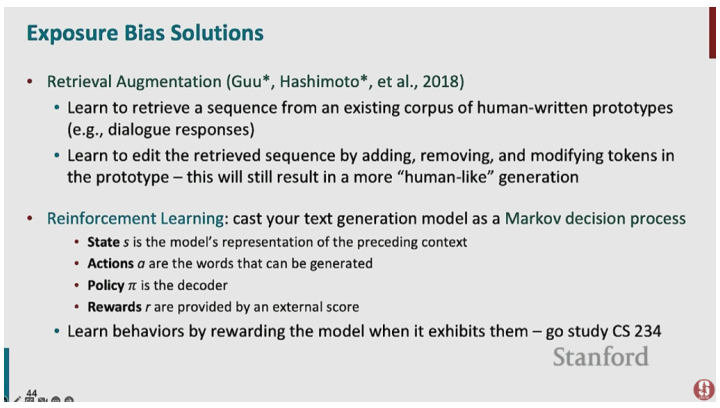
- Exposure Bias:
  - Discrepancy of what model sees during training and testing



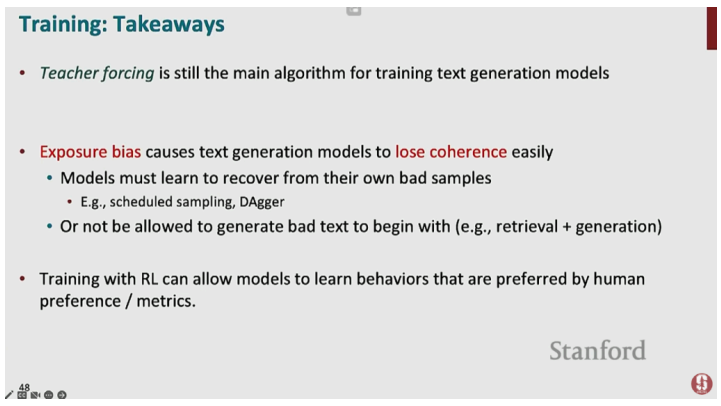
  - To reduce exposure bias:



  - (the favorite solution) Retrieval Augmented Generation:
    - For RAG: (sort of) gold standard tokens are also available during decoding



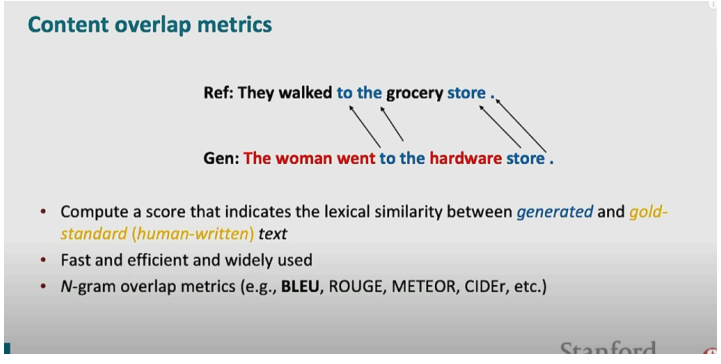
Training: Takeaways



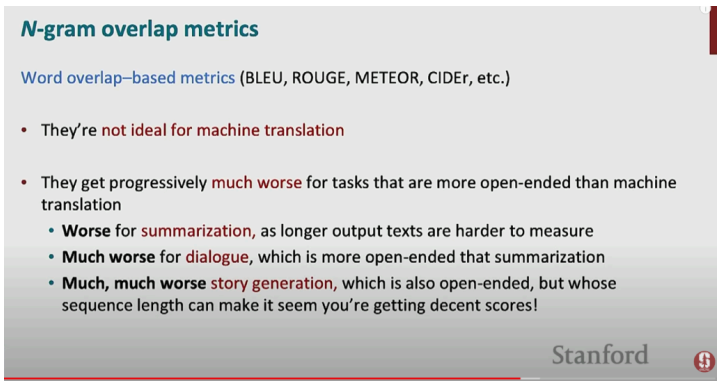
Evaluating NLG Systems

- Mainly 3 methods
  - Content Overlap
  - Model-based metrics
  - Human Evaluation

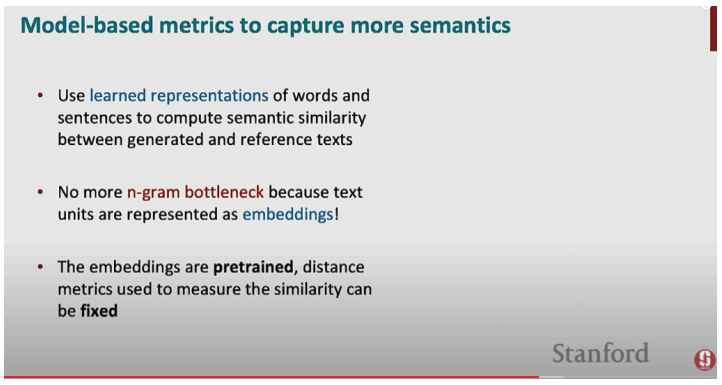
Content Overlap:



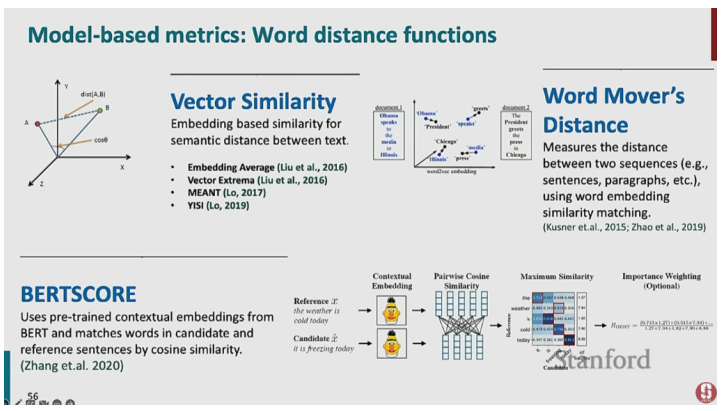
- Drawbacks of N-gram Overlap metrics:
    - They don't have a concept of semantic relatedness. I.e. If correct answer to the question is "Heck Yes" then "Heck No" is also rated as correct because of the overlap of "heck".
    - Similarly, "yup" will be considered as wrong because of no overlap with "heck yes"



Model-based metrics:

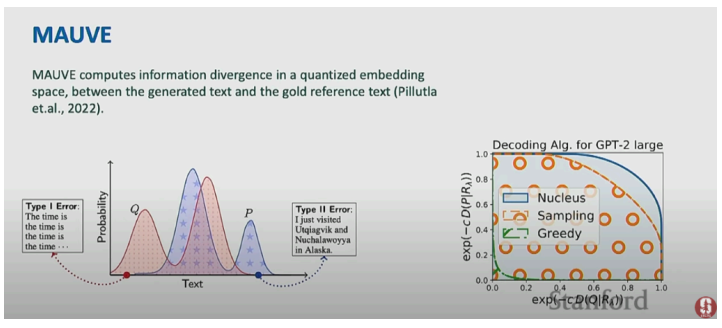


- How to go from word embeddings to sentence embedding?
    - Averaging the word embeddings (yikes!!)



Evaluating open-ended NLG systems:

MAUVE Score:



- (Continuous) embeddings are discretized by using e.g. KNN. In such discretized space, we can then plot histograms, find TP/FP etc