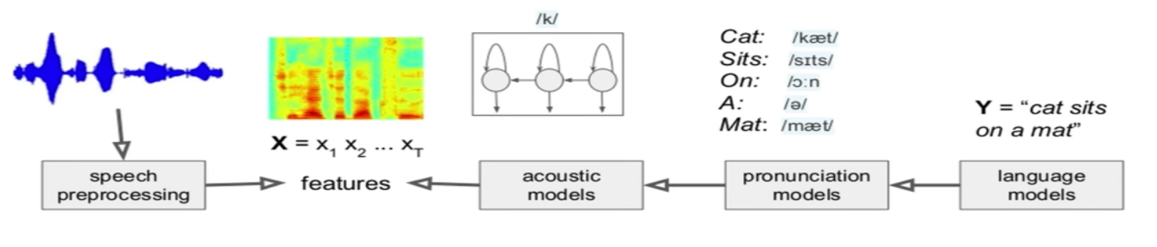
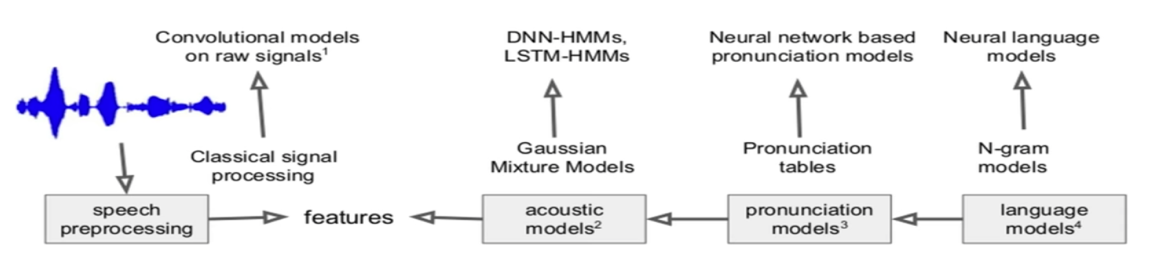
Lecture 12 | End-to-end models for speech processing

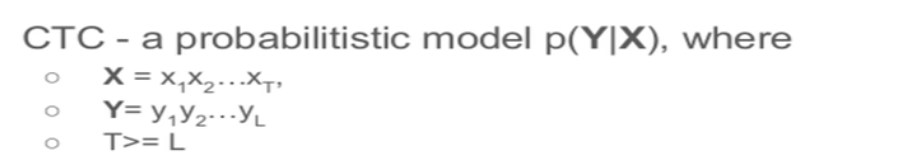
* Speech recognition
  + The classic way of building a speech recognition system is to build a **generative** model
  + Building a statistical model of speech starting from text sequences to audio features



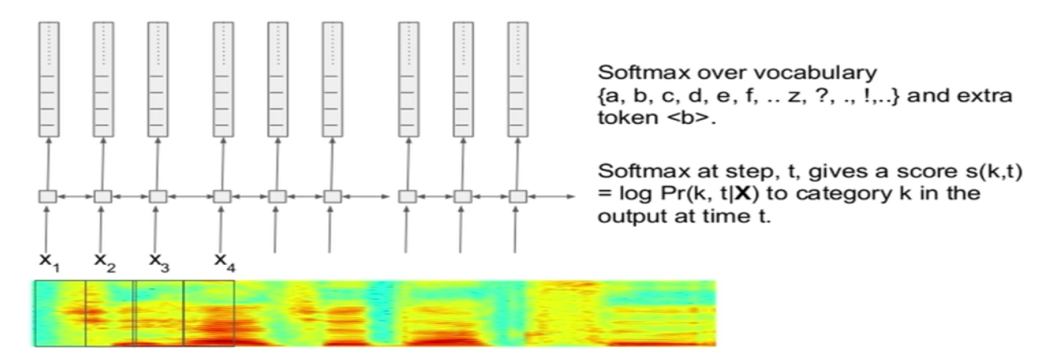
* + - Different statistical models used in different componets
      * N-gram models
      * Pronunciation tables
      * Gaussian Mixture Models
      * Classical signal processing
* Neural network speech recognition
  + Each of the components seems to be better off with a neural network



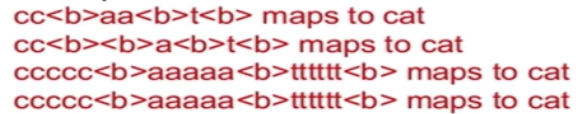
* + However, each component is trained independently, with a different objective. Therefore, errors in one component may not behave well with errors in another component
  + We should train models that encompass all of these components together (end-to-end models)
    - Connectionist Temporal Classification (CTC)
    - Sequence to sequence (Listen Attend and Spell)
* End-to-end speech recognition
  + Treat it as a modelling task
  + Audio X input (audio/processed spectrogram) and corresponding output text Y (text sequence)
  + Perform speech recognition by learning a probabilistic model p(Y|X)
* Connectionist Temporal Classification (CTC)



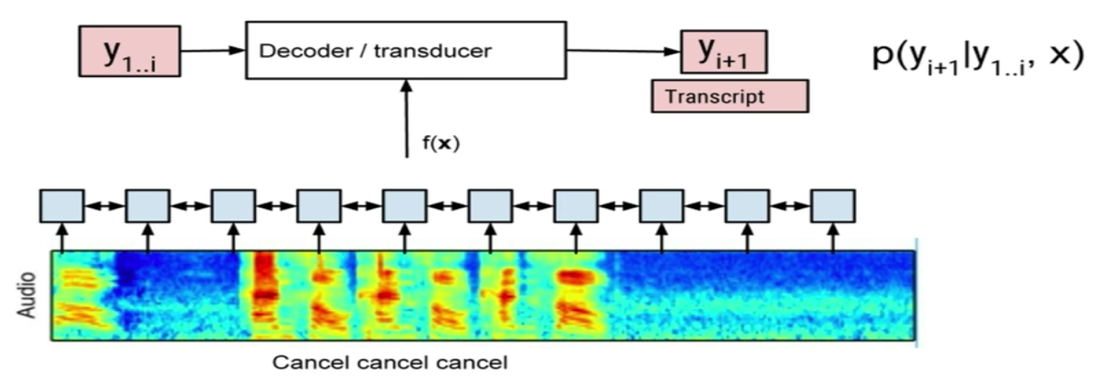
* + It has a specific structure that is suited for speech

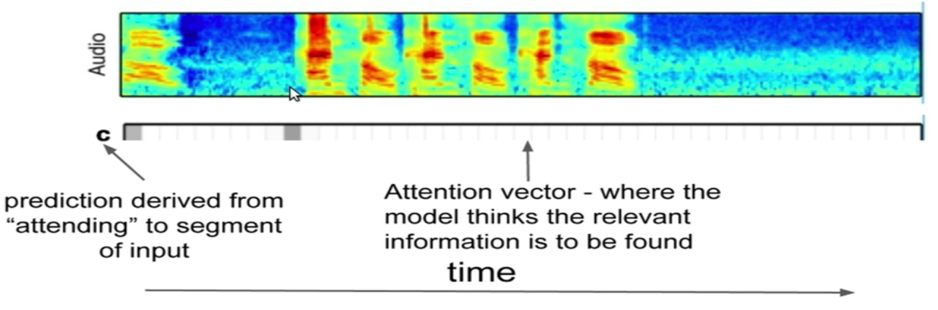


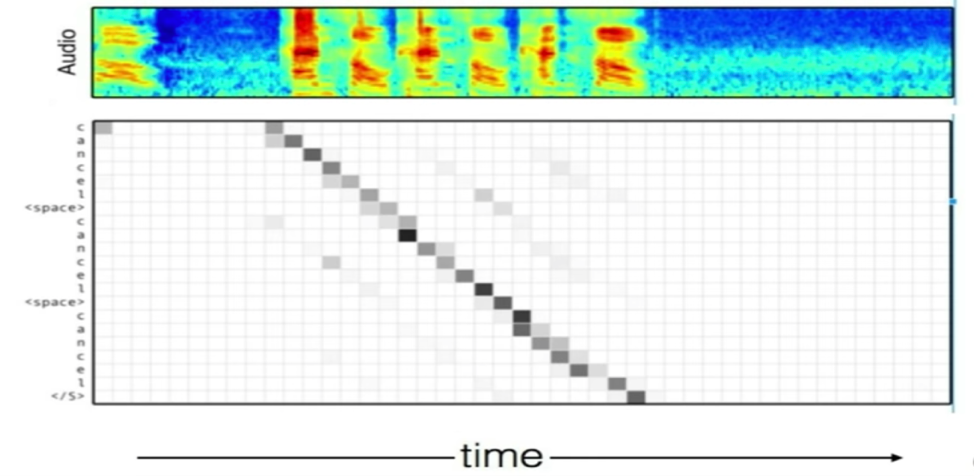
* + Repeated tokens are deduplicated
  + Any original transcript, maps to all possible paths in the duplicated space:



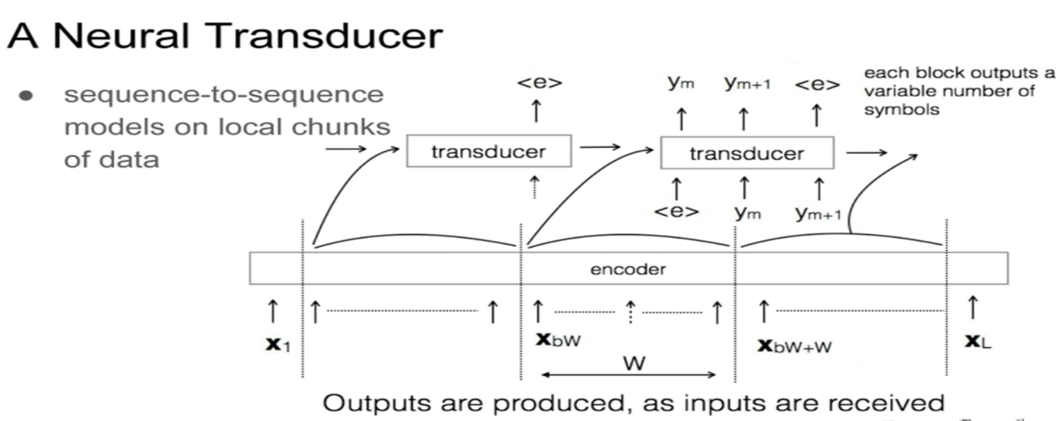
* + The score (log probability) of any path is the sum of the scores of individual categories at the different time steps
  + The probability of any transcript is the sum of probabilities of all paths that correspond to that transcript
  + Due to dynamic programming, we can compute both the log probability p(Y|X) and its gradient, which can be used for backpropagation
  + Transcript can sound correct but it lacks the correct spelling and grammar
    - More training data can help but eventually we will need a language model to fix this
    - With a simple language model rescoring, word error rate (WER) goes from 30.1% to 8.7%
  + Google’s CTC implementation fixed these problems by integrating a language model into CTC during training
* Sequence-to-Sequence with attention speech recognition

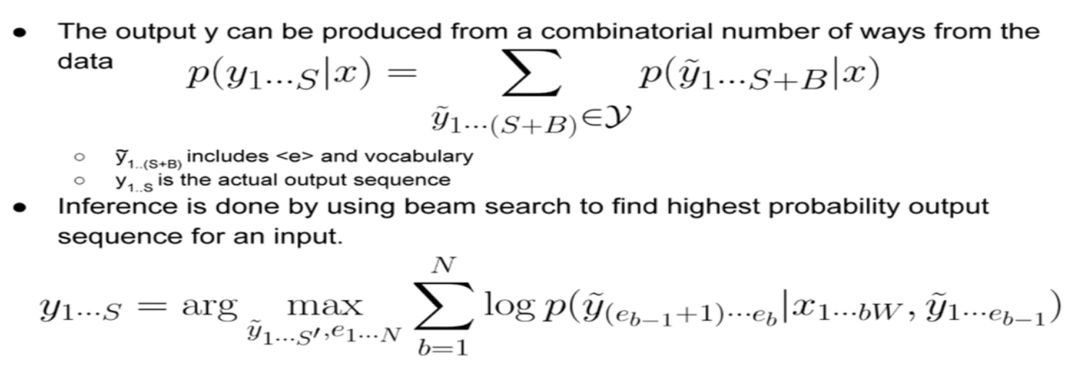


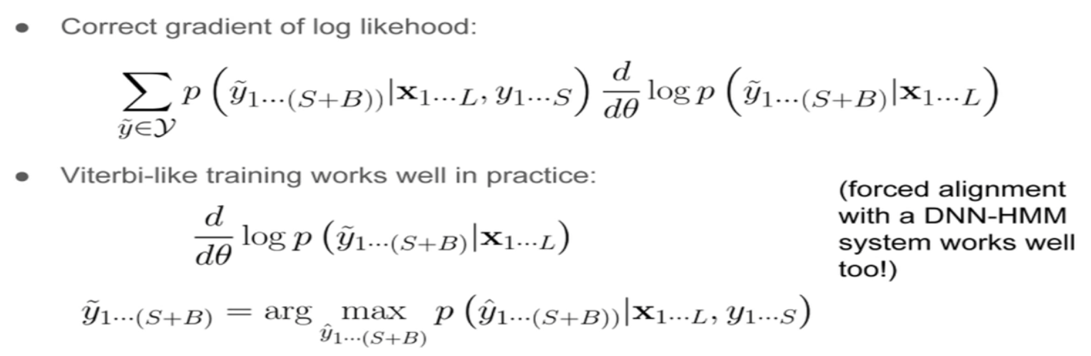




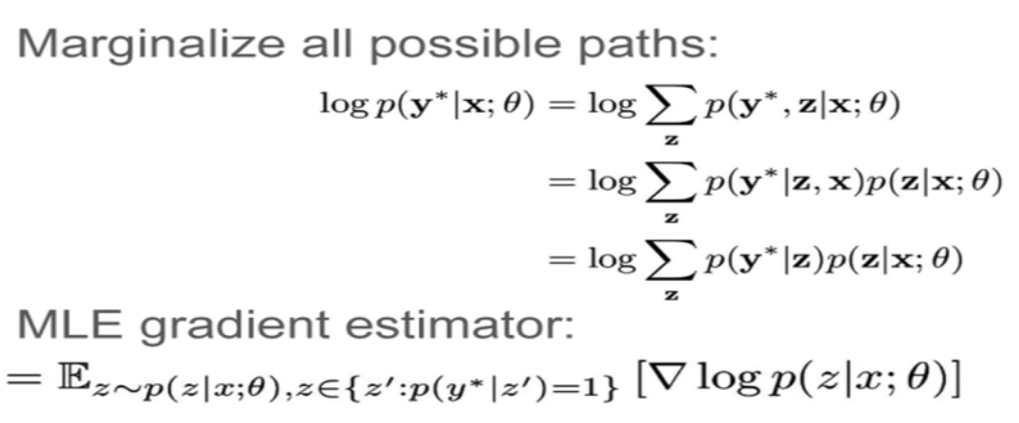
* + Listen Attend and Spell (LAS) – do more research!
* Online sequence-to-sequence models
  + Overcome limitations of the sequence-to-sequence models
    - Therefore, no need to wait for the entire input sentence to arrive
    - Attention over the entire sequence is an overkill
  + Produce outputs as inputs arrive







* + Finding best path is tricky, beam search fails easily. Approximate dynamic programming is use to find the best alignment
* Choosing the correct output targets
  + For speech, you would want to use N-grams of characters
  + So, should we decide the n-grams beforehand?
  + We can have multiple decompositions per sequence based on number of n-grams to choose
  + Its not clear which decomposition is the best way, therefore we built a model to learn this automatically:
    - Latent Sequence Decompositions



* **Entropy regularisation** prevents your softmax from ever being overconfidence at word boundaries