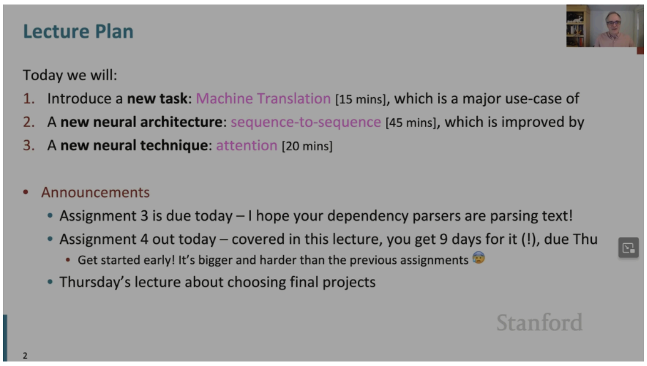
Notes

28 March 2024 08:10

Outline:

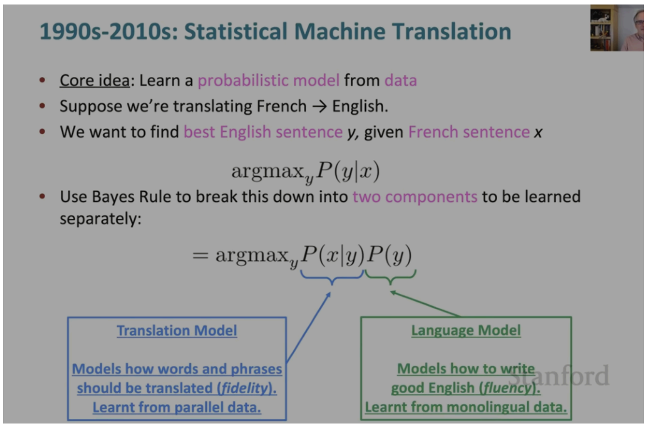


Machine Translation:

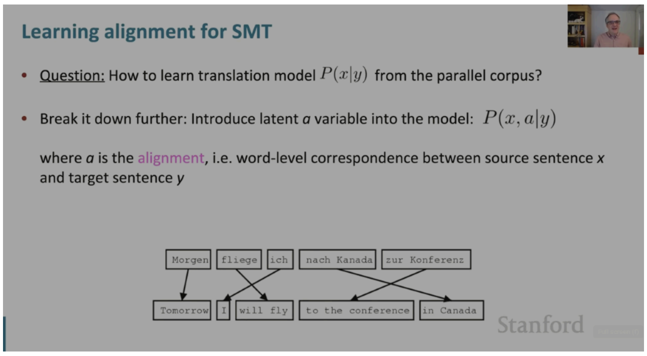
- Task of translating a sentence from one language (the source language) to another language (the target language)

Pre-Neural formulation:

- Prerequisites: human translated parallel data (same data in source and target languages)

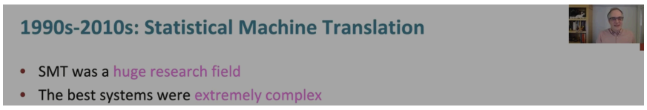


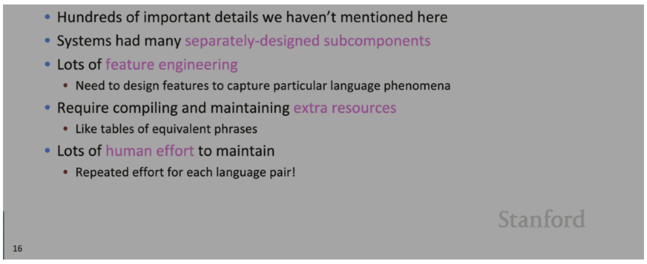
- Using the alignment variable – to establish word-level correspondence between source and target language sentences



- Issues with alignment: Alignment can be one-to-many, many-to-one. Such alignments are defined by a parameter fertility.

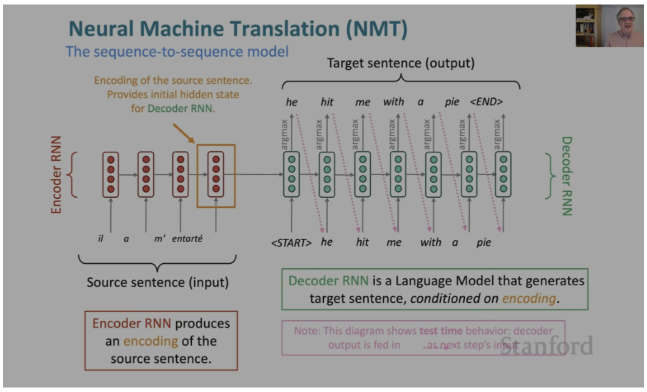
- Also some words from source language might not exist in target language and vice versa. e.g. Le Japon from French is Japan in English



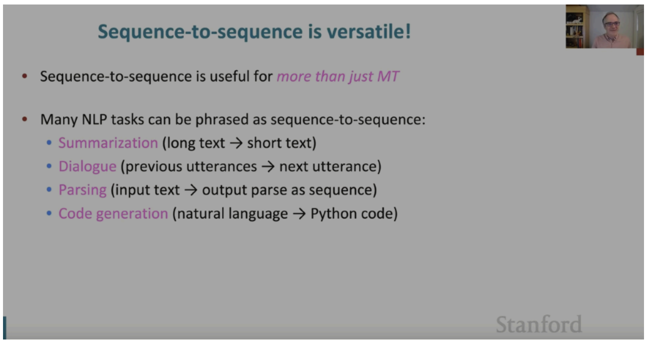


Neural Machine Translation:

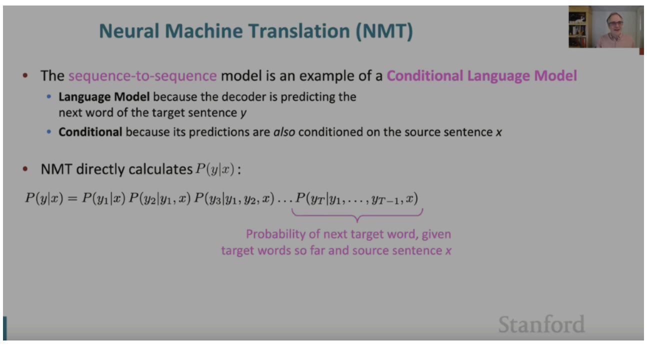
- A way to do translation using a single end-to-end neural network.
- The most widely used neural network(NN) architecture is called as Sequence to Sequence model I.e. using two RNNs – one NN to encode the source sentence and another to decode the sentence in target language
- Below is an overview of seq2seq model during inference



- During training: The architecture stays same. Except the decoder is not being used in autoregressive manner but the decoder is also trained in teacher forcing manner.



Seq2seq model is a Conditional Language Model

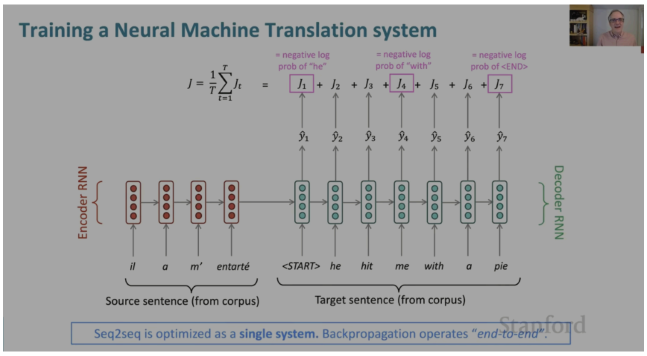


Training of NMT systems

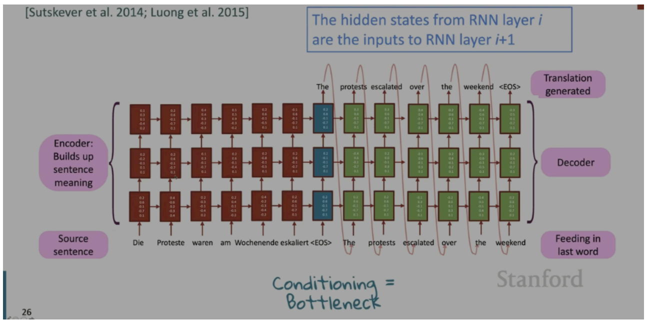
- (Once again) Prerequisites: human translated parallel data
- Build batches: N Source sentences + N corresponding target sentences

- Feed source sentence to encoder. Use the last hidden stage of encoder as input to decoder

- Train decoder in a teacher forcing manner by comparing decoder predictions at each time steps vs the GT target language sentence



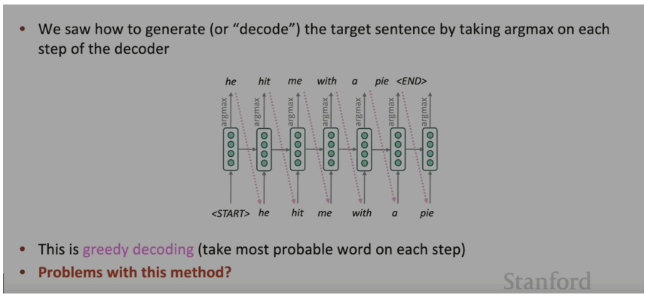
- Single encoder-Decoder RNNs did not work well in practice. Hence multiple (stacked) RNNs were used as encoder + decoder. Below is a representation of such a stacked RNN-based NMT system



Decoding:

- Generating the words in target language using the context given by the encoder (more generally – predicting next word using encoder context)

- Until now – we take argmax at each timestep in decoder. This is called as Greedy Decoding.



- Problems with Greedy Decoding: Settle for locally optimal prediction (word). No way to correct errors / consider global

Beam Search:

- Retain the top-k most probable outputs at each step of decoder (outputs are called as hypothesis). Where k is the beam size

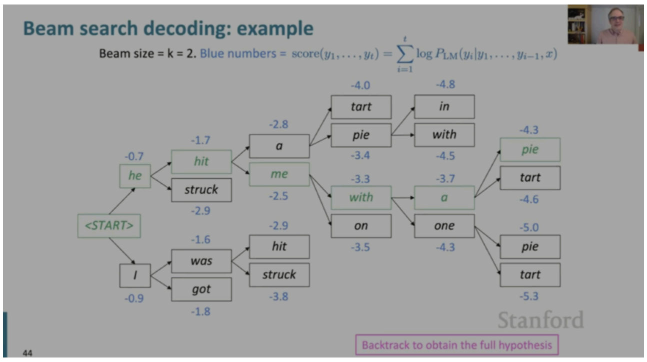
- Each hypothesis is scored by assigning a log probability
  - Hypothesis scores are negative
  - Scores are added from along the time steps are we progress

- At each time steps, num_hypothesis=k are retained and their output is fed to the model in autoregressive manner

- The decoding processes is carried out until all hypothesis return an END token or they reach a pre-determined length

- At the end, we trace back through the tree to obtain the hypothesis with highest scores.

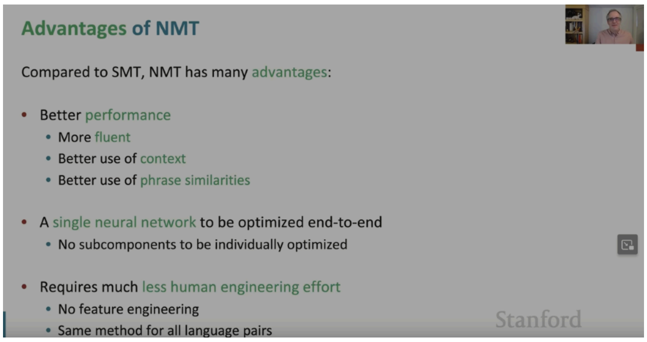
- Not guaranteed to produce optimal results



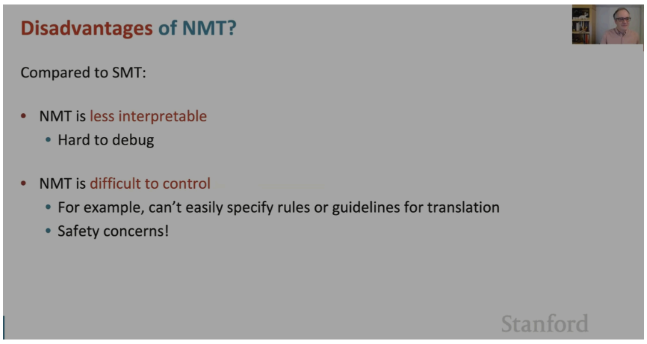
- Stopping Criteria:
  - Different hypothesis might produce END token at different timestamps.
  - Keep the "finished hypothesis" aside and continue with others until they produce END token or a pre-determined length is reached

- Scoring of completed hypothesis:
  - As said above: as hypothesis scores are negative – longer hypothesis will have lower scores. -- We can't just add up the local scores and then select a hypothesis
  - Normalize the scores wrt. The length of the hypothesis

Advantages of NMT

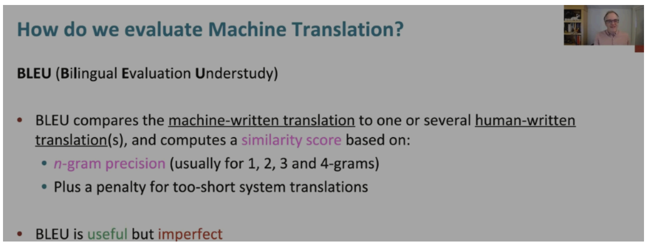


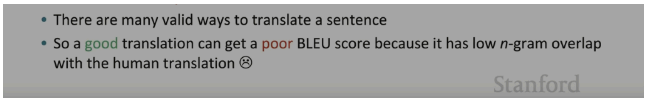
Disadvantages of NMT



Evaluating NMT Systems

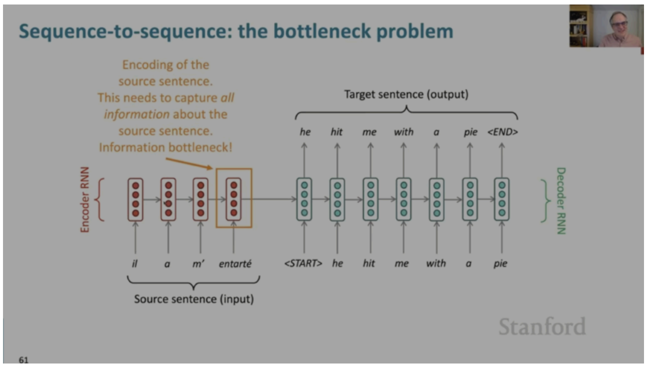
- Bilingual Evaluation Understudy
  - N-gram precision: overlap between machine translation and human translation when using one word window(1-gram), two words window (2-gram) and so on





Attention

- Information bottleneck in RNN Encoder: loss of information happens as we progress from one timestamp to another. I.e. not all information from prev. hidden layer (I.e. encoding of prev word) is getting propagated to the next hidden layer



- Core Idea of attention: At each stage in decoder, use a encoder signal from corresponding timestamp to focus more on a particular part of the sentence. See the below image --

- Explanation of the below image: This image shows attention score calculation for first timestamp of the encoder.
  - Assume that "START" is first timestamp in decoder. It has received the encoder feature representation (output of last block of encoder)
  - To generate the attention scores – we take dot product between each of the encoder's hidden stages and decoder's first hidden stage
  - Using softmax the attention scores are converted into probability distribution -- I.e. Attention distribution
  - Attention Output: is then a weighted average of all encoder hidden stages wrt. The attention probability distribution.
  - This attention output is then combined with the decoder's hidden stage