CS4347

Sound and Music Computing

L5: Automatic Speech Recognition (ASR)

Wang Ye

www.comp.nus.edu.sg/~wangye

wangye@comp.nus.edu.sg

Office: AS6-04-08

Topics to Cover (*selective approach*)

Part A: The Core

- Introduction
- Review of DFT, Audio Representation, and Machine Learning
- Music Representation, Analysis and Transcription
- Automatic Music Transcription (AMT)
- Automatic Speech Recognition (ASR)
- Generative Models for Text-to-Speech (TTS) & Singing Voice Synthesis (SVS)

Midterm break

Part B: The Breadth

- Spoken language assessment
- Singing voice processing
- Music production audio effects
- Automatic Music Generation
- Synthesis of sound & music – a DSP approach
- Project presentations/demo

Topics Today

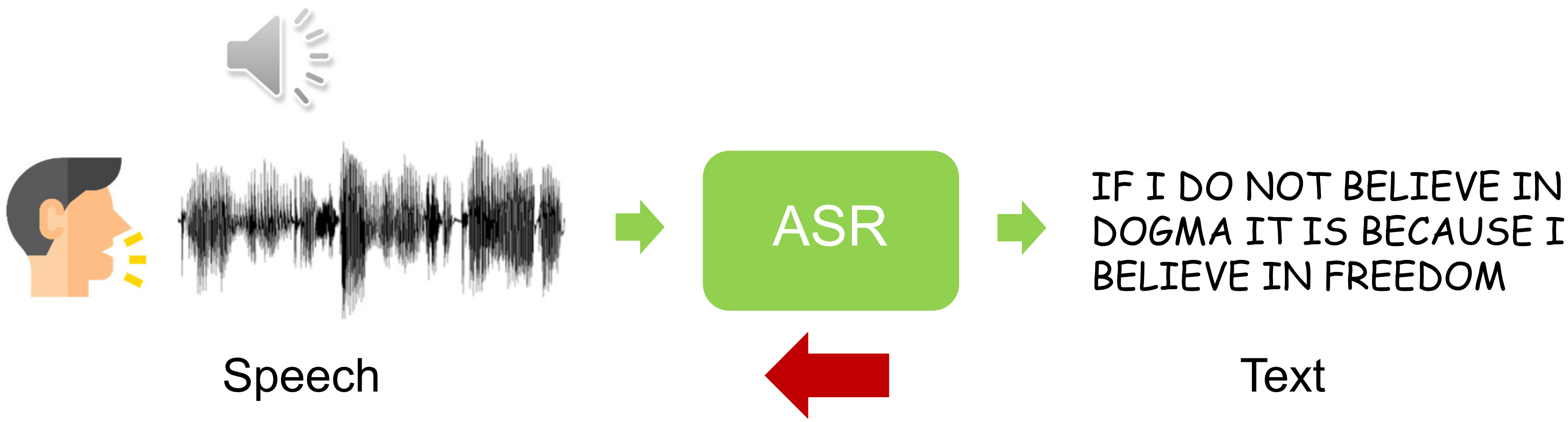
Part A: Overview of Automatic Speech Recognition (ASR)

Part B: Development of ASR System

Part C: End-to-End ASR System

Part D: Automatic Lyric Transcription (ALT)

What is Automatic Speech Recognition (ASR)?



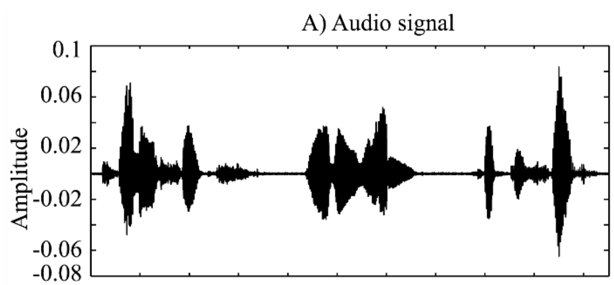
- ASR task aims to transcribe a speech waveform into a text transcript
- Also known as speech-to-text (STT); the reverse is known as TTS
- Variants – keyword spotting (KWS), voice command recognition
- Speaker identification/verification
- Variants – e.g, speaker diarization
- Large vocabulary continuous speech recognition (LVCSR)

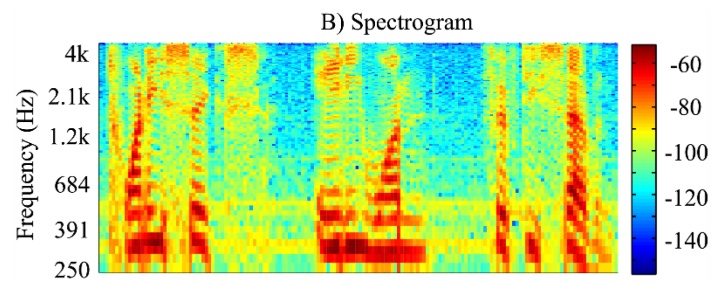
Draw an analogy between speech and music

Speech is usually represented as a sequence of acoustic features

Acoustic Features

$$X = \{\boldsymbol{x}_t \in R^D | t = 1,2, \ldots, T\}$$





Raw waveform:

- 1-D Temporal domain representation
- Sampling rate, channels
- High-dimensional

Spectral features:

- Time-frequency domain representation
- MFCC, STFT, filter bank
- Low-dimensional

Text transcript is represented as a sequence of tokens

Text transcript

$W = \{w_n \in V | n = 1,2,..,N\}$

IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I BELIEVE IN FREEDOM

Vocabulary (types of tokens):

- Word
- Character
- Phoneme
- Sub-word

Text transcript is represented as a sequence of tokens

Text transcript

$$W = \{w_n \in V \mid n = 1,2,..,N\}$$

IF I DO NOT BELIEVE IN $\underbrace{\text{DOGMA}}_{w_i}$ IT IS $\underbrace{\text{BECAUSE}}_{w_j}$ I BELIEVE IN FREEDOM

Vocabulary:

- Word
- Character
- Phoneme
- Sub-word

Text transcript is represented as a sequence of tokens

Text transcript

$$W = \{w_n \in V \mid n = 1,2,..,N\}$$

IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I BELIEVE IN FREEDOM

w_i w_j

Vocabulary:

- Word
- Character
- Phoneme
- Sub-word

Text transcript is represented as a sequence of tokens

Text transcript

$W = \{w_n \in V | n = 1,2,..,N\}$

IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I BELIEVE IN FREEDOM

$\underbrace{\text{/n/}}_{w_i}$ /ao/ /t/

$\underbrace{\text{/iy/}}_{w_j}$ /z/

Vocabulary:

- Word
- Character
- Phoneme
- Sub-word

Text transcript is represented as a sequence of tokens

Text transcript
$$W = \{w_n \in V \mid n = 1,2,..,N\}$$

IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I BELIEVE IN FREEDOM

$\underbrace{\text{'\_BE'}}_{w_i}$, 'CAUSE' '_BE', $\underbrace{\text{'LIEVE'}}_{w_j}$

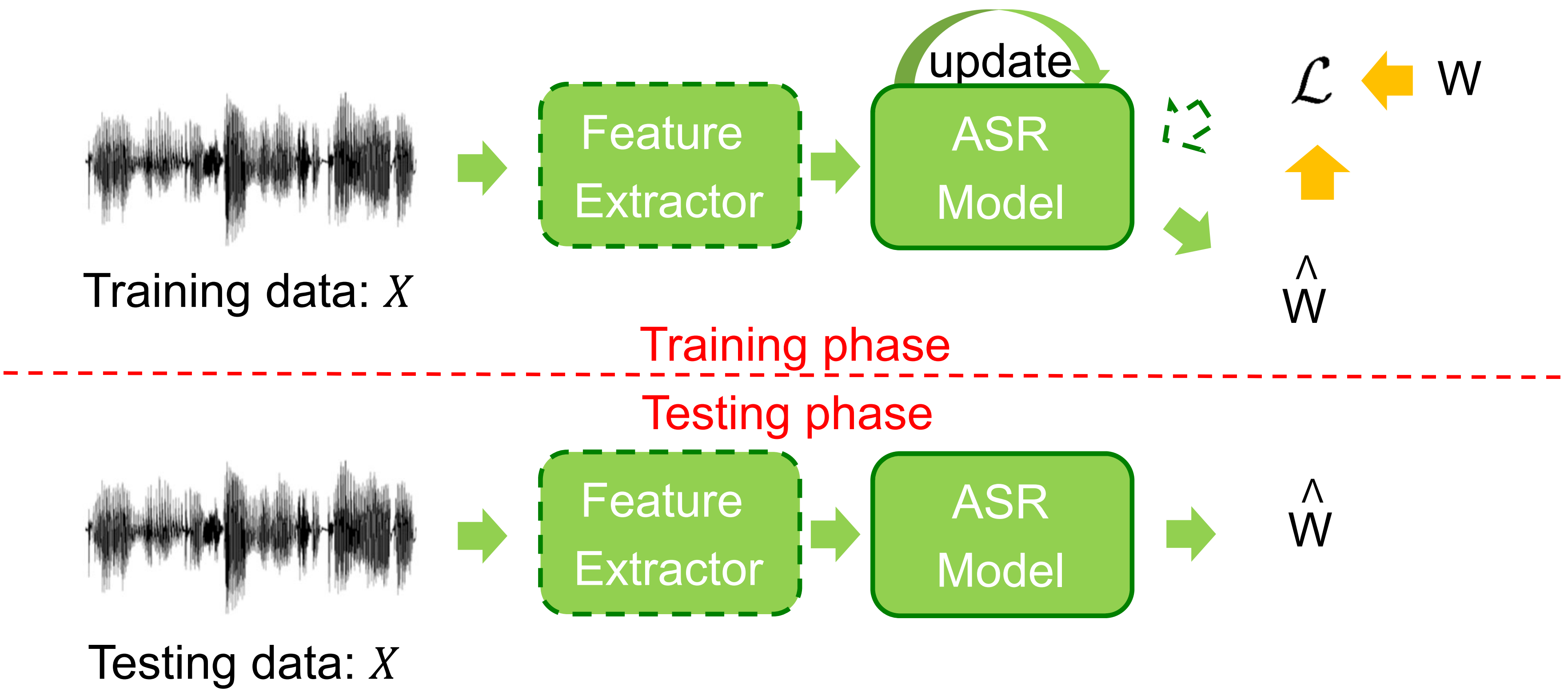
Vocabulary:

- Word
- Character
- Phoneme
- Sub-word

How to build an ASR system?

Modeling ASR task as a machine learning problem:

- Supervised learning dominates the paradigm



- Semi-supervised and unsupervised learning are being investigated for ASR

How to evaluate an ASR system?

Word Error Rate (WER): most widely used metric

Ref.	IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I BELIEVE IN FREEDOM
Out.	IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I BELIEVE IN KINGDOM

S

Substitutions

How to evaluate an ASR system?

Word Error Rate (WER): most widely used metric

Ref.	IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I BELIEVE IN FREEDOM
Out.	IF I DO NOT BELIEVE IN DOGMA IT BECAUSE I BELIEVE IN FREEDOM

D

Deletions

How to evaluate an ASR system?

Word Error Rate (WER): most widely used metric

Ref.	IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I BELIEVE IN FREEDOM
Out.	IF I DO NOT BELIEVE IN DOGMA IT IS BECAUSE I AM BELIEVE IN FREEDOM

I

Insertions

How to evaluate an ASR system?

Word Error Rate (WER): most widely used metric

WER ↓ $\leftrightarrow$ Performance ↑

$$\boxed{WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C}}$$

- S is the number of word substitutions
- D is the number of word deletions
- I is the number of word insertions
- C is the number of correct words
- N is the total number of words in the reference

Similarly, we can define:

Character Error Rate (CER)

Phoneme Error Rate (PER), etc.

Topics Today

Part A: Overview of Automatic Speech Recognition (ASR)

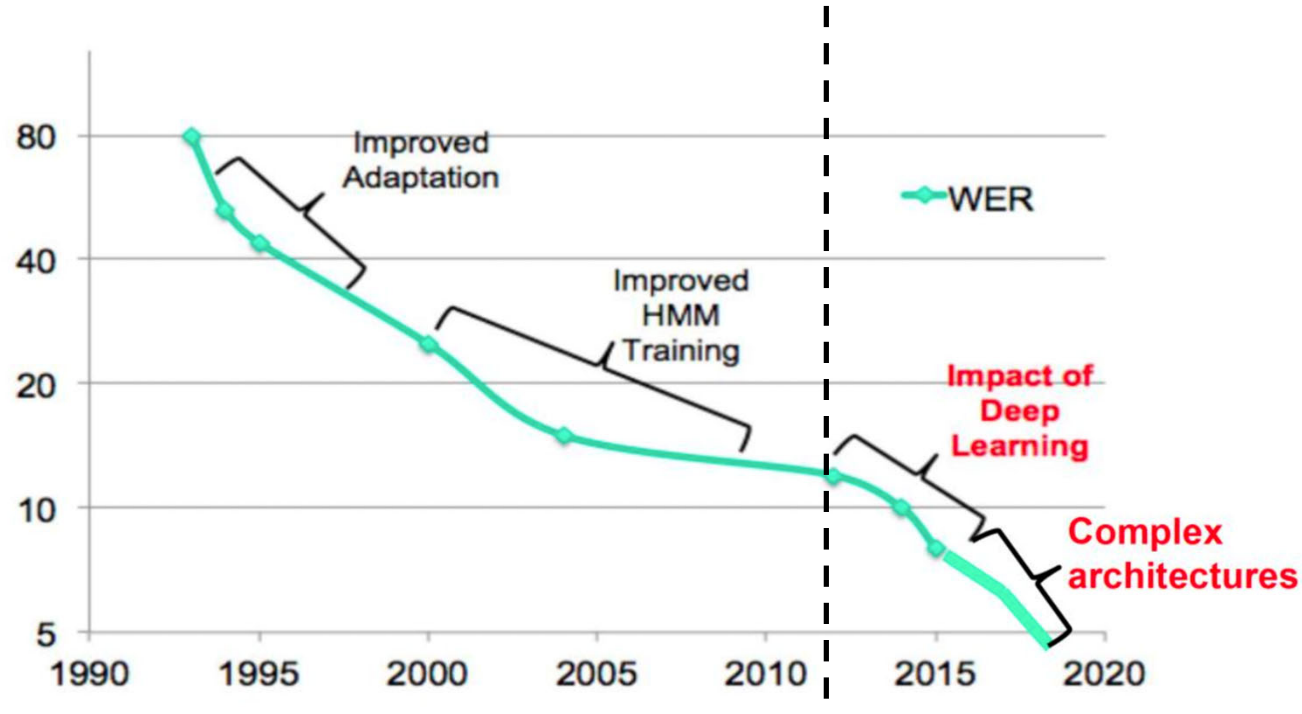
Part B: Development of ASR System

Part C: End-to-End ASR System

Part D: Automatic Lyric Transcription (ALT)

Development of ASR System over the years

ASR Performance on Switchboard dataset (Benchmark English Corpus)



HMM/GMM framework dominated ASR for half a century

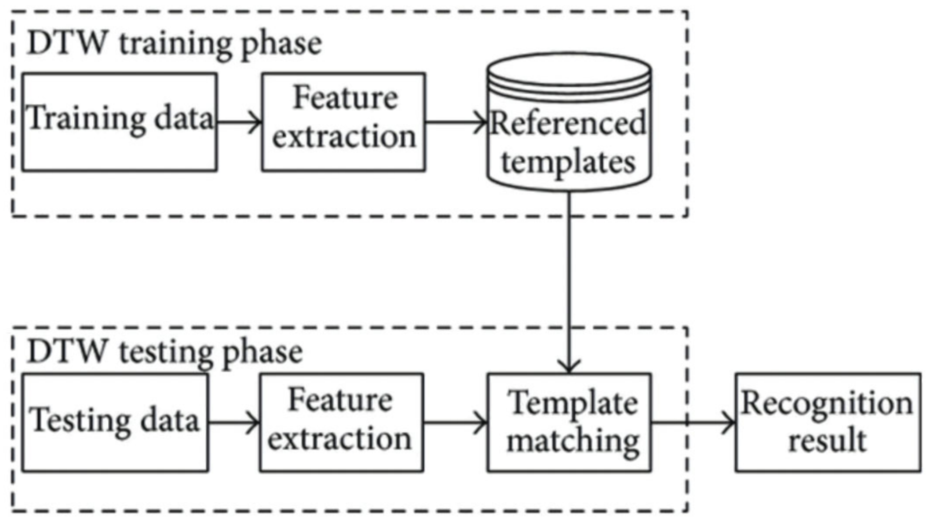
DL based methods have emerged in the past decade

Image from Prasanta Kumar Ghosh’s presentation at ICASSP 2022

Template-based ASR

Training phase:

- ➢ Build a database for referenced templates
- ➢ Each template is the representation of an actual speech segment together with its transcription, neighboring templates, meta-information, etc.



Testing phase:

- ➢ Match the speech segments of test data with referenced templates through DTW (Dynamic Time Warping) algorithm

e.g., digit recognition

Godin C, Lockwood P. DTW schemes for continuous speech recognition: a unified view. Computer Speech & Language, 1989, 3(2): 169-198.

Statistical ASR

Statistical Approach:

- During the testing (or called decoding): Aims to find the most likely sequence of tokens (W) given the sequence of acoustic features (X)

$$\boxed{\hat{W} = \underset{W \in V}{\operatorname{argmax}} P(W|X)}$$

Text transcript

$$W = \{w_n \in V | n = 1,2,..,N\}$$

Acoustic Features

$$X = \{\boldsymbol{x}_t \in R^D | t = 1,2,\dots,T\}$$

Statistical ASR

Statistical Approach:

- During the testing (or called decoding): Aims to find the most likely sequence of tokens (W) given the sequence of acoustic features (X)
- During the training: Aims to train the model Φ to maximize the probability given the pairs of W and X

$$\boxed{\Phi^* = \underset{\Phi}{\operatorname{argmax}} P_{\Phi}(W|X)}$$

Text transcript

$$W = \{w_n \in V | n = 1,2,..,N\}$$

Acoustic Features

$$X = \{\boldsymbol{x}_t \in R^D | t = 1,2,\ldots,T\}$$

Statistical ASR: HMM-based ASR

Hidden Markov Model (HMM)-based ASR:

- Adopt Bayes’ Theorem to optimize the posterior
- Model the likelihood via Acoustic Model and prior via Language Model

$$\boxed{\Phi^* = \underset{\Phi}{\operatorname{argmax}}\, P_\Phi(W|X)}$$

$\Updownarrow$ Bayes’ Theorem

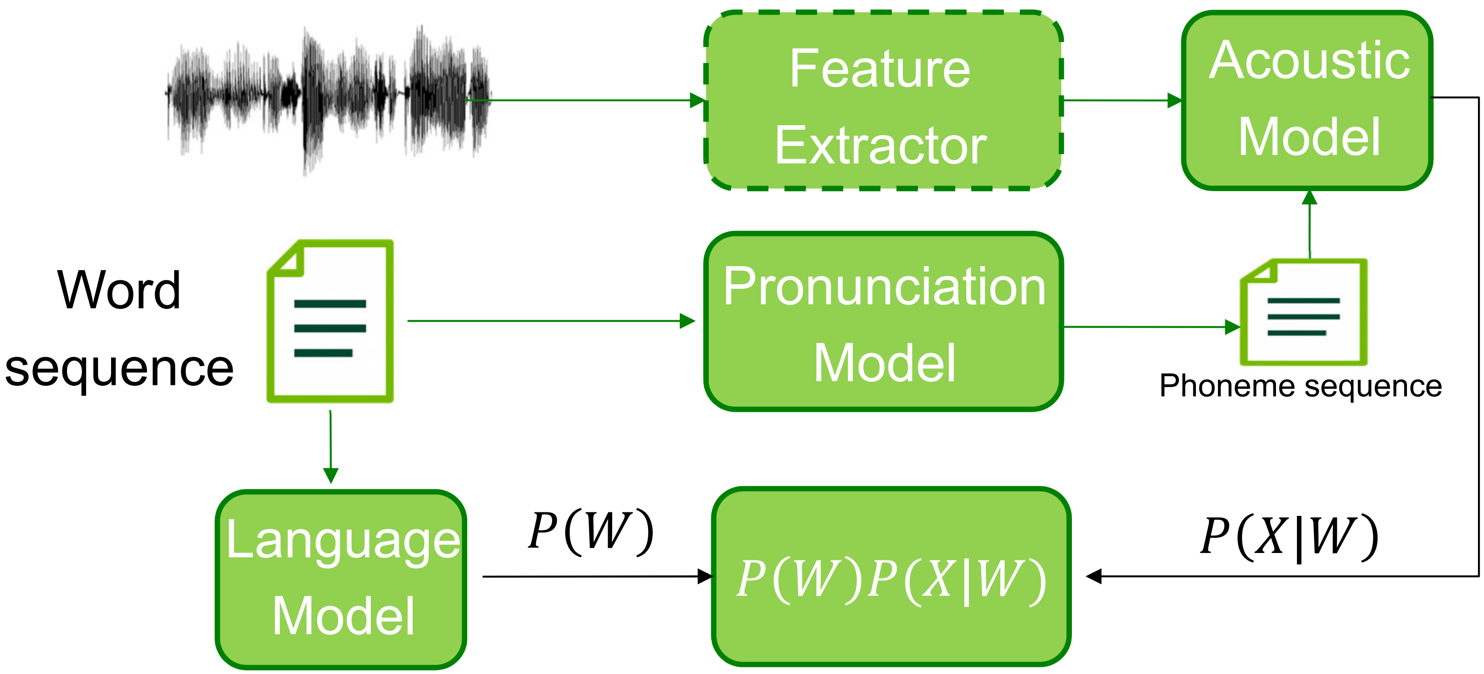
$$\boxed{\Phi^*, \Theta^* = \underset{\Phi^*, \Theta^*}{\operatorname{argmax}}\, P_\Phi(X|W) P_\Theta(W)}$$

Acoustic Model   Language Model

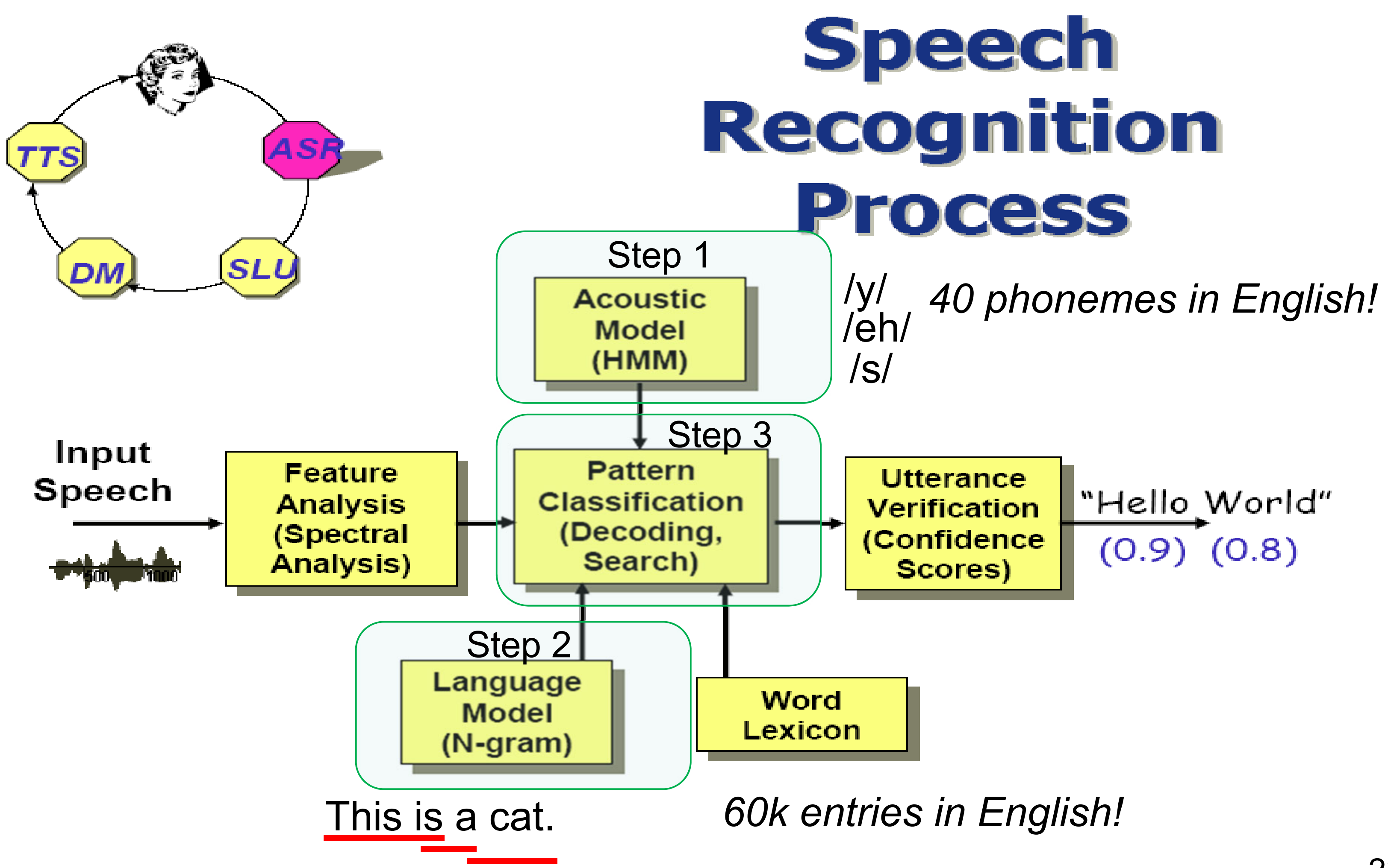
Statistical ASR: HMM-based ASR

HMM-based ASR:

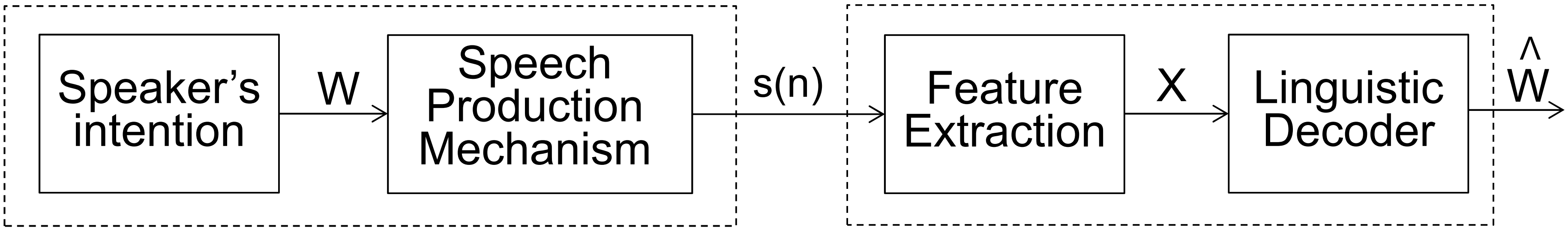
- ➢ Adopt Bayes' Theorem to optimize the posterior
- ➢ Model the likelihood via Acoustic Model and prior via Language Model
- ➢ Split word sequence into phoneme sequence via Pronunciation Model



HMM based Automatic Speech Recognition (ASR) System



Basic ASR Formulation (Bayes Method)



Speaker model (generative model) Speech recognizer

$$
\begin{aligned}
\hat{W} &= \arg\max_{W} P(W \mid X) \\
&= \arg\max_{W} \frac{P(X \mid W)\,P(W)}{P(X)} \\
&= \underbrace{\arg\max_{W}}_{\text{Step 3}} \; \underbrace{P_A(X \mid W)}_{\text{Step 1}}\,\underbrace{P_L(W)}_{\text{Step 2}}
\end{aligned}
$$

Posterior probability (X produced by W)

Probability that a given model (W) produces X (acoustic model) – it is also called likelihood

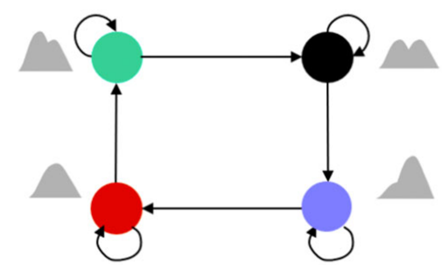
Prior probability of the model (language model)

Prior probability of the observation

Statistical ASR: HMM-based ASR

HMM-based ASR:

- ➢ Acoustic Model is HMM-based
- ➢ State can be the phoneme s_t



State transition
$p(s_t|s_{t-1})$

State sequence (hidden) s_t

Emission distribution
$p(\boldsymbol{x}_t|s_t)$

Observation sequence
(feature vectors) $\boldsymbol{x}_t$

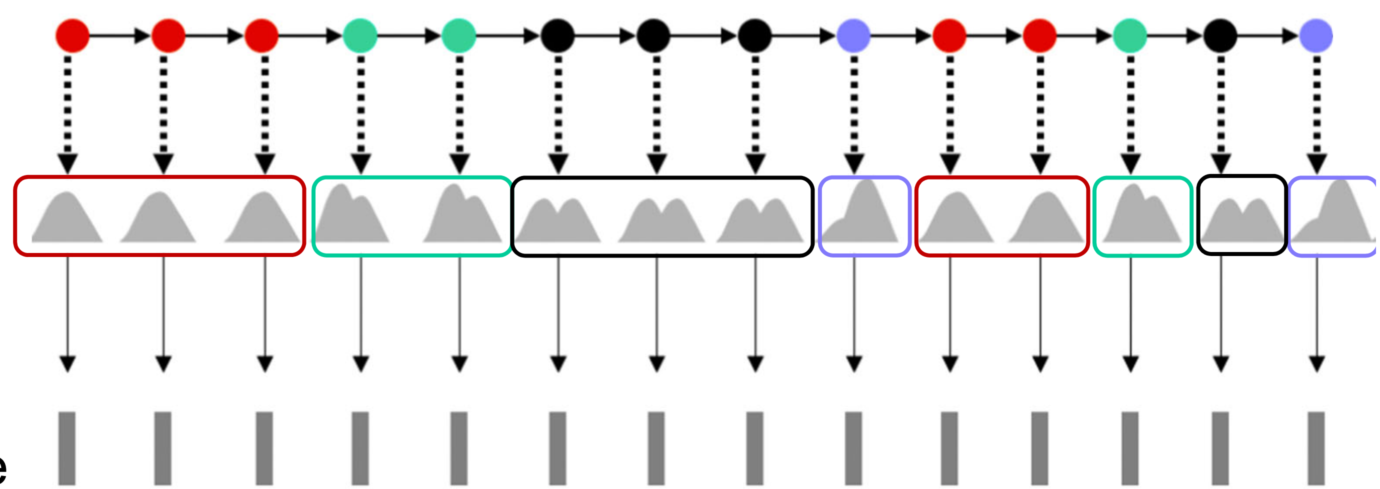


Image from Prasanta Kumar Ghosh's presentation at ICASSP 2022

Statistical ASR: HMM-based ASR

GMM-HMM ASR

- Model the emission probability density via Gaussian Mixture Model (GMM)
- GMM is a linear combination of Gaussian distribution

$$p(\boldsymbol{x}) = \sum_{m=1}^{M} P(m)p(\boldsymbol{x}|m) = \sum_{m=1}^{M} P(m)N(\boldsymbol{x}; \boldsymbol{\mu}_m, \sigma_m^2 \boldsymbol{I})$$

DNN-HMM ASR

- Model the emission probability density via Deep Neural Network (DNN)

Statistical ASR: End-to-End ASR

End-to-End ASR:

- Directly optimize the following objective
- Achieve State-of-the-art Performance on benchmark ASR datasets

$$\boxed{\Phi^* = \underset{\Phi}{\operatorname{argmax}}\, P_\Phi(W|X)}$$

Word sequence
$W = \{w_n \in V | n = 1,2,..,N\}$

Acoustic features
$X = \{\boldsymbol{x}_t \in R^D | t = 1,2,\ldots,T\}$

Focus of this lecture and assignment 3!

Topics Today

Part A: Overview of Automatic Speech Recognition (ASR)

Part B: Development of ASR System

Part C: End-to-End ASR System

Part D: Automatic Lyric Transcription (ALT)

Overview of End-to-End ASR System

Modeling ASR task as a sequence-to-sequence classification problem

Acoustic Features

$X = \{\boldsymbol{x}_t \in R^D | t = 1,2, \dots, T\}$

Pairs

Word sequence

$W = \{w_n \in V | n = 1,2,..,N\}$

Classify

$\boldsymbol{x}_t \longrightarrow w_n$

Deep Learning Model

Compare with the generative model: HMM/GMM

Problems:

- X and W can vary in length, normally $T \gg N$
- The ratio of the lengths of X and W can very, $\frac{T}{N}$ is different for each pair
- ***Accurate alignment*** of X and W is absent

Overview of End-to-End ASR System

Problems:

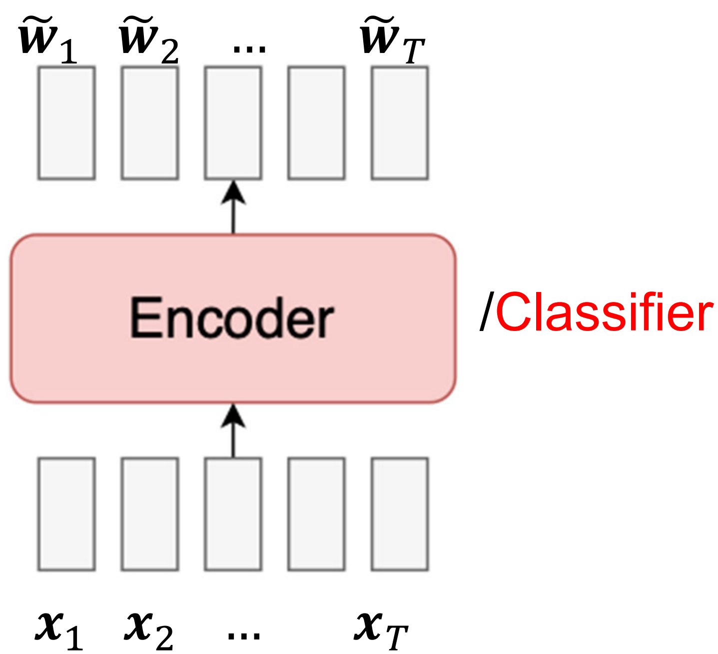
- X and W can vary in length, normally $T \gg N$
- The ratio of the lengths of X and W can very, $\frac{T}{N}$ is different for each pair
- Accurate alignment of X and W is absent

Solutions:

- ***Connectionist Temporal Classification (CTC)***
- Encoder/Decoder with Attention
- Hybrid CTC-Attention
- Transducer

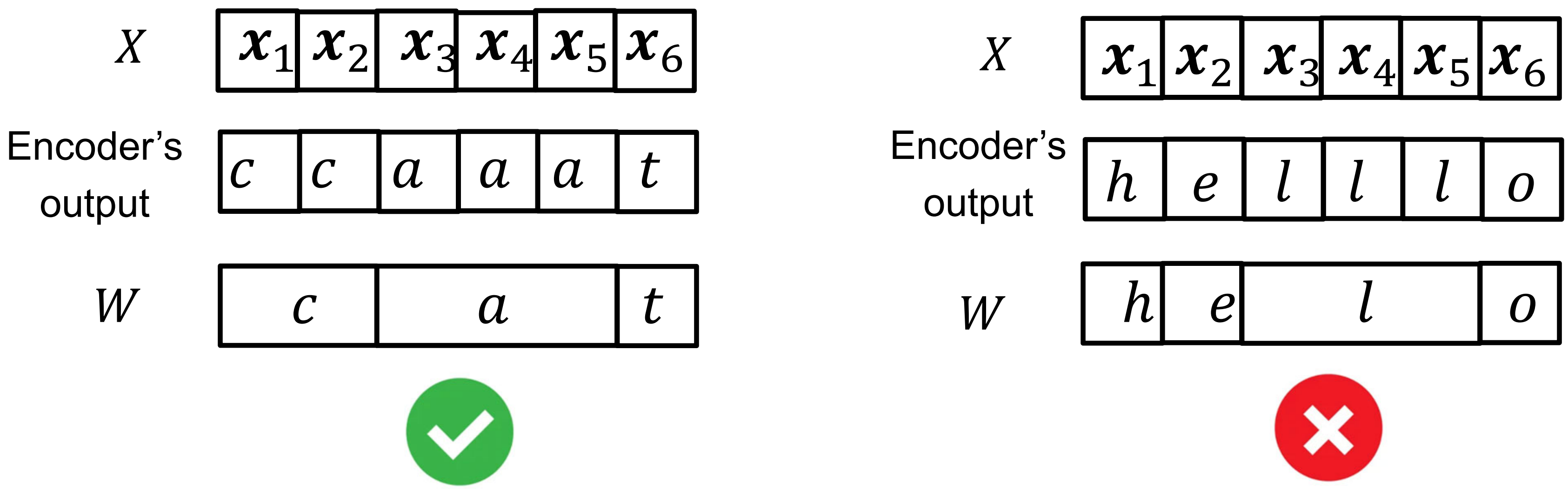
Connectionist Temporal Classification (CTC)

- CTC is introduced to label unsegmented sequences directly
- CTC adopts an encoder to predict frame-level text transcript, or called alignment



Motivation of CTC

- Given acoustic features $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_T]$ and output text transcription $W = [w_1, w_2, \dots, w_N]$, how can we align them?
- Naïve Method: remove the repeated tokens



[1] Graves A, Fernández S, Gomez F, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks[C]//Proceedings of the 23rd international conference on Machine learning. 2006: 369-376.

[2] Hannun A. Sequence modeling with ctc[J]. Distill, 2017, 2(11): e8.

How does CTC work?

- CTC introduces a ***blank token ε*** (different from space token), which is removed after merging the repeated predicted tokens

X: $\boldsymbol{x}_1$ $\boldsymbol{x}_2$ $\boldsymbol{x}_3$ $\boldsymbol{x}_4$ $\boldsymbol{x}_5$ $\boldsymbol{x}_6$ $\boldsymbol{x}_7$ $\boldsymbol{x}_8$ $\boldsymbol{x}_9$ $\boldsymbol{x}_{10}$ $\boldsymbol{x}_{11}$ $\boldsymbol{x}_{12}$ $\boldsymbol{x}_{13}$ $\boldsymbol{x}_{14}$ $\boldsymbol{x}_{15}$ $\boldsymbol{x}_{16}$ $\boldsymbol{x}_{17}$

Predicted Alignment: h e l l ε l o ε ε $-$ w o o ε r l d

Classification

Merge repeated: h e l ε l o ε $-$ w o ε r l d

B

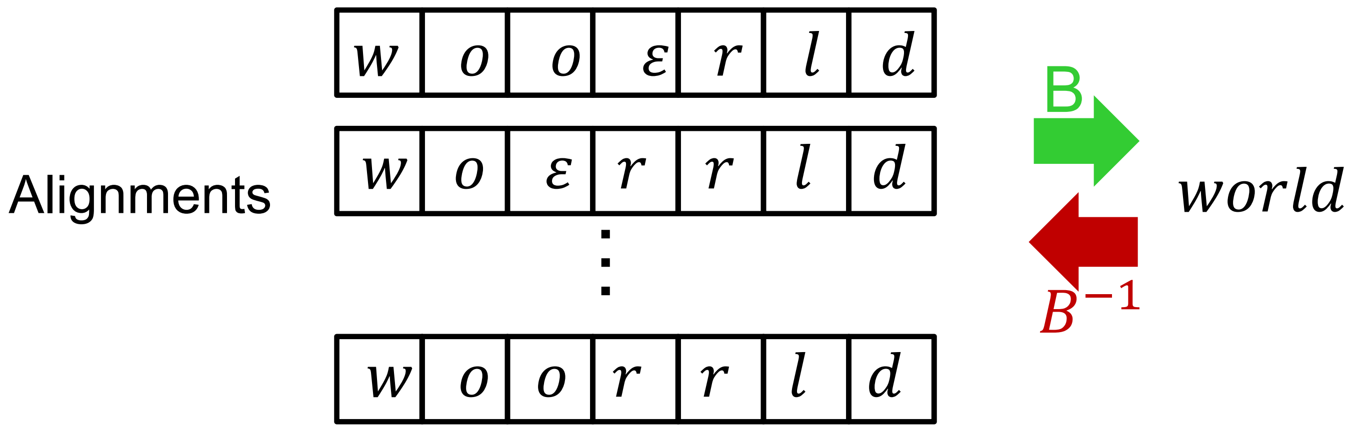
Remove blank token: h e l l o $-$ w o r l d

B^{-1}

W: h e l l o $-$ w o r l d

How to compute CTC loss?

- A correct text transcript corresponds to multiple alignments



- The probability $P(W|X)$ can be represented as

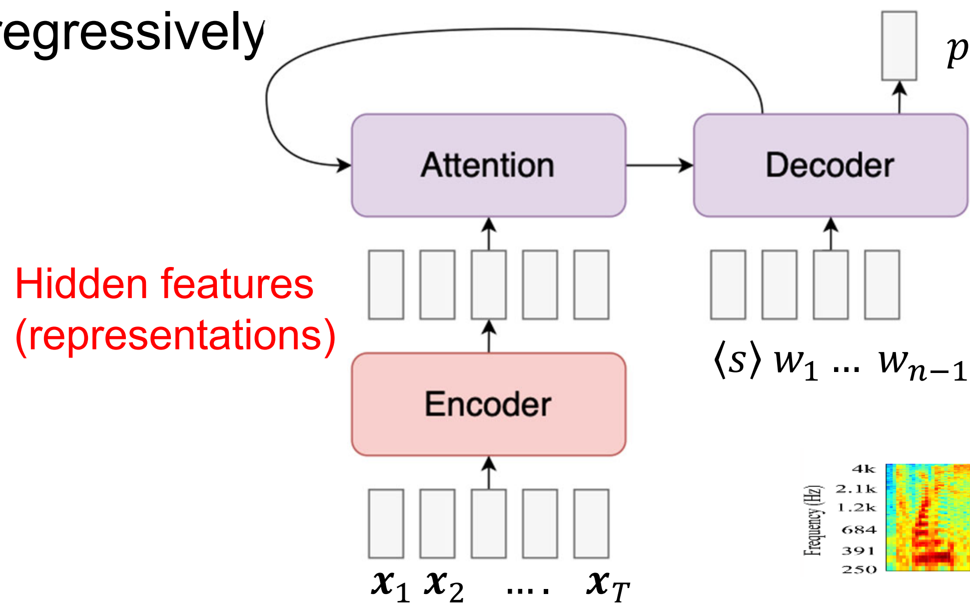
$$P(W|X) = \sum_{\pi \in B^{-1}(W)} \prod_{t=1}^{T} p(\pi_t|\boldsymbol{x}_t)$$

- Loss function can be written as:

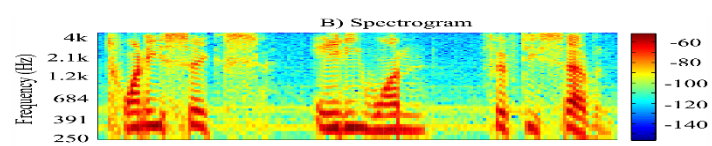
$$L_{CTC} = -\log \sum_{\pi \in B^{-1}(W)} \prod_{t=1}^{T} p(\pi_t|\boldsymbol{x}_t)$$

Encoder/Decoder with Attention

- Encoder: transform the acoustic features into a sequence of hidden features
- Attention: allows the decoder to pay attention to different parts of hidden features when predicting each token
- Decoder: accepts previous tokens and hidden features, decodes them autoregressively



VAD



[1] Bahdanau D, Chorowski J, Serdyuk D, et al. End-to-end attention-based large vocabulary speech recognition[C]//2016 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2016: 4945-4949.

Encoder/Decoder with Attention

- Encoder: transform the acoustic features into a sequence of states
- Decoder: autoregressively predict the output tokens
- Attention: allows the decoder to pay attention to different parts of hidden states when predicting each token
- Attention type: content-aware attention / location-aware attention / multi-head attention

[1] Bahdanau D, Chorowski J, Serdyuk D, et al. End-to-end attention-based large vocabulary speech recognition[C]//2016 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2016: 4945-4949.
[2] Chorowski J K, Bahdanau D, Serdyuk D, et al. Attention-based models for speech recognition[J]. Advances in neural information processing systems, 2015, 28.
[3] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

Encoder/Decoder with Attention

- The probability $P(W|X)$ can be represented as

$$P(W|X) = \prod_{n=1}^{N} p(w_n|w_{1:n-1}, X)$$

- Loss function can be written as:

$$L_{S2S} = -\log \prod_{n=1}^{N} p(w_n|w_{1:n-1}, X)$$

Hybrid CTC-Attention

- Multi-task learning: combining CTC and Encoder/Decoder Attention

- Training Loss function

$$
\begin{aligned}
L_{Hybrid} &= \lambda L_{CTC} + (1-\lambda) L_{S2S} \\
&= -\lambda \log \sum_{\pi \in B^{-1}(W)} \prod_{t=1}^{T} p(\pi_t | \boldsymbol{x}_t) \\
&\quad -(1-\lambda) \log \prod_{n=1}^{N} p(w_n | w_{1:n-1}, X)
\end{aligned}
$$

[1] Kim S, Hori T, Watanabe S. Joint CTC-attention based end-to-end speech recognition using multi-task learning[C]//2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2017: 4835-4839.

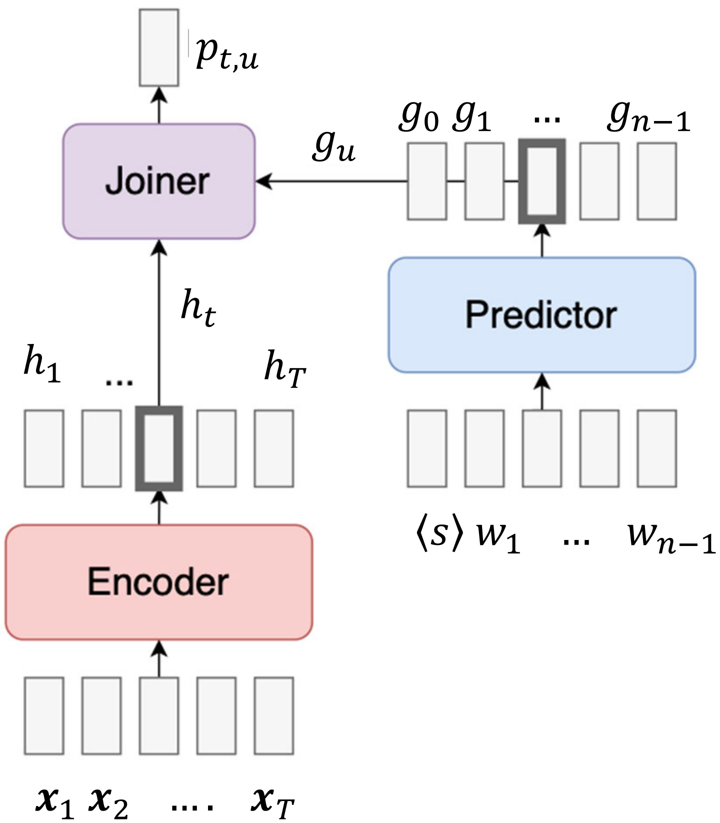
Transducer

➢ Encoder: transform the acoustic features into a sequence of hidden features

➢ Predictor: autoregressively predict the output tokens by only taking previous tokens

➢ Joiner: combine the hidden features and predictor outputs and outputs the distribution over possible tokens (including a ∅ token)

$$p_{t,u} = P(w_{t,n}|w_{1:n-1}, X)$$



[1] Lugosch, Loren. “Sequence-to-sequence learning with Transducers”, 2020. https://lorenlugosch.github.io/posts/2020/11/transducer/

How to Implement an End-to-End ASR system?

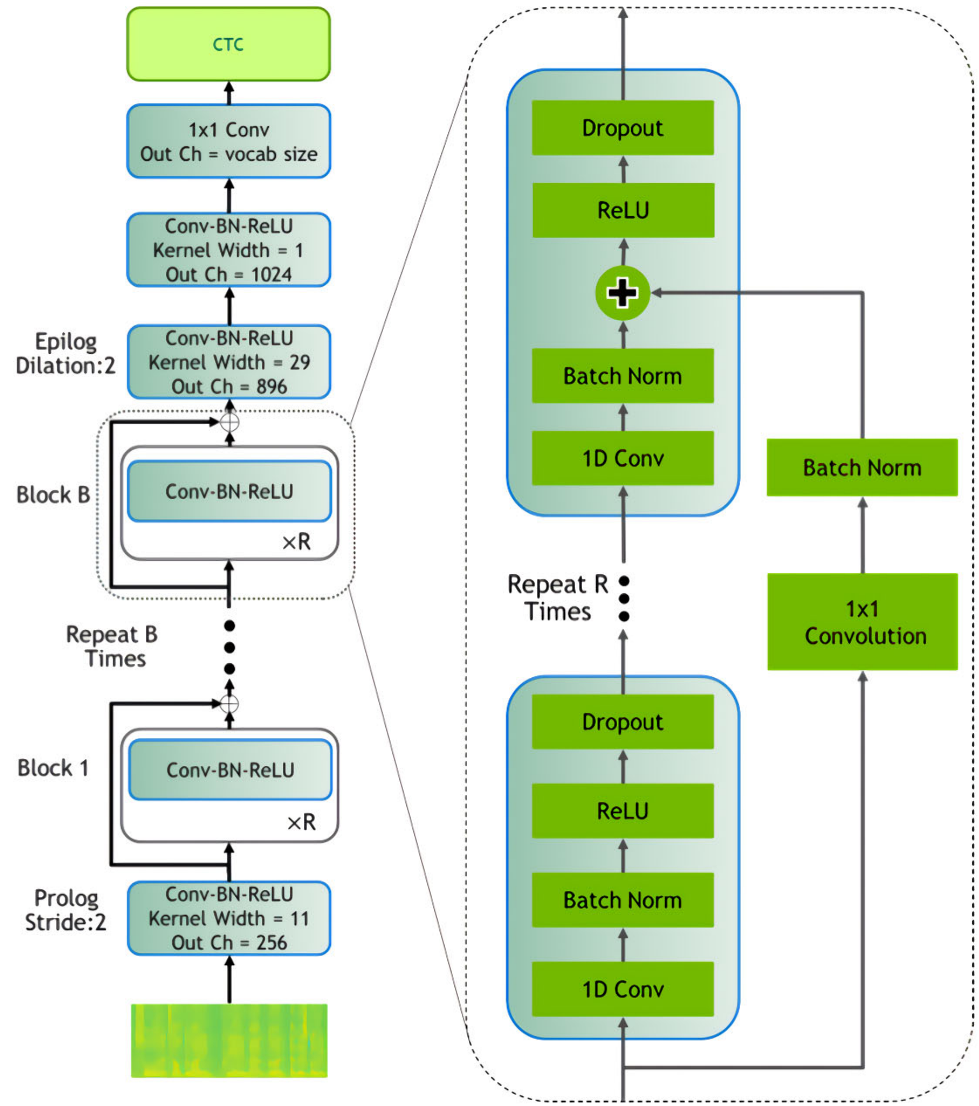
Encoder, Decoder, Predictor, Joiner are implemented by deep learning models:

- Multi-layer Perceptron (MLP)
- Convolutional Neural Networks (CNN)
- Recurrent Neural Networks (RNN)
- Transformer
- Representative variants: Jasper, Conformer, wav2vec 2.0, etc.

Encoder performs representation learning, which makes it an important part of the whole model

Example 1: Jasper

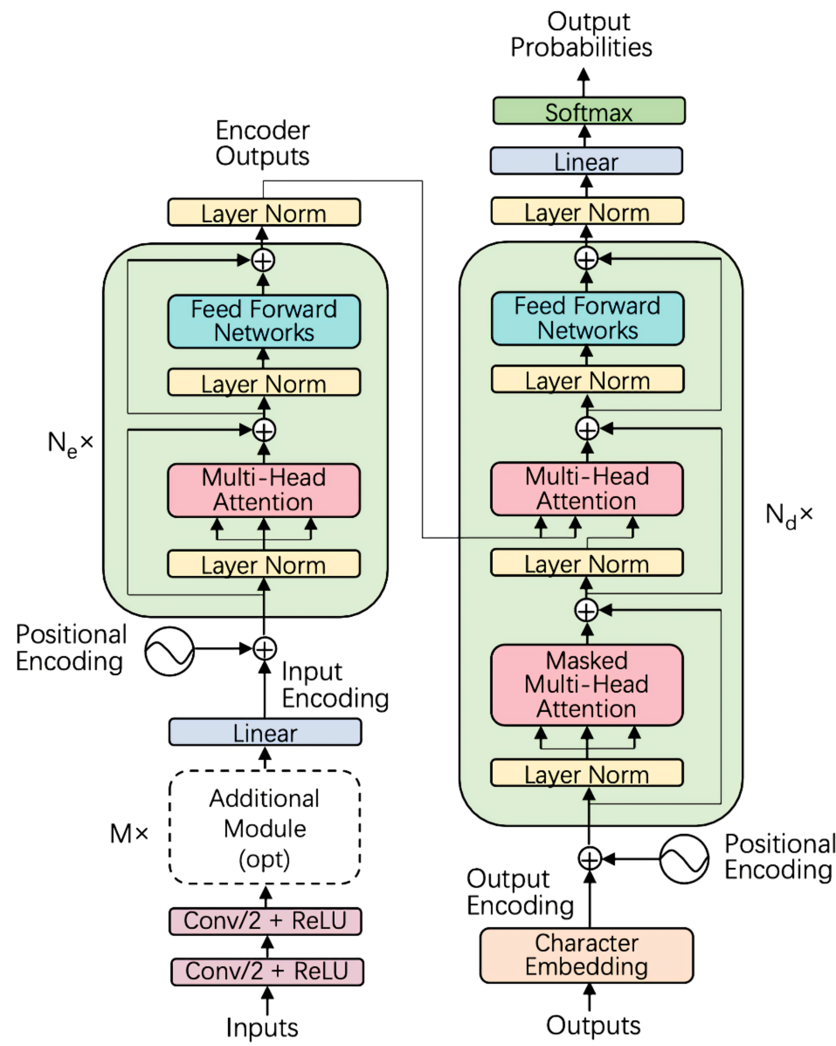
- Implemented the CTC encoder as a 1D-CNN
- Residual blocks and Dense blocks design



[1] Li J, Lavrukhin V, Ginsburg B, et al. Jasper: An end-to-end convolutional neural acoustic model[J]. arXiv preprint arXiv:1904.03288, 2019.

Example 2: Speech-Transformer

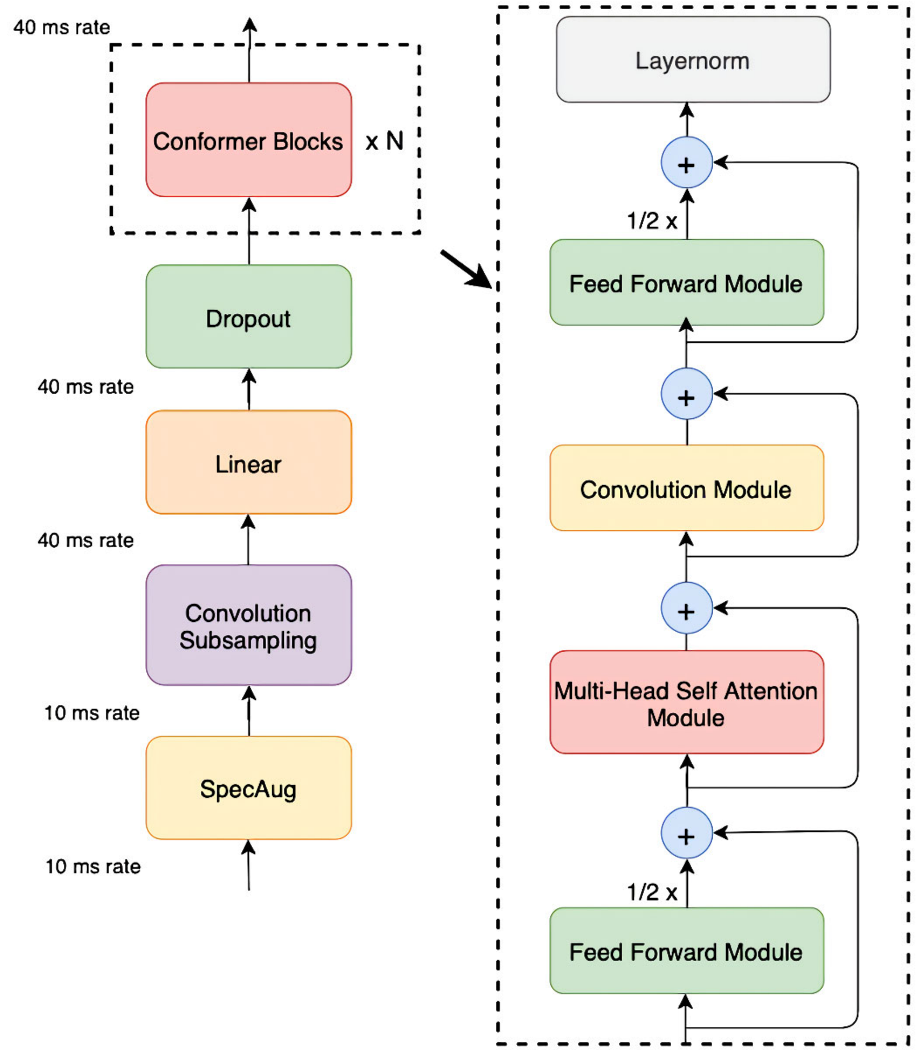
- Motivated by the success of Transformer in NLP field
- CNN are used to exploit the structure locality of inputs



[1] Dong L, Xu S, Xu B. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition[C]//2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018: 5884-5888.

Example 3: Conformer

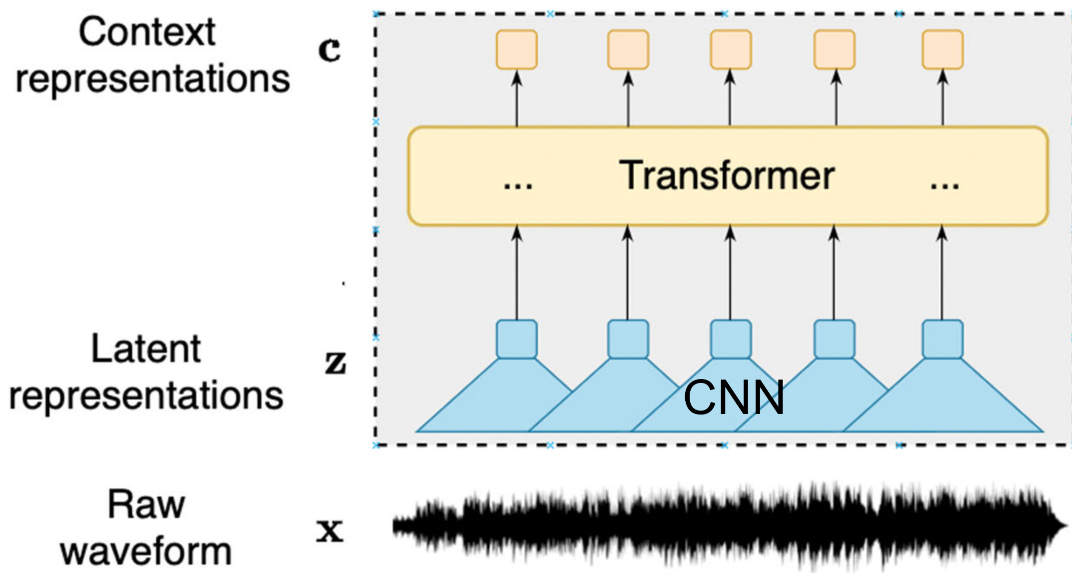
- Conformer combines both CNN and Transformer
- Fully exploit local/global dependencies among features of different frames



[1] Gulati A, Qin J, Chiu C C, et al. Conformer: Convolution-augmented transformer for speech recognition[J]. arXiv preprint arXiv:2005.08100, 2020.

Example 4: wav2vec 2.0 architecture

- wav2vec 2.0 can be adopted as the encoder to extract deep representations from raw audio
- wav2vec 2.0 includes a CNN and a Transformer



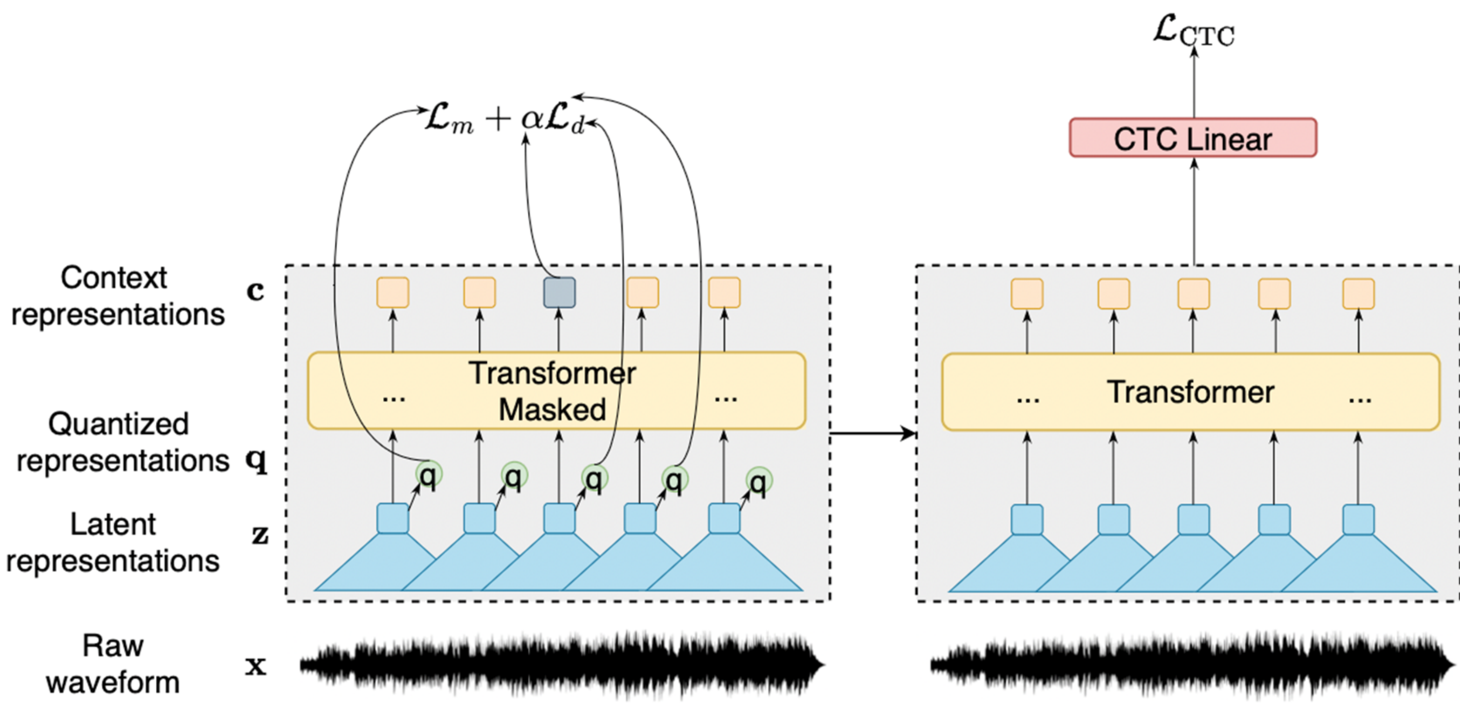
[1] Baevski A, Zhou Y, Mohamed A, et al. wav2vec 2.0: A framework for self-supervised learning of speech representations[J]. Advances in Neural Information Processing Systems, 2020, 33: 12449-12460.

Example 4: wav2vec 2.0 training

➢ Training wav2vec 2.0 has two stages:

Stage I: Self-supervised Contrastive Learning

Stage II: Supervised Fine-tuning



Language Model (LM)

- HMM-based ASR: LM participates in the testing phase

Language Model

$$\boxed{\hat{W} = \underset{W \in V}{\operatorname{argmax}} P_\Phi(X|W) P_\Theta(W)}$$

- End-to-End ASR: LM can serve as ***regularization*** to original optimization objective, empirically improve ASR performance

$$\boxed{\hat{W} = \underset{W \in V}{\operatorname{argmax}} P_\Phi(W|X)}$$

$$\Downarrow$$

Language Model

$$\boxed{\hat{W} = \underset{W \in V}{\operatorname{argmax}} P_\Phi(W|X) P_\Theta(W)^\beta}$$

What is LM?

- LM is a system which can predict the next token w_{t+1} given a sequence of previous tokens $w_1, \dots, w_t$

$$P(w_{t+1} | w_1, w_2, \dots, w_t)$$

- LM can also model the probability of a sequence of tokens $w_1, \dots, w_T$

$$P(w_1, w_2, \dots, w_T) = \prod_{t=1}^{T} \underbrace{P(w_t | w_1, \dots, w_{t-1})}_{\text{LM's output}}$$

- LM families: ***n-gram LM***, RNNLM, Transformer LM

n-gram LM definition

- An *n-gram* is a chunk of n consecutive tokens, take words as examples

  Unigrams: "he", "is", "a", "hero"

  Bigrams: "he is", "is a", "a hero"

  Trigrams: "he is a", "is a hero"

  4-grams: "he is a hero"

- *(n-1)*-order ***Markov assumption***: *n-gram* LM assumes w_{t+1} only depends on the previous *n-1* tokens $w_{t-n+2}, \dots, w_t$

$$P(w_{t+1}|w_1, w_2, \dots, w_t) = P(w_{t+1}|w_{t-n+2}, , \dots, w_t)$$

$$= \frac{P(w_{t-n+2}, , \dots, w_t, w_{t+1})}{P(w_{t-n+2}, , \dots, w_t)}$$

n-gram prob

(n-1)-gram prob

How to build an n-gram LM?

- How to compute n-gram probability and (n-1)-gram probability?

n-gram prob

$$P(w_{t+1}|w_1, w_2, \dots, w_t) = \frac{P(w_{t-n+2},, \dots, w_t, w_{t+1})}{P(w_{t-n+2},, \dots, w_t)}$$

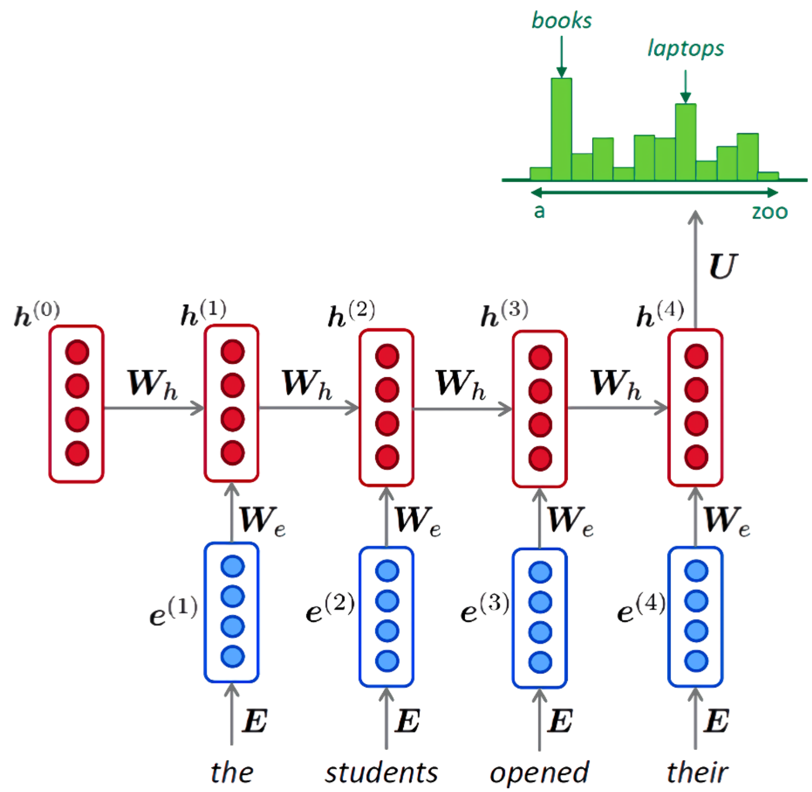
(n-1)-gram prob

- Solution: Count the frequencies of n-grams and (n-1)-grams in ***large corpus of text***

- n-gram LM is a probabilistic model without deep learning

RNN LM definition

- Directly model the $P(w_{t+1}|w_1, w_2, \ldots, w_t)$ by deep neural networks
- To enable the inputs with arbitrary length, RNN is adopted



E.g., Given four tokens, predict the next token

Image from Prasanta Kumar Ghosh's presentation at ICASSP 2022

RNNLM implementation

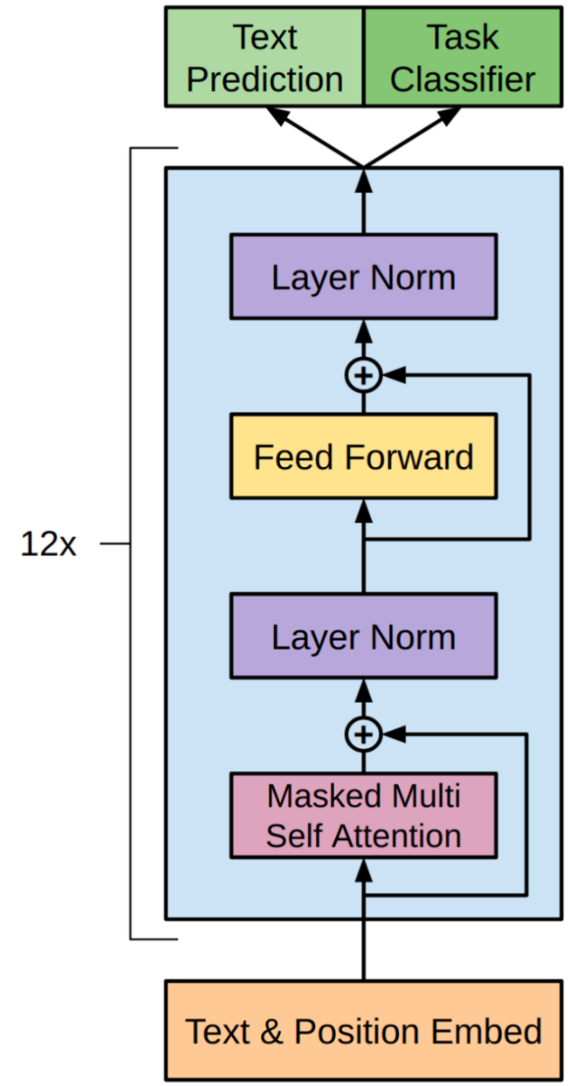
- Directly model the $P(w_{t+1}|w_1, w_2, \ldots, w_t)$ by deep neural networks
- To enable the inputs with arbitrary length, RNN is adopted
- LSTM and GRU can alleviate vanishing gradient problem comparing to vanilla RNN

- Train an RNNLM: loss function is defined as cross-entropy between predicted probability distribution of next token and true next token
- Evaluate an RNNLM: standard evaluation metric is perplexity

$$\text{perplexity} = \prod_{t=1}^{T} \left(\frac{1}{P_{LM}(w_{t+1}|w_1, w_2, \ldots, w_t)} \right)^{1/T}$$

Transformer LM

- Model $P(w_{t+1}|w_1, w_2, \ldots, w_t)$ by Transformer
- Transformer LM is capable of modeling long dependencies among tokens
- Training and evaluation of Transformer LM are like RNNLM



[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. Advances in neural information processing systems, 2017, 30.

Decoding for ASR

- HMM-based ASR:

$$\hat{W} = \underset{W \in V}{\operatorname{argmax}} P_{\Phi}(X|W) P_{\Theta}(W) = \underset{W \in V}{\operatorname{argmax}} \log P_{\Phi}(X|W) + \log P_{\Theta}(W)$$

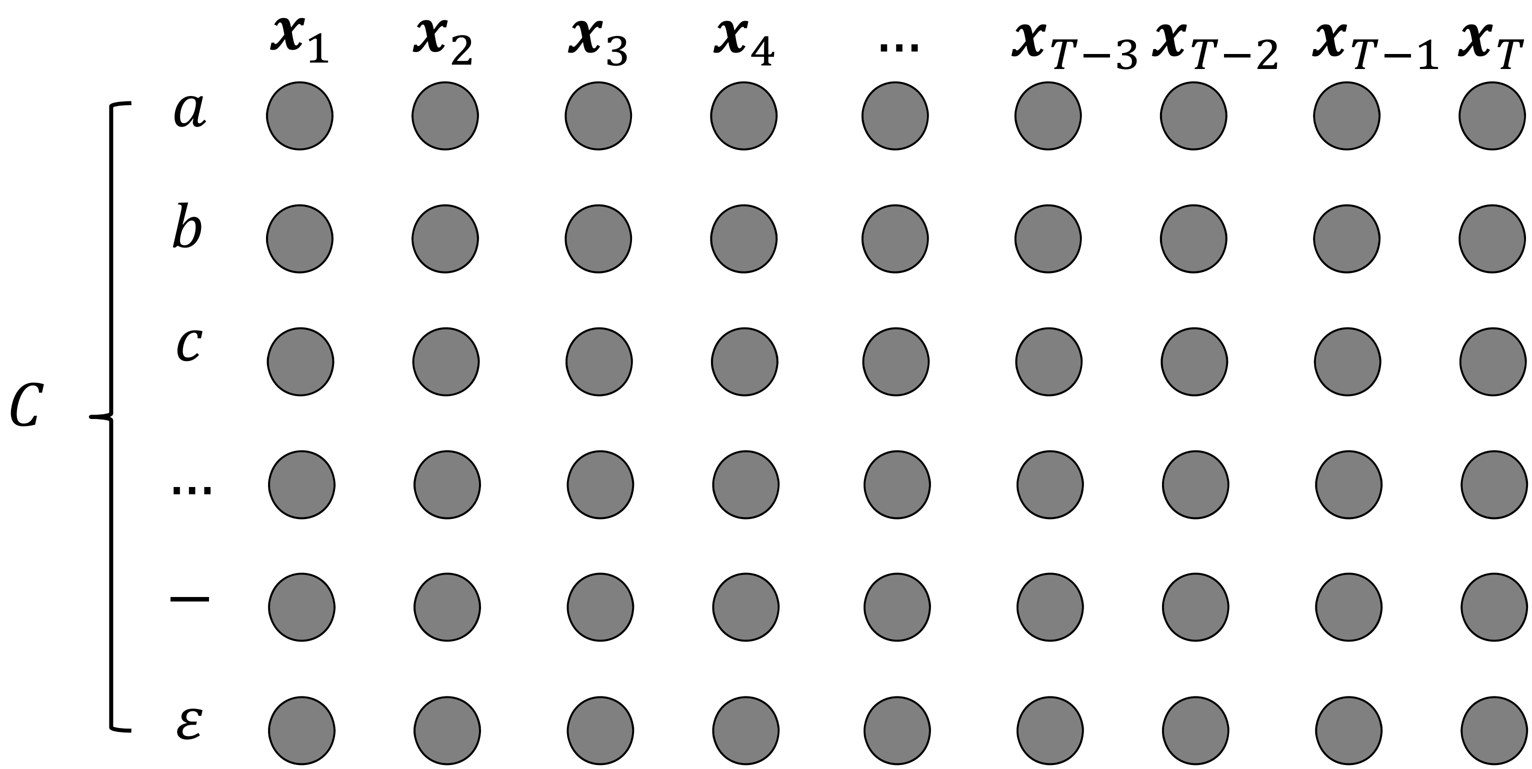
- End-to-End ASR:

$$\hat{W} = \underset{W \in V}{\operatorname{argmax}} P_{\Phi}(W|X) P_{\Theta}(W)^{\beta} = \underset{W \in V}{\operatorname{argmax}} \log P_{\Phi}(W|X) + \beta \log P_{\Theta}(W)$$

- But how to optimize the above objectives, thus searching for the optimal W?

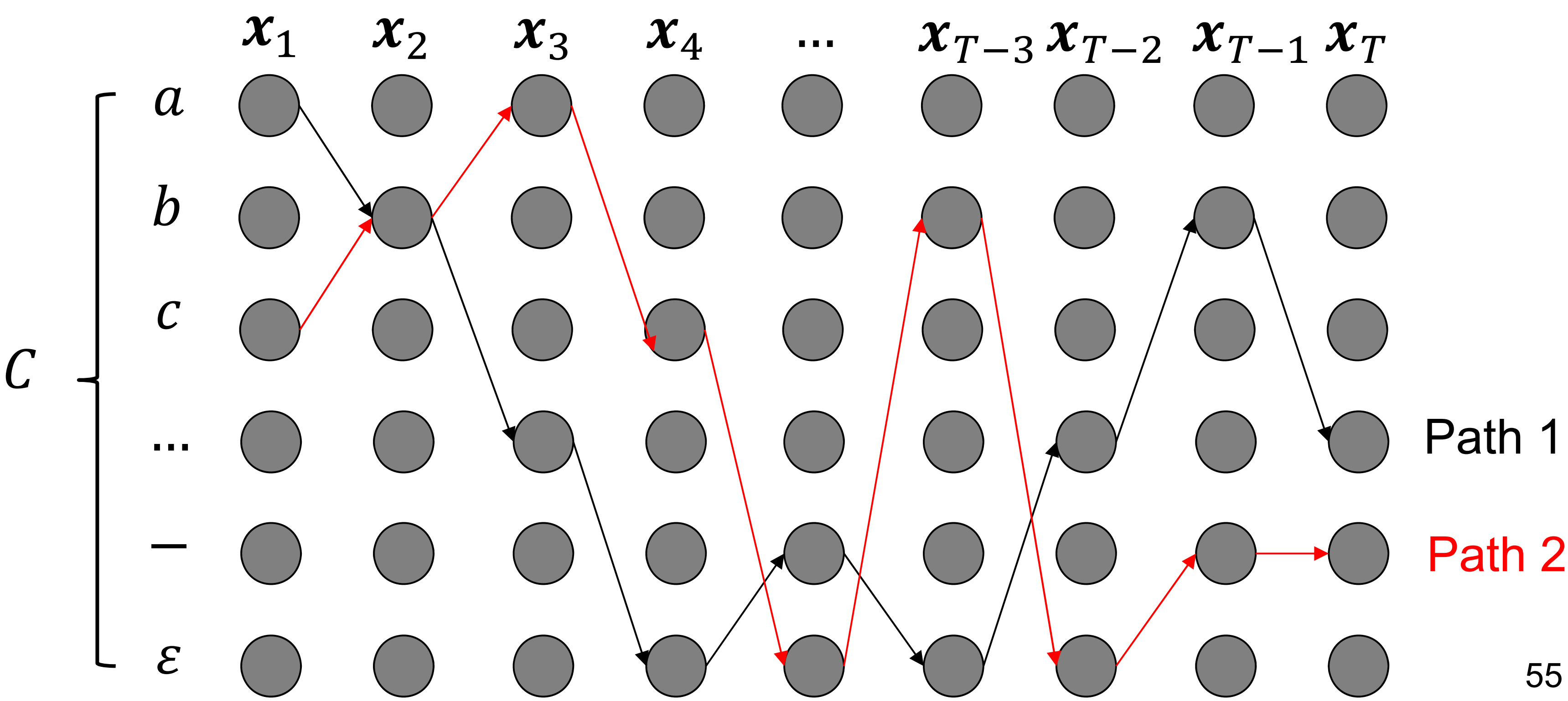
CTC Decoding

Take the CTC decoding for example. Suppose we use character as the token, X has T frames and the vocabulary has C tokens



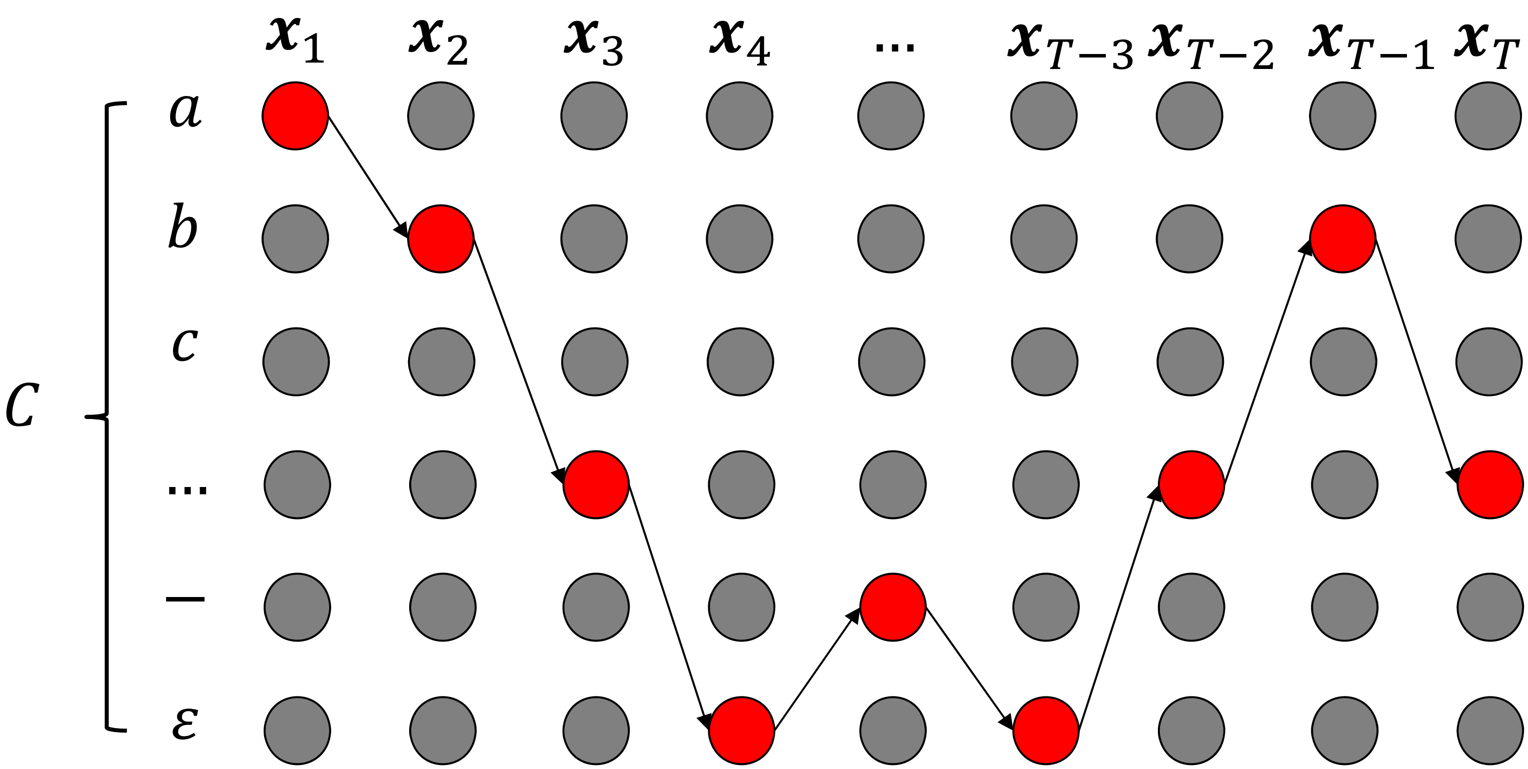
Brute Force

- Check all the possible path, combine all identical tokens (merge repeated tokens and remove blank token) and compare the probability of the token sequences
- Computational complexity is $O(C^T)$, best performance



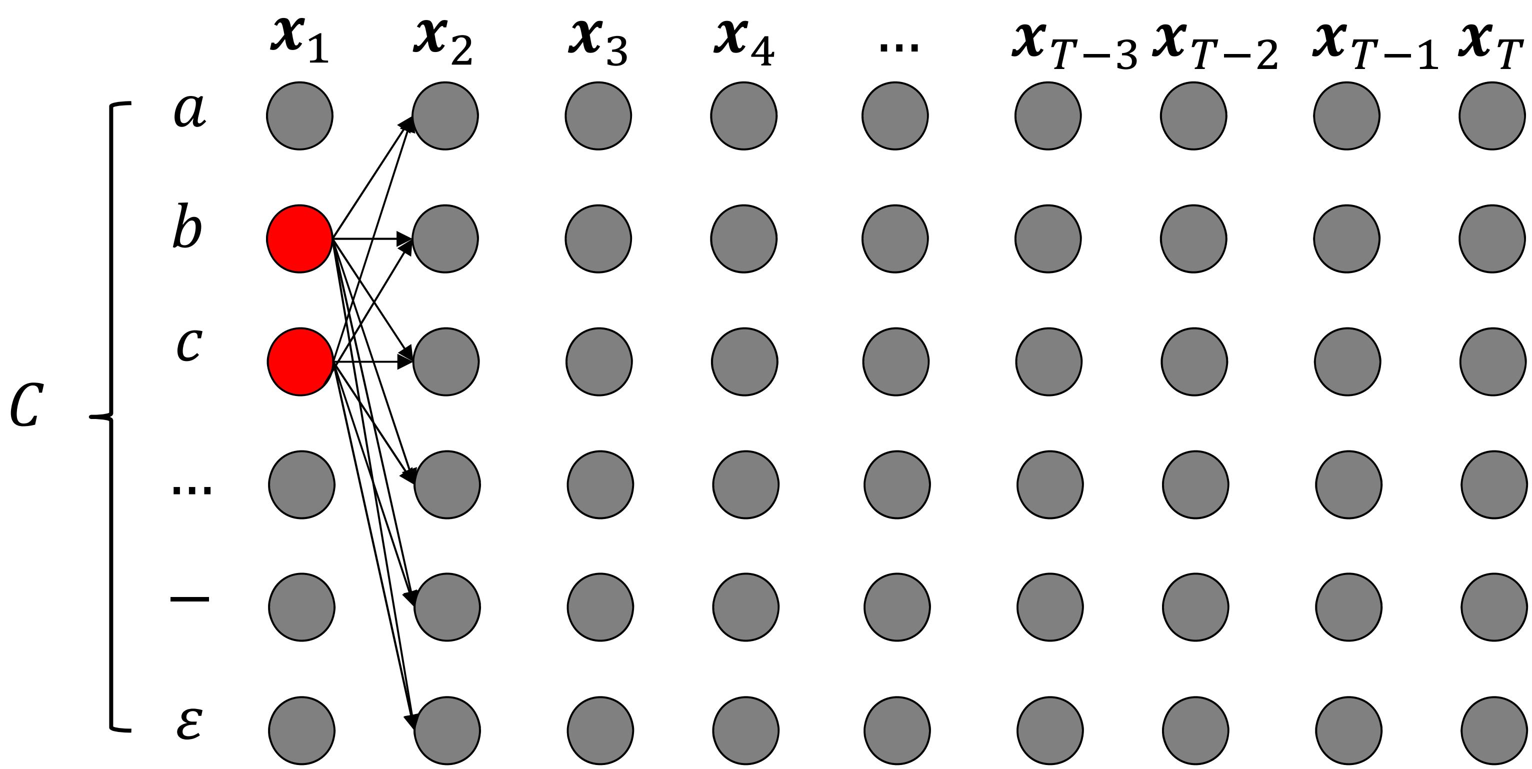
Best path Decoding

- Choose the most likely character per frame
- May not be the most probable token sequence (text transcript) because a text transcript corresponds to multiple paths
- Computational complexity is $O(TC)$, suboptimal performance



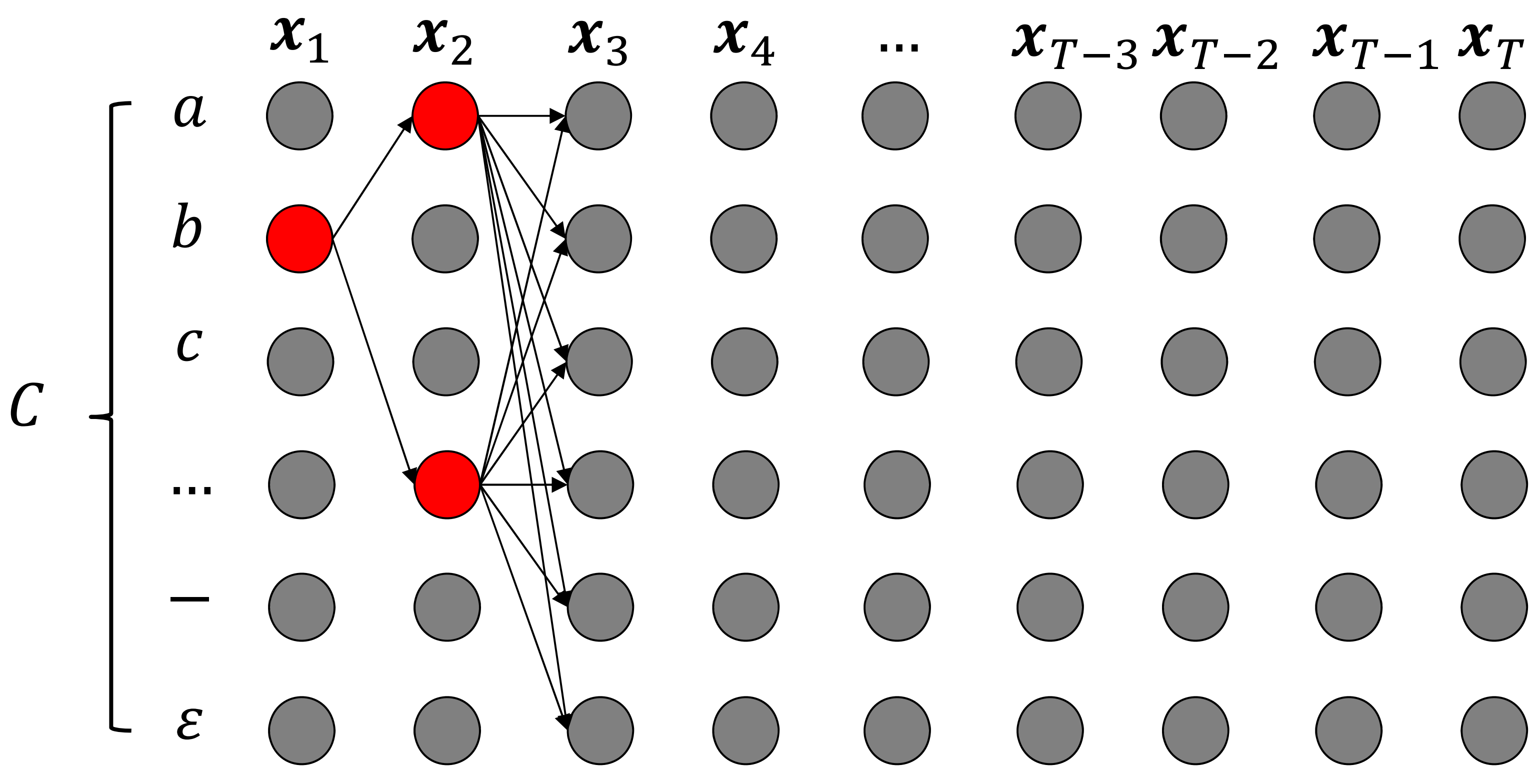
Beam Search

- At each frame, choose the ***B* text candidates (beams)** with highest probability and store them for the next frame
- NOTE: beams are the **text candidates not the alignment candidates**



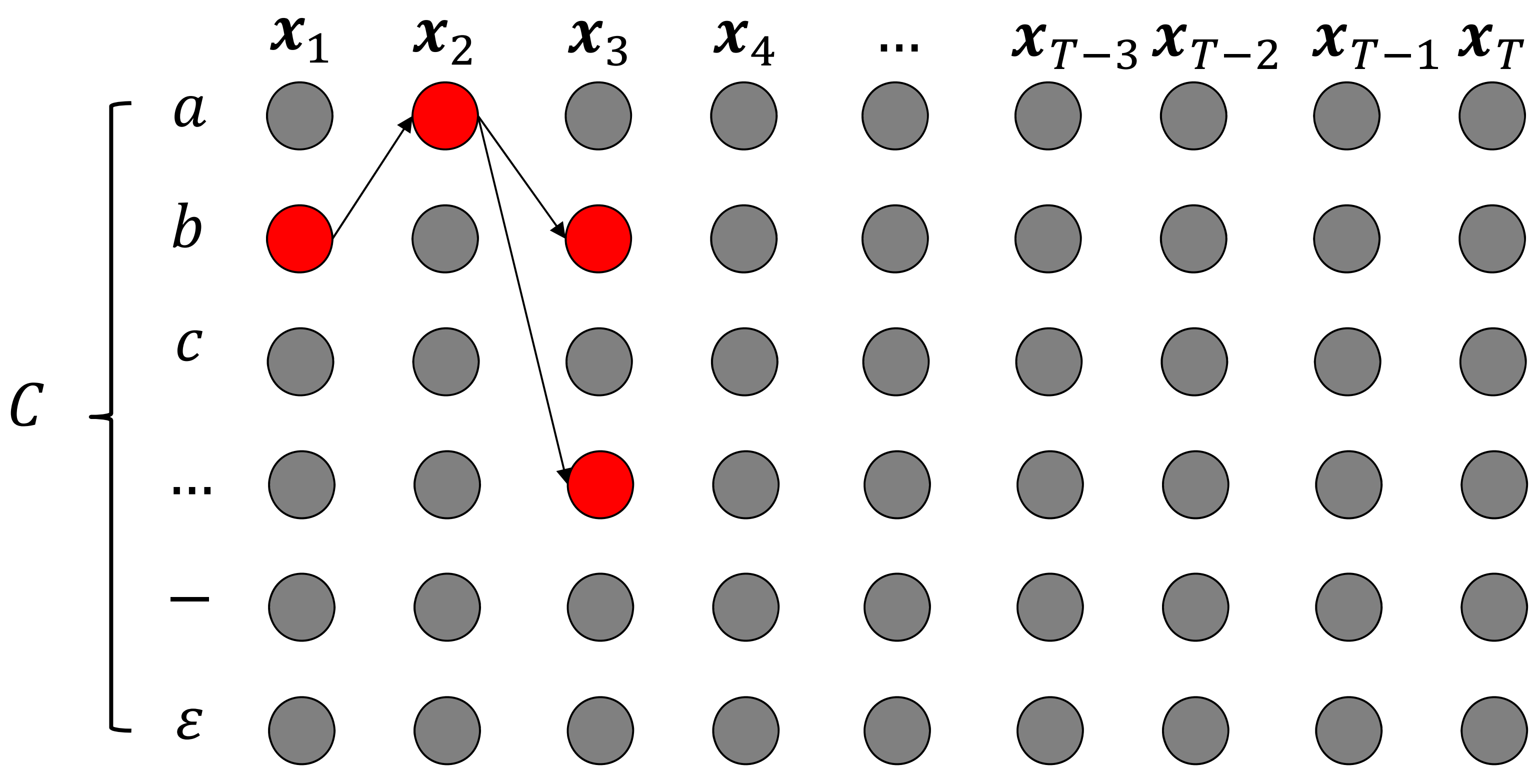
Beam Search

- At each frame, choose the ***B* text candidates (beams)** with highest probability and store them for the next frame
- NOTE: beams are the ***text candidates not the alignment candidates***



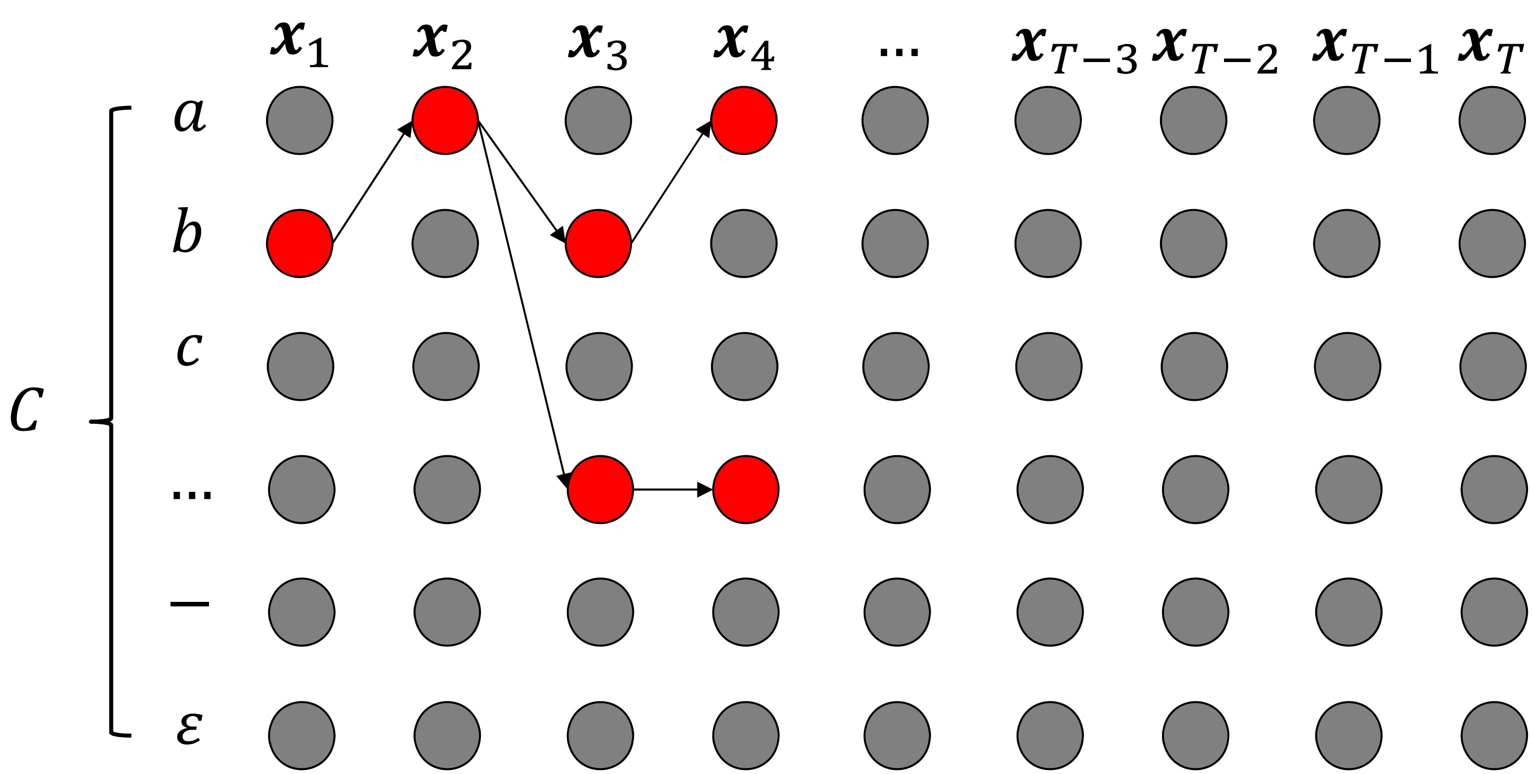
Beam Search

- At each frame, choose the ***B text candidates (beams)*** with highest probability and store them for the next frame
- NOTE: beams are the ***text candidates not the alignment candidates***



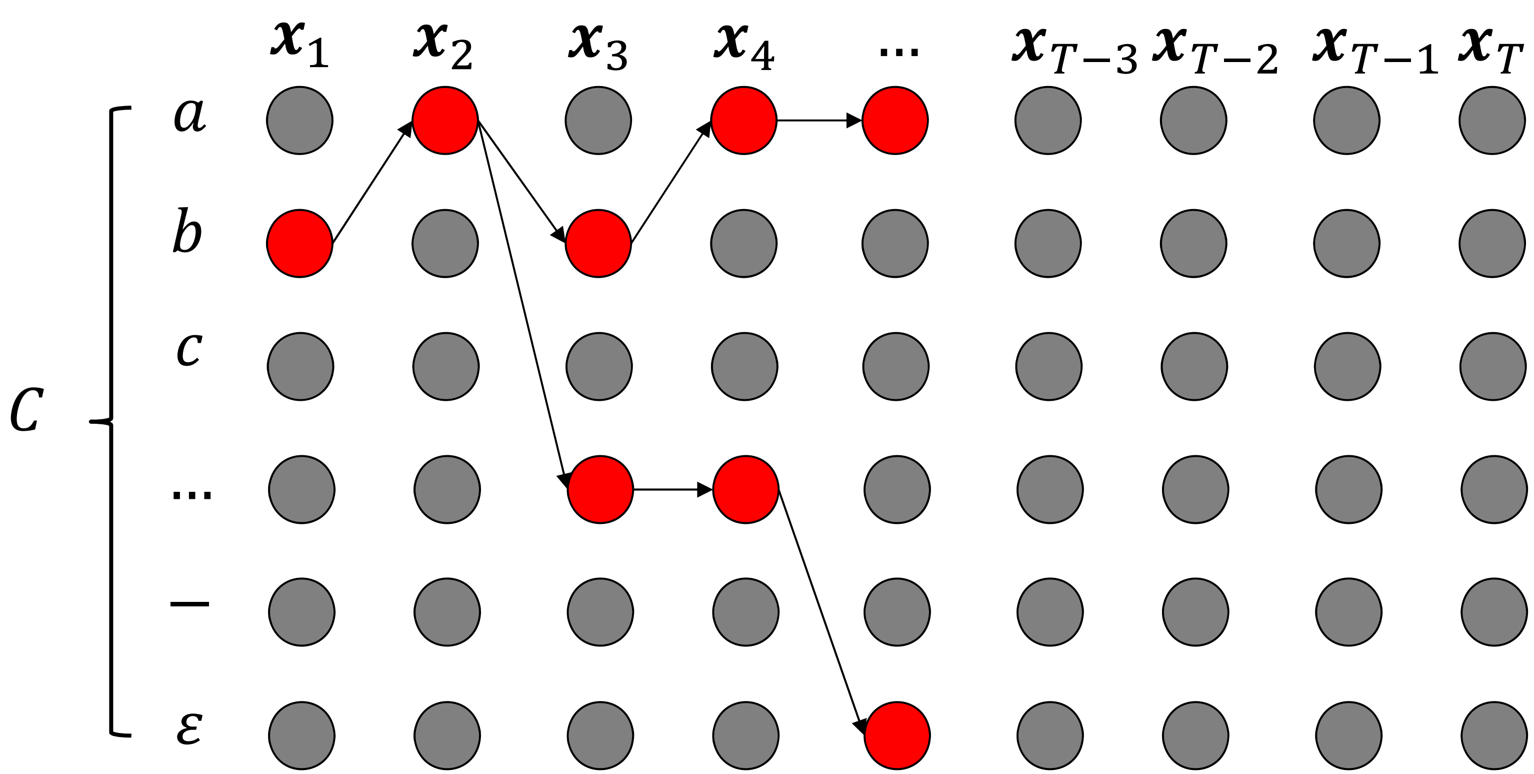
Beam Search

- At each frame, choose the ***B* text candidates (beams)** with highest probability and store them for the next frame
- NOTE: beams are the ***text candidates not the alignment candidates***



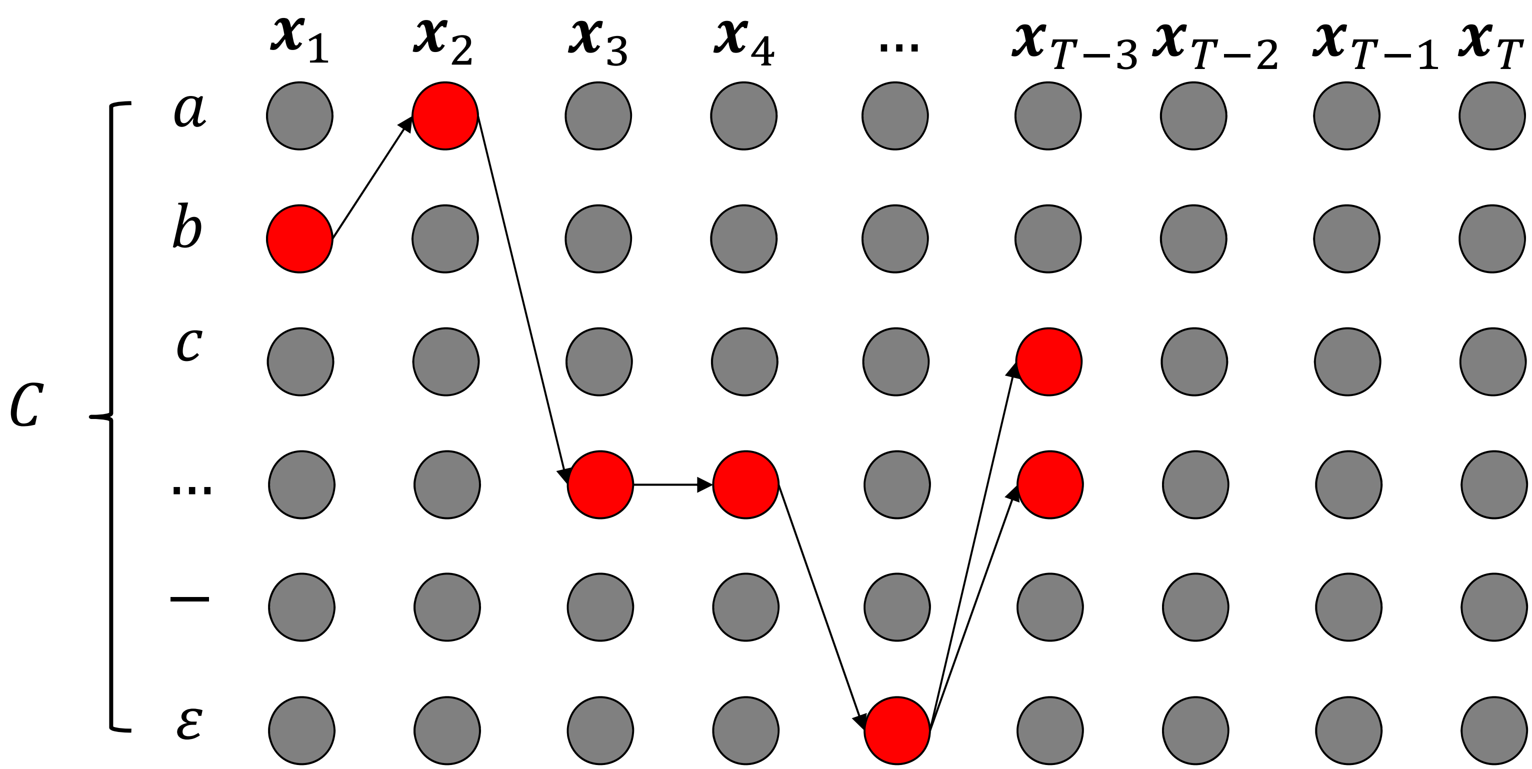
Beam Search

- At each frame, choose the ***B text candidates (beams)*** with highest probability and store them for the next frame
- NOTE: beams are the ***text candidates not the alignment candidates***



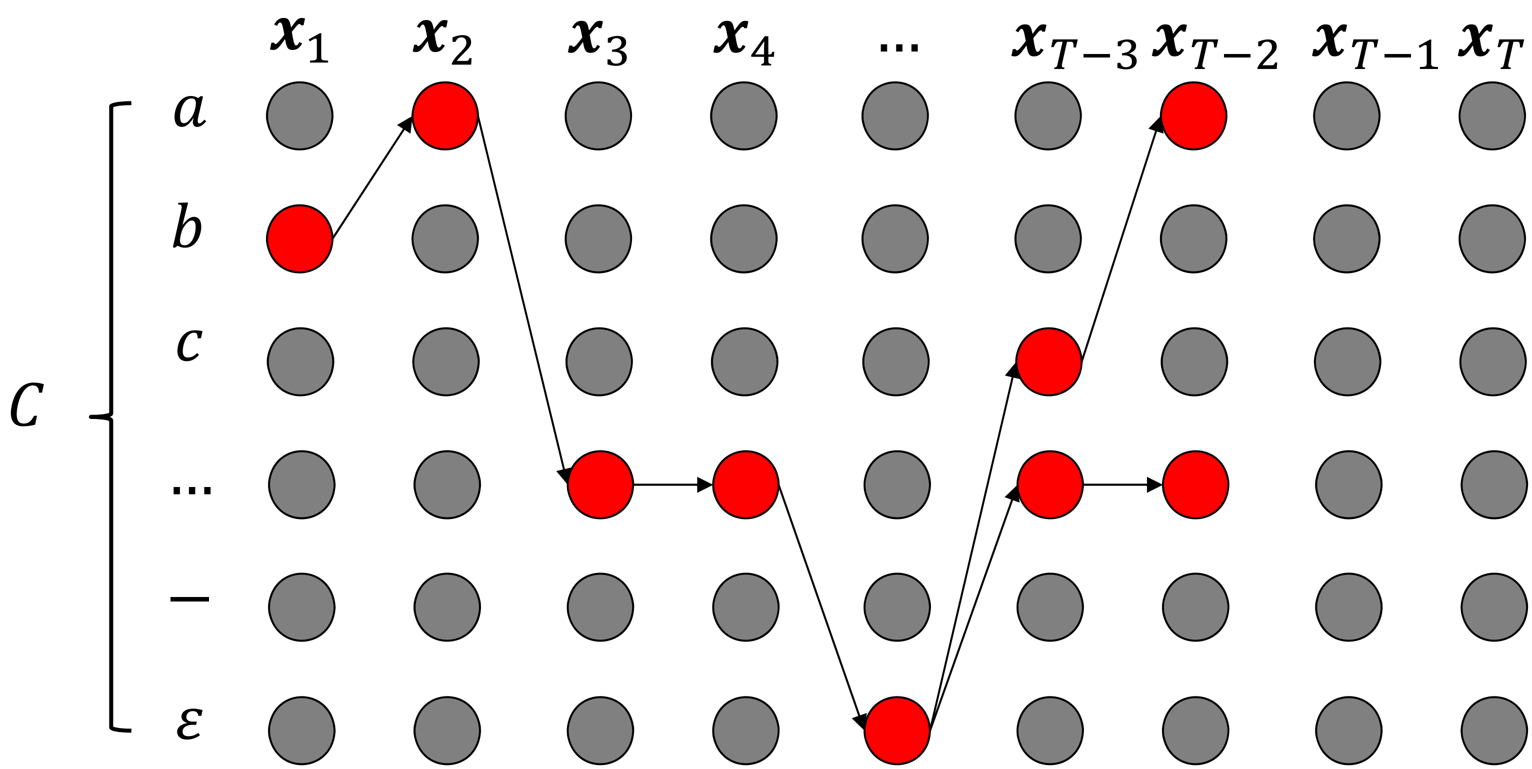
Beam Search

- At each frame, choose the ***B text candidates (beams)*** with highest probability and store them for the next frame
- NOTE: beams are the ***text candidates not the alignment candidates***



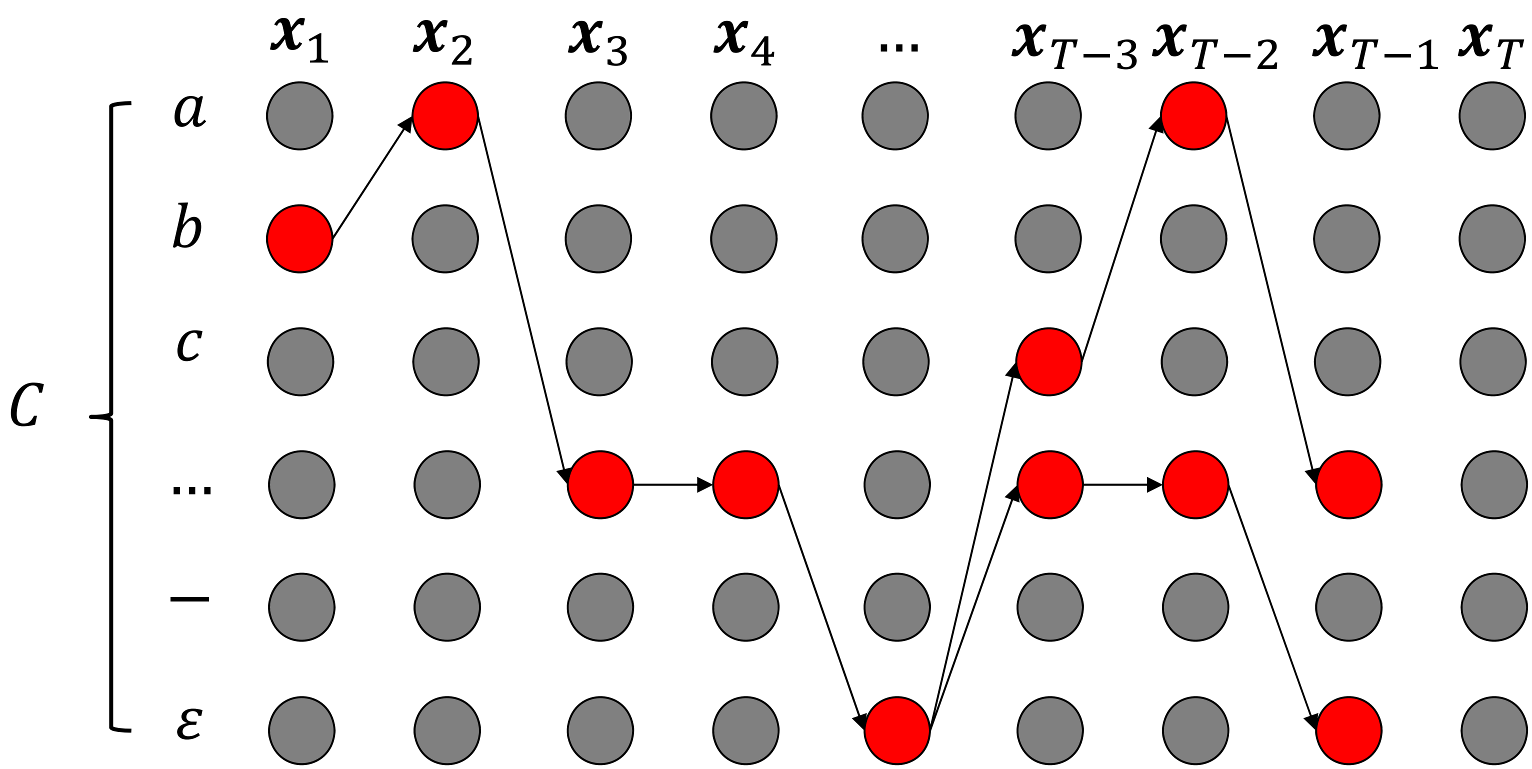
Beam Search

- At each frame, choose the ***B text candidates (beams)*** with highest probability and store them for the next frame
- NOTE: beams are the ***text candidates not the alignment candidates***



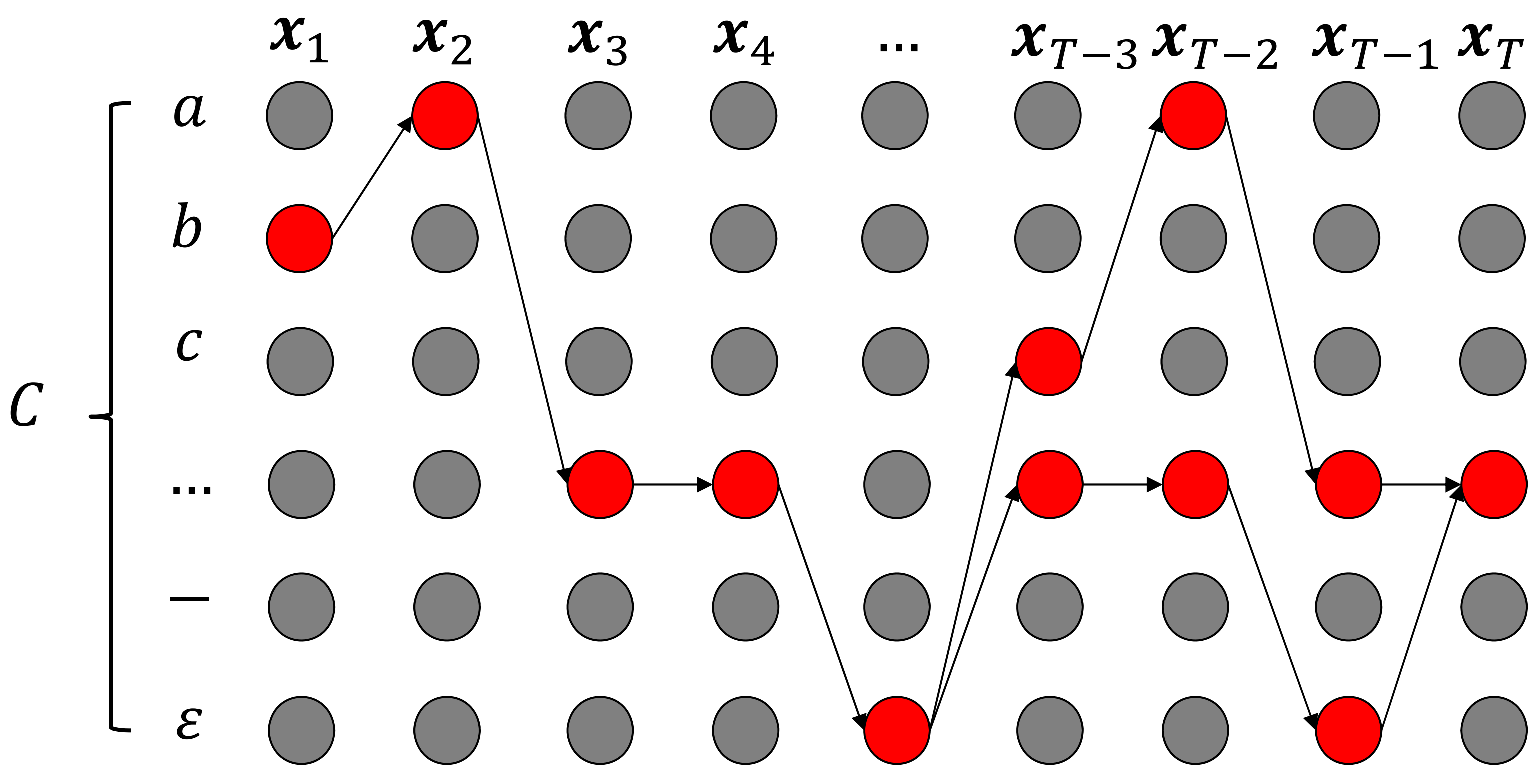
Beam Search

- At each frame, choose the ***B* *text candidates (beams)*** with highest probability and store them for the next frame
- NOTE: beams are the ***text candidates not the alignment candidates***



Beam Search

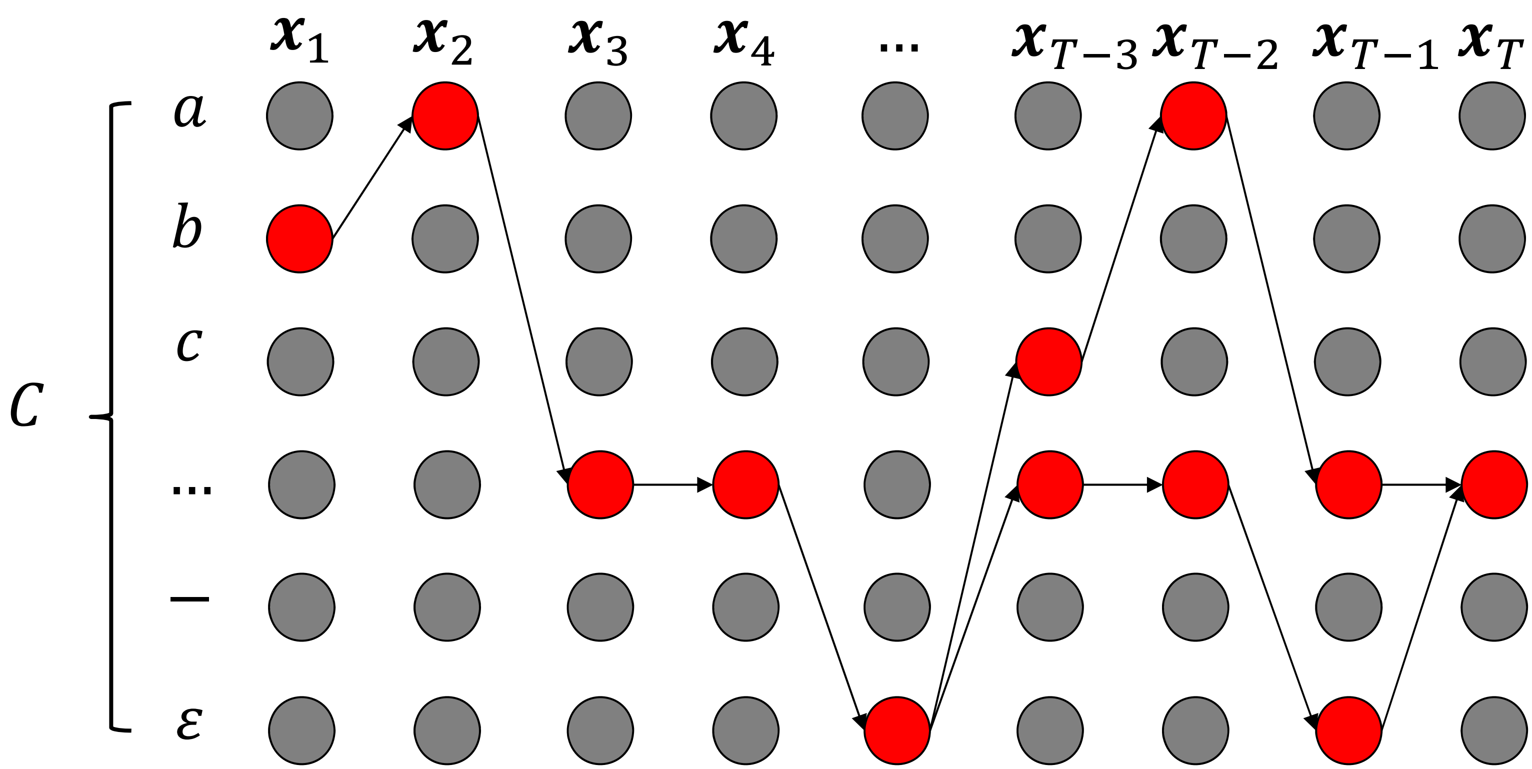
- Finally, choose the best beam as the prediction
- Computational Complexity: $O(TCB)$



Beam Search

- Please refer to the following material for detailed explanation

https://towardsdatascience.com/beam-search-decoding-in-ctc-trained-neural-networks-5a889a3d85a7



Topics Today

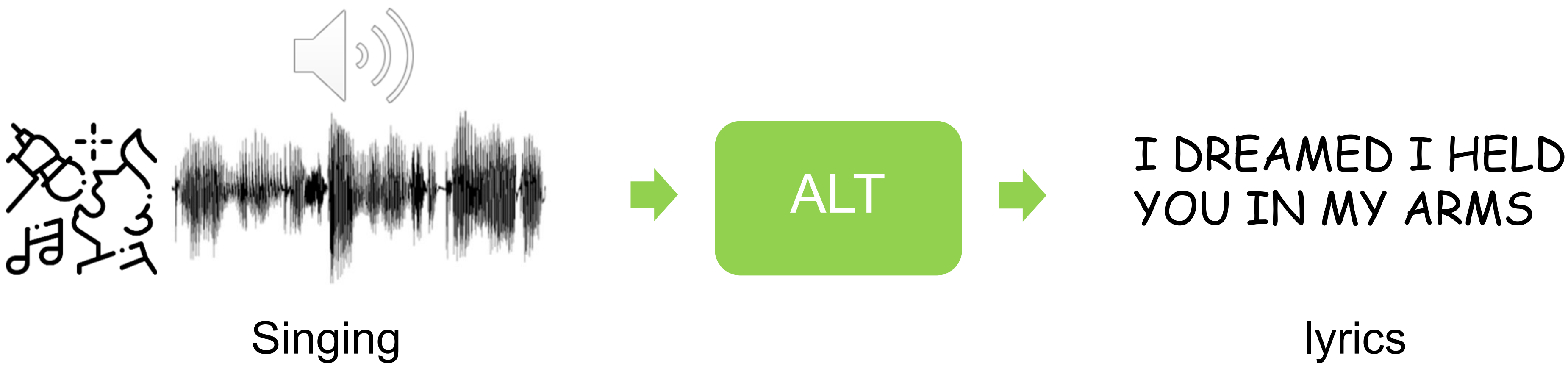
Part A: Overview of Automatic Speech Recognition (ASR)

Part B: Development of ASR System

Part C: End-to-End ASR System

Part D: Automatic Lyric Transcription (ALT)

What is Automatic Lyric Transcription (ALT)?



- ALT task aims to transcribe a singing waveform into lyrics
- A counterpart task of ASR

Relationship between ALT and ASR

Similarity:

- Both ASR and ALT aim to transcript the text from audio
- Singing voice and speech are produced by the same organ
- The target transcription have the same vocabularies

Discrepancies:

- Singing voice is less intelligible and harder to be recognized
- Pitch and duration are different

[1] Zhang C, Yu J, Chang L C, et al. PDAugment: Data Augmentation by Pitch and Duration Adjustments for Automatic Lyrics Transcription[J]. arXiv preprint arXiv:2109.07940, 2021.

[2] Gu X, Ou L, Ong D, Wang Y. MM-ALT: A Multimodal Automatic Lyric Transcription System[C]//Proceedings of the 30th ACM International Conference on Multimedia. 2022.

How to build an ALT system?

Take advantage of similarities between ASR and ALT

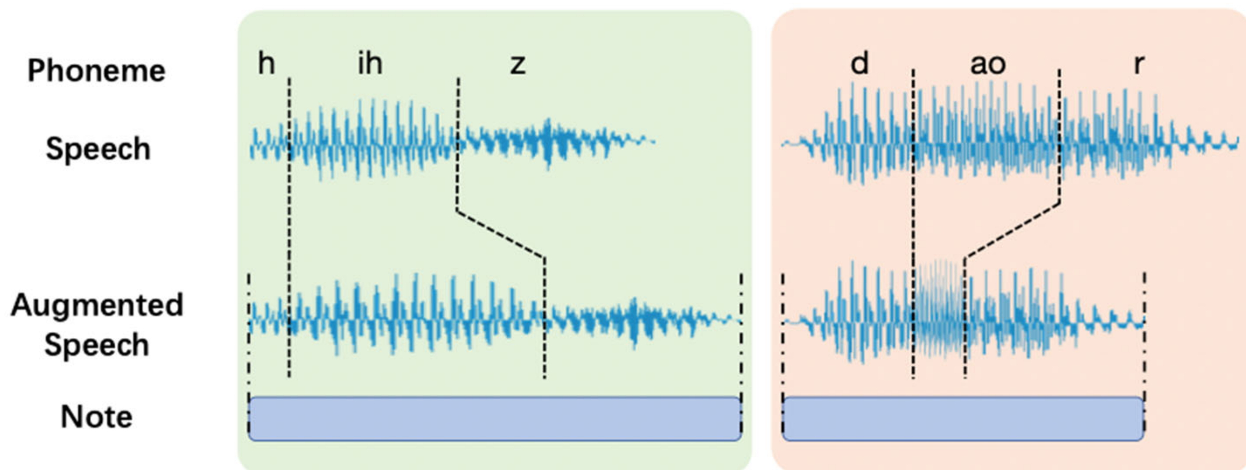
- Adopt the same pipeline to train an ALT system

How to deal with the discrepancies

- Adjust speech data to generate “song-like” data
- Adapt the knowledge from ASR model to ALT model

Example 1: PDAugment

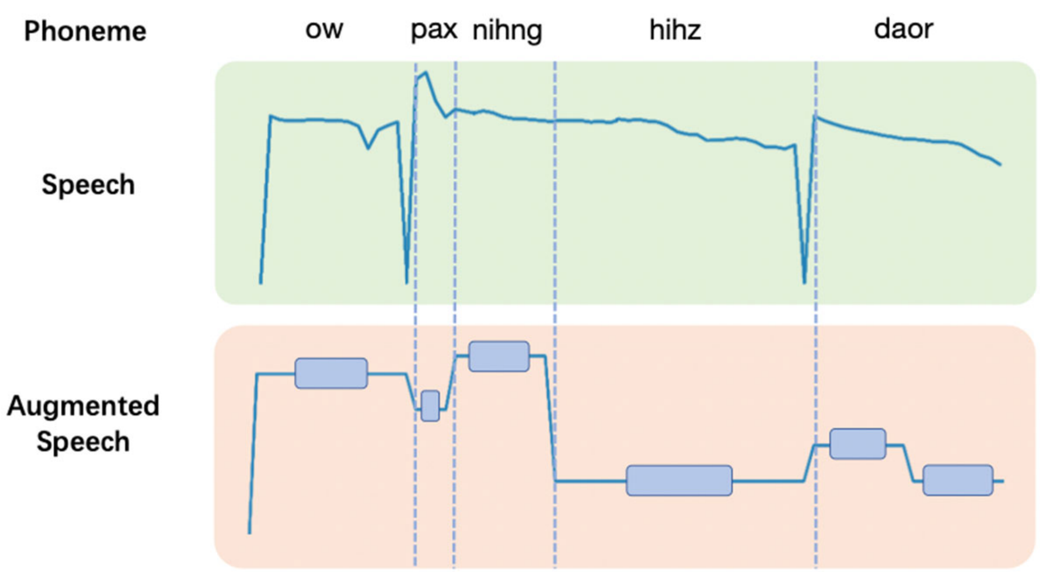
➢ Generate “song-like” training data by adjusting ***duration*** of speech



[1] Zhang C, Yu J, Chang L C, et al. PDAugment: Data Augmentation by Pitch and Duration Adjustments for Automatic Lyrics Transcription[J]. arXiv preprint arXiv:2109.07940, 2021.

Example 1: PDAugment

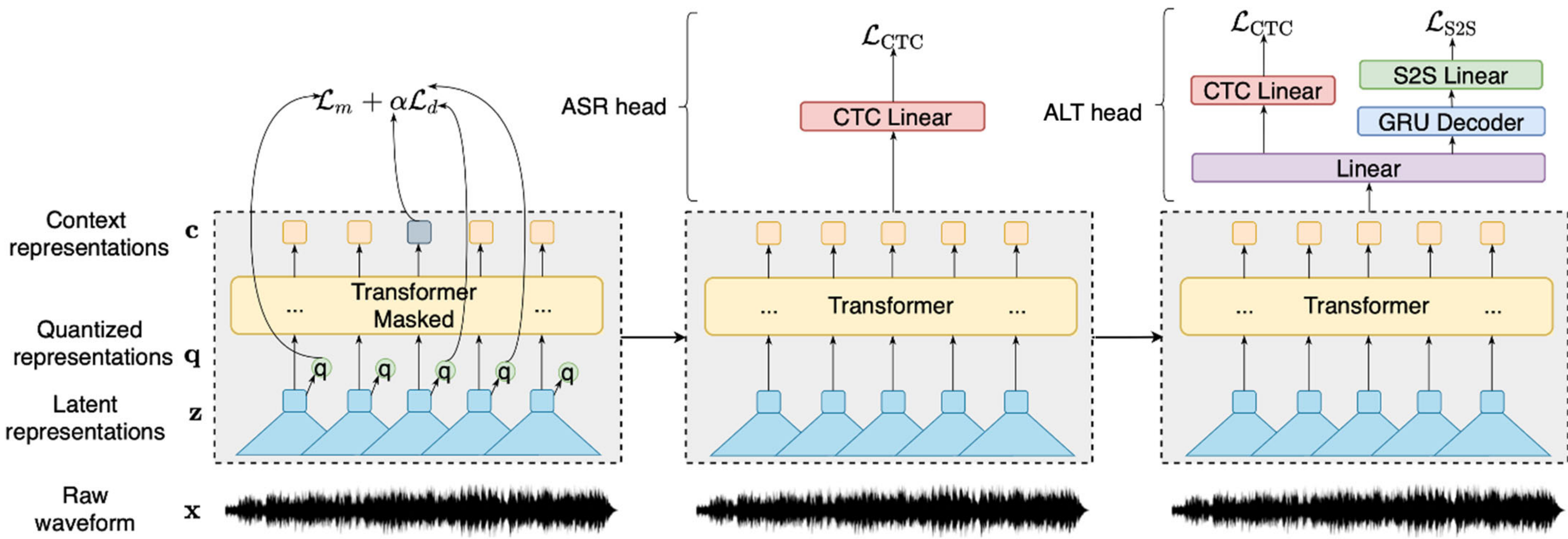
➢ Generate "song-like" training data by adjusting ***pitch*** of speech



[1] Zhang C, Yu J, Chang L C, et al. PDAugment: Data Augmentation by Pitch and Duration Adjustments for Automatic Lyrics Transcription[J]. arXiv preprint arXiv:2109.07940, 2021.

Example 2: Transfer Learning

- Fetch a wav2vec 2.0 after pre-training and fine-tuning on speech data
- Modify the head network and finetune the model on singing data
- Inherit the low-resource learning property



[1] Gu X, Ou L, Ong D, Wang Y. MM-ALT: A Multimodal Automatic Lyric Transcription System[C]//Proceedings of the 30th ACM International Conference on Multimedia. 2022.

Take home message

To implement ASR and ALT systems, please consider the following platforms:

- SpeechBrain: https://github.com/speechbrain/speechbrain (recommended)
- Fairseq: https://github.com/facebookresearch/fairseq (more powerful but more challenging)

If interested, please also consider the following material:

- Tutorial on SpeechBrain: https://colab.research.google.com/drive/1aFgzrUv3udM_gNJNUoLaHIm78QHtxdIz?usp=sharing#scrollTo=gzNb1cZzvxUh
- Stanford lecture: https://www.youtube.com/watch?v=3MjIkWxXigM

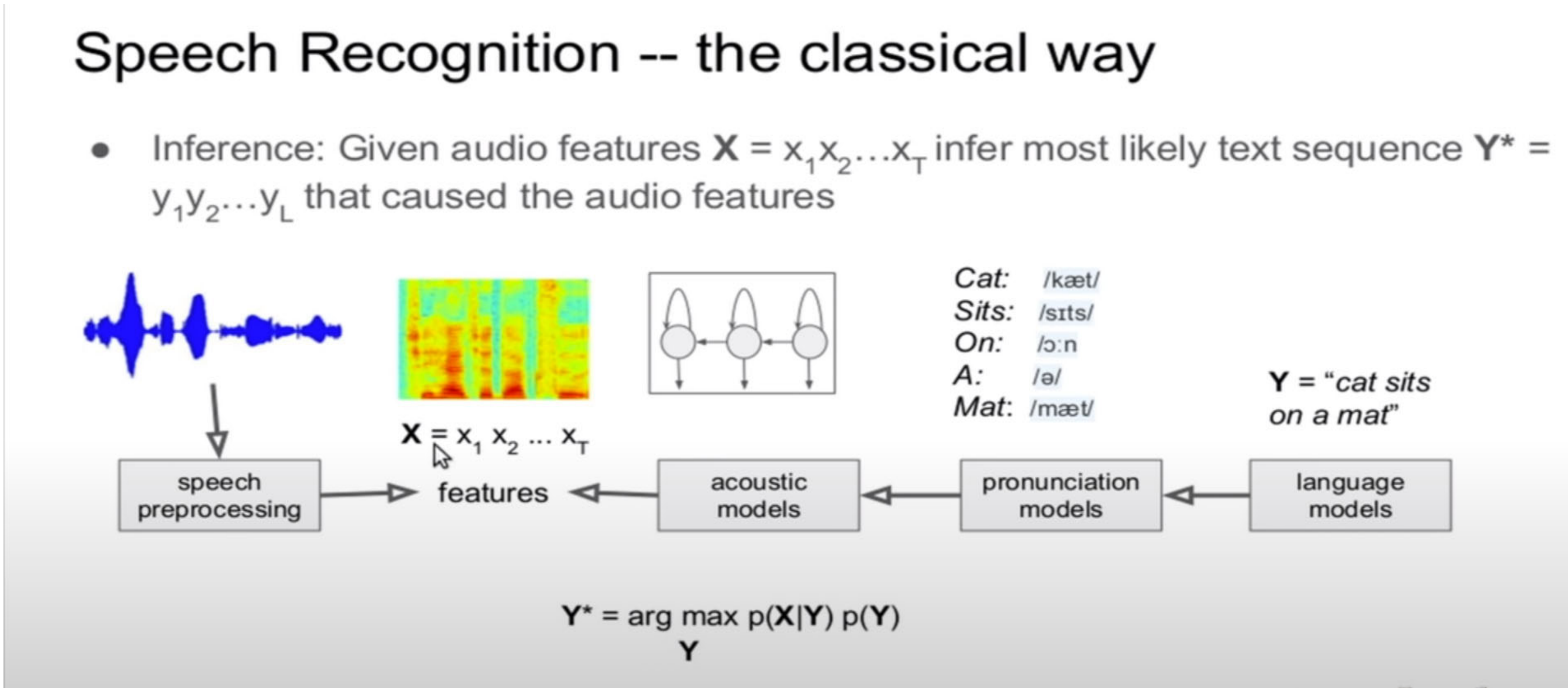
Appendix 1: A nice introduction to Bayes Theorem, HMM and Viterbi Decoding.

You are encouraged to watch it!

A friendly introduction to Bayes Theorem and Hidden Markov Models
By Luis Serrano
https://www.youtube.com/watch?v=kqSzLo9fenk

Appendix 2: End-to-End Models for Speech Processing

https://www.youtube.com/watch?v=3MjIkWxXigM



Audio Features HMM-GMM Dictionary ***Text***

Here is another nice tutorial on ASR:

https://www.youtube.com/watch?v=q67z7PTGRi8

Appendix 3: An intuitive introduction to language model!

The A.I. Hacker - Michael Phi

I Built a Personal Speech Recognition System for my AI Assistant

https://www.youtube.com/watch?v=YereI6Gn3bM

Appendix 4: What is a language model then?

Language modelling is to assign probability values to sequences of words.
A subject studied by *computational linguists*.

The probability values help us to answer the following questions, for example:

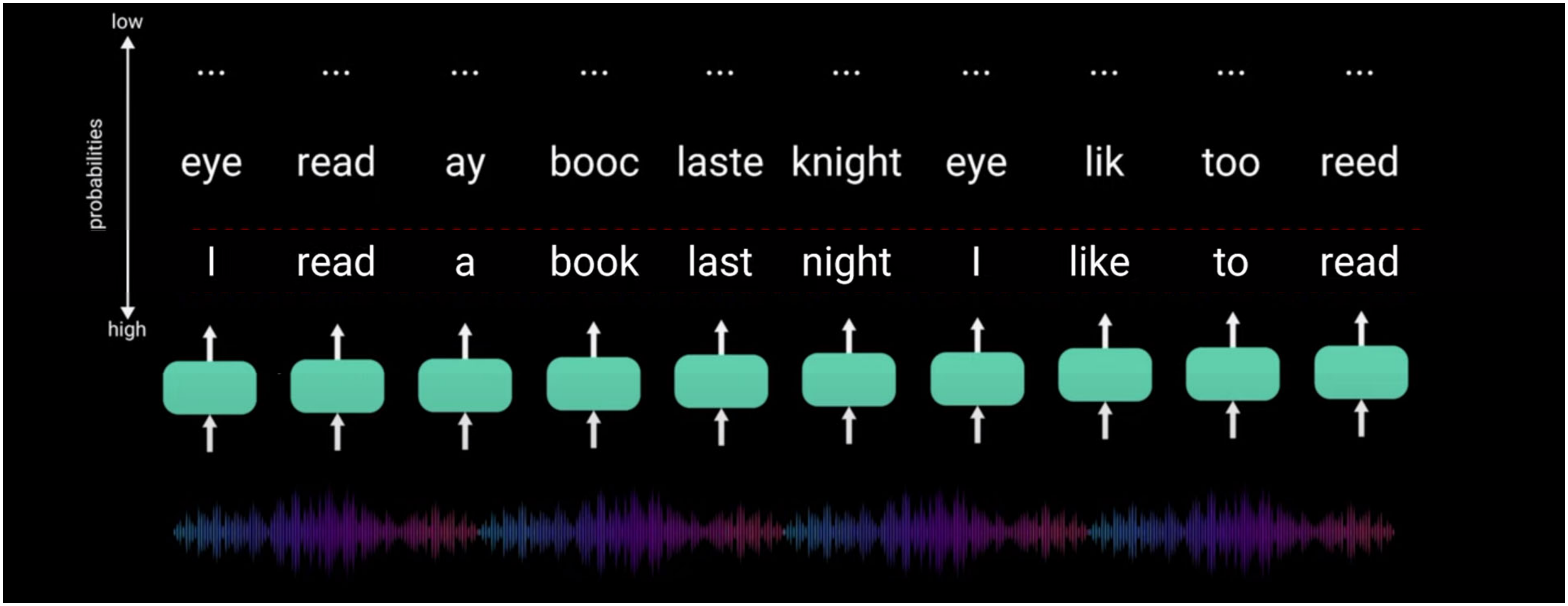
Which sentence is grammatically correct?

P("he eat pizza") < P("he eats pizza")

Which word order is correct?

P("love I cats") < P("I love cats")

Appendix 5: If we only have an acoustic model



A pronunciation token can be different words!

Appendix 6: Language model explained

Whats a more likely sentence?

Probability(I read a book) = 0.95

Probability(I red a book) = 0.25

Appendix 7: Language model explained

hypothesis (beams)

Probability(i read a book last night i like to read) = 0.95

Probability(i red a book last night i like to read) = 0.35

Probability(i red a book last night i like to reed) = 0.15