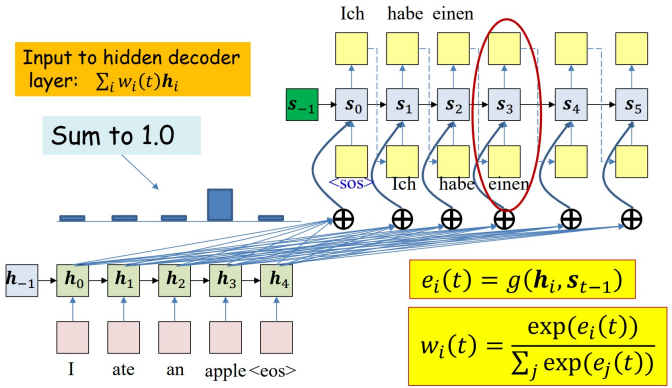
RNN-T Based ASR Systems

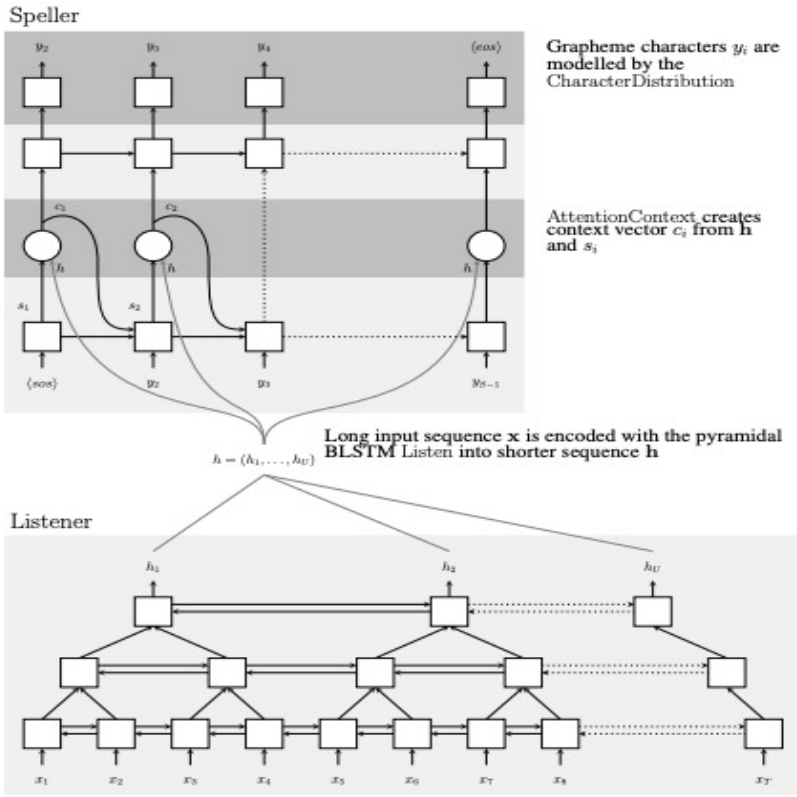
Mahaveer Jain, Facebook

ASR And Machine Translation As Sequence To Sequence Problem



Machine Translation

Image from Bhiksha Raj's lecture on attention models



LAS attends to all audio embeddings and uses text history produced so far to generate probability distribution over output units

$$f(y_u | y_{u-1}, y_{u-2}, \ldots, y_0, h_0, h_2, h_3, \ldots, h_v)$$

ASR

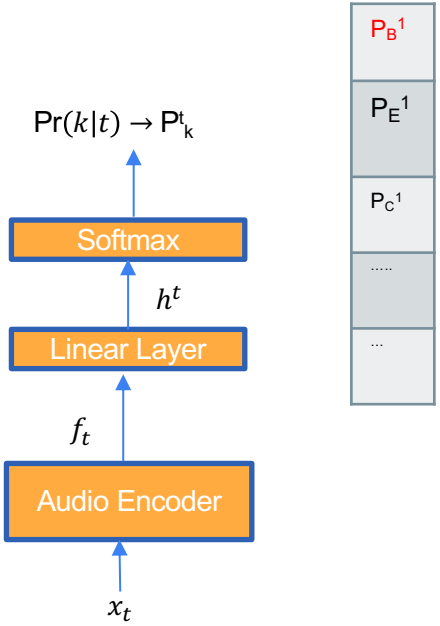
Image from Chan et al., "Listen, Attend and Spell"

ASR As Sequence To Sequence Problem

- Alignment not known during training
- At each audio frame index, neural network generates probability of producing each output symbol. e. g. Probability of generating output symbol “k” at 1st frame.

$$P_K^1 = prob(s_1 = K \mid f_1)$$

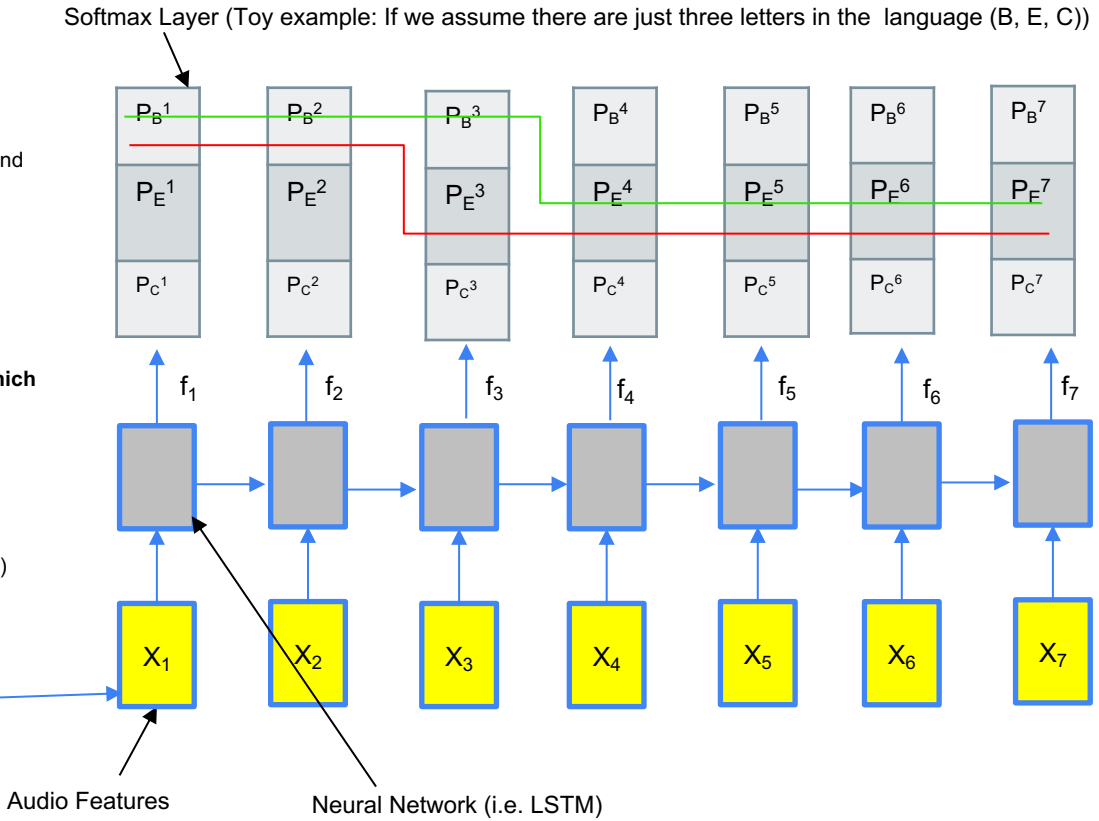
- #audio frames != # letters. We also don’t know which portion of audio was emitted when.
- Long short-term memory (LSTM), B-LSTM (bi-directional LSTM) and, Transformer are some of the commonly used audio encoders.



ASR As Sequence To Sequence Problem

- Input Sequence: Audio frame features
- Output Sequence : Words from True Transcript Text written as letters
- Example Training Instance: Input Sequence : 7 audio frames and True Transcript Text: “BE”
- At each audio frame index, network* generates probability of producing each output symbol. E. g. Probability of generating output symbol “K” at 1st frame.
- #au $P_K^1 = prob(s_1 = K \mid f_1)$ ıent i.e. which portion of audio aligns to what letter in true transcript.
- Collapse continuous occurrences of same letter into one:
  - First alignment choice (BBEEEEE -> BE)
  - Second alignment choice (BBBEEEE -> BE)
  - and so on…..
- How to deal with repeated letters (as in word *BEE*) ? (next slide)

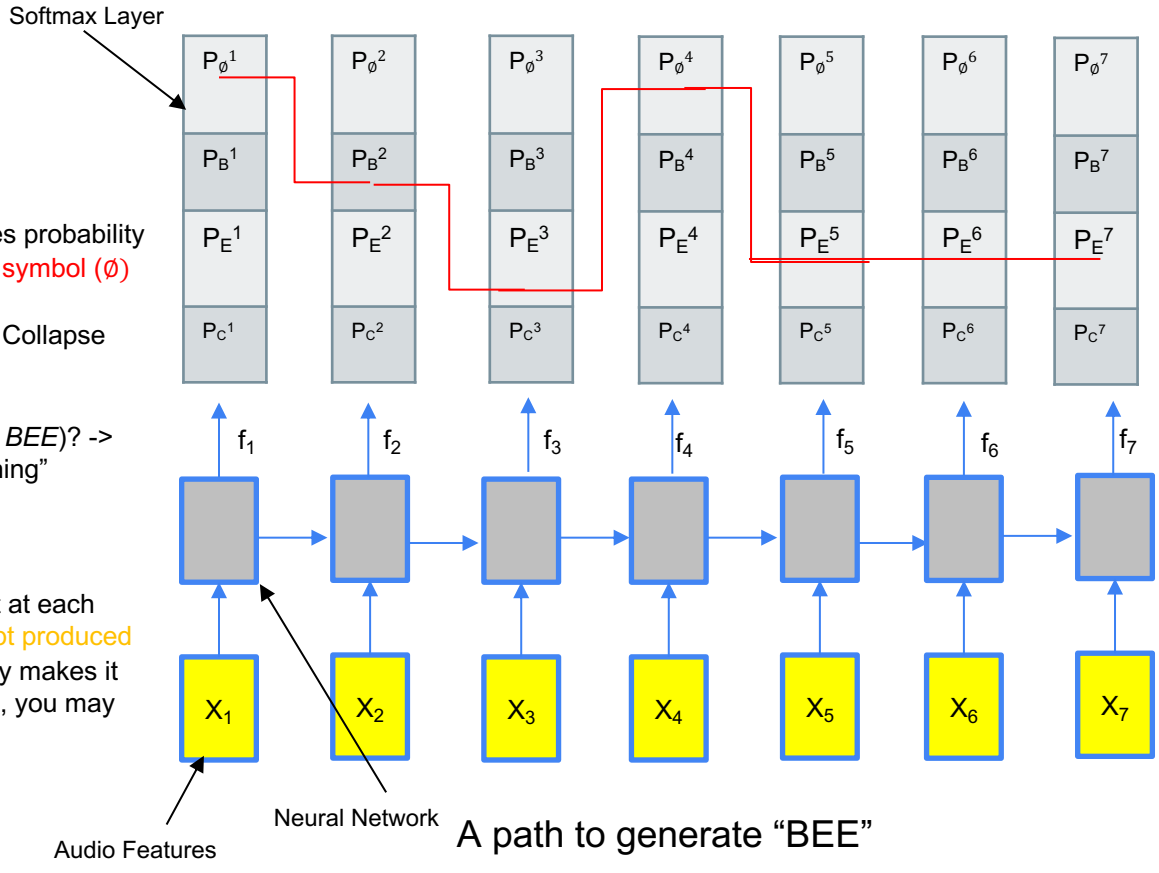
Audio



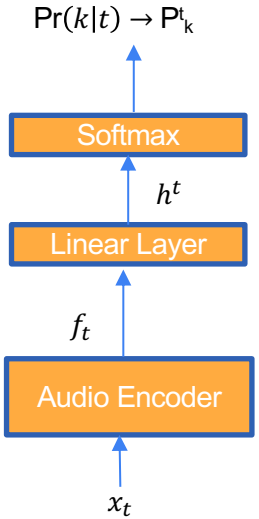
network* -> Softmax layer and neural network that produces f_t

Connectionist Temporal Classification (CTC)

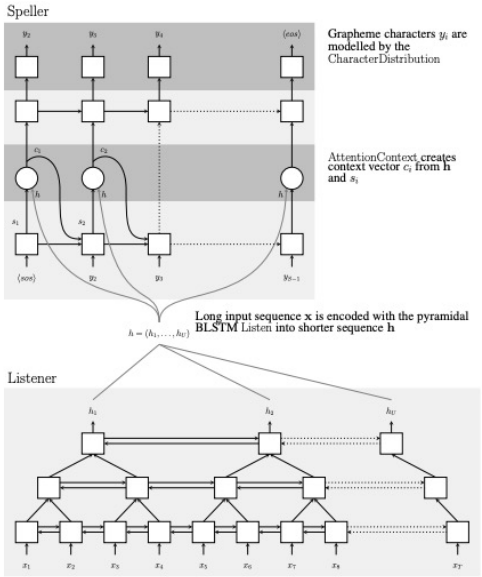
- Input Sequence: Audio frame features
- Output Sequence: Letters
- At each audio frame index, network generates probability of producing each output symbol and blank symbol (ø)
- How to deal with #audio frame != # letters -> Collapse letters
- How to deal with repeated letters (as in word *BEE*)? -> blank symbol (ø) . ø also indicates “emit nothing” (silence). øBEøEEø
- Conditional Independence: Probability output at each frame does not depend on history of transcript produced so far, this non auto-regression on text history makes it less effective on learning LM information and, you may still need language model -> RNN-T.



Conditional Independence Assumption



>



CTC:

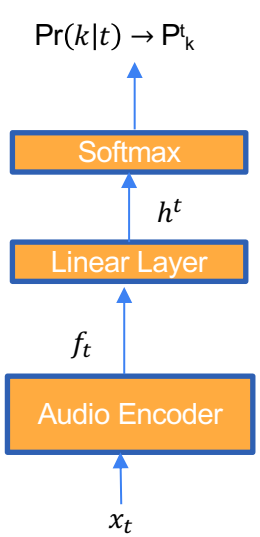
- Only uses audio embedding at time “t” (x_t) to generate probability distribution over output units. CTC assumes that each output unit is conditionally independent of others.
- Streaming

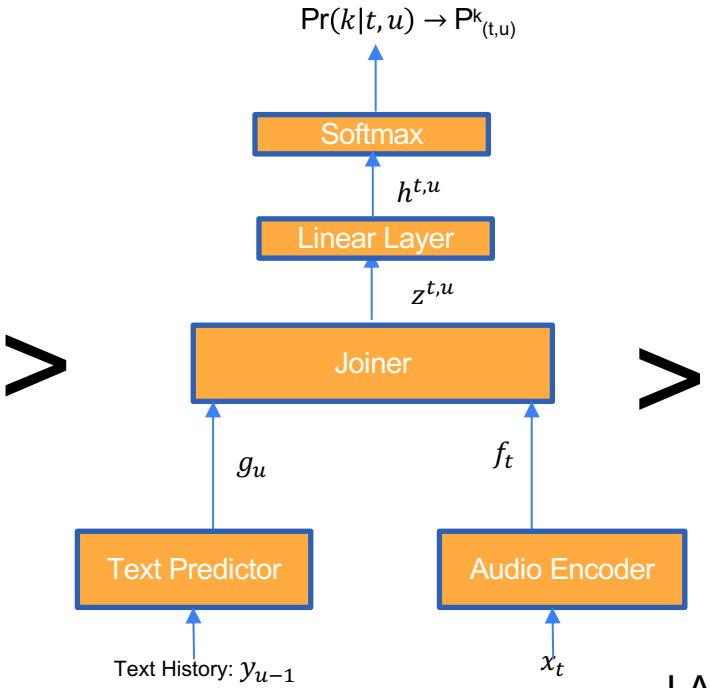
LAS:

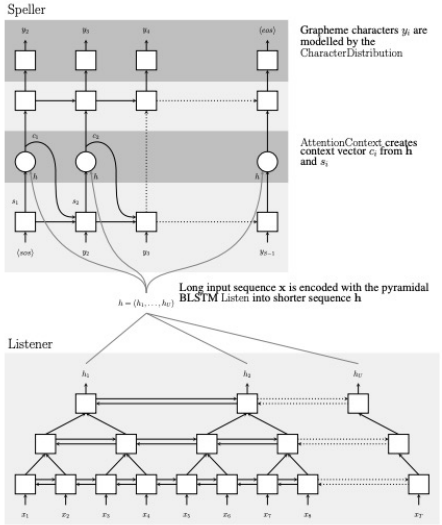
Attends to all audio embeddings and uses text history produced so far to generate probability distribution over output units.

$$f(y_u|y_{u-1}, y_{u-2}, \ldots, y_0, h_0, h_2, h_3, \ldots, h_v)$$

Conditional Independence Assumption







CTC:

- Only uses audio embedding at time "t" (x_t) to generate probability distribution over output units. CTC assumes that each output unit is conditionally independent of others.
- Streaming

RNN-T:

- Uses both audio embedding at time "t" (x_t) and text history produced so far (y_{u-1}) to generate probability distribution over output units.
- Streaming

LAS:

Attends to all audio embeddings and uses text history produced so far to generate probability distribution over output units.

$$f(y_u|y_{u-1}, y_{u-2}, \ldots, y_0, h_0, h_2, h_3, \ldots, h_v)$$

Why RNN-T ASR From Production Point Of View

- Single deployable non modularized neural model, all components of ASR in one model.
- Allows compact on-device streaming ASR. Does not need a decoder graph which can be large. Unlimited words in vocab. Standard on-device ASR choice across industry.

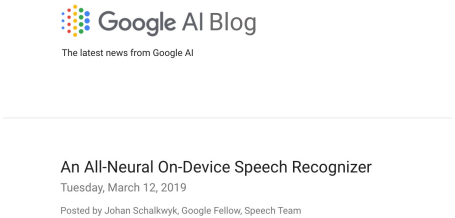


Image from https://ai.googleblog.com/2019/03/an-all-neural-on-device-speech.html

- Achieves comparable accuracy and compute with much smaller size model compared to modularized (hybrid) systems for production when training data is the same.

Table 3. *Comparison of Hybrid model with RNN-T*

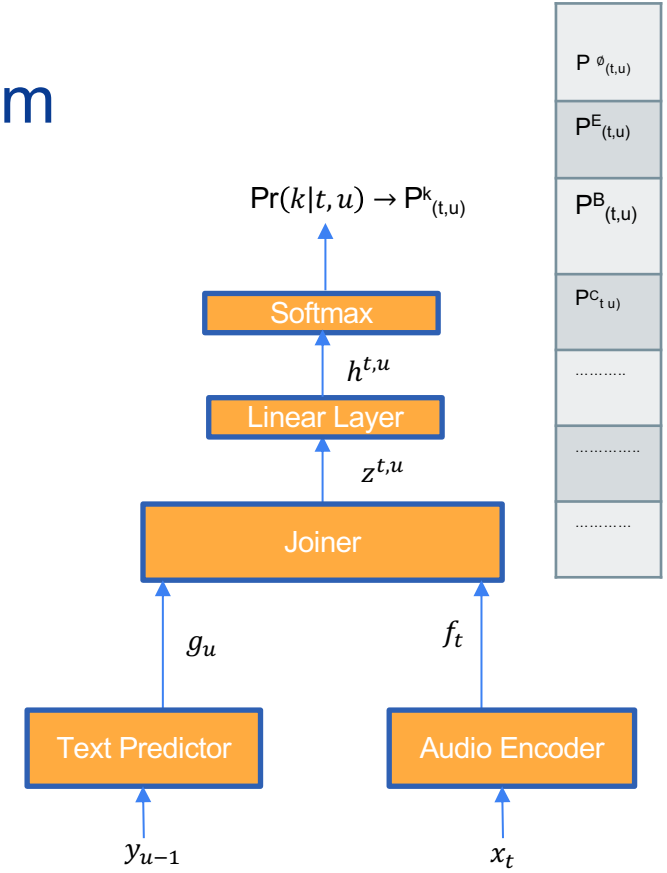
Test Set	System	WER	Throughput	rtf@40
vid-clean	hybrid	14.0	55	.70
vid-clean	RNN-T	14.0	63	.60
vid-noisy	hybrid	20.7	55	.71
vid-noisy	RNN-T	21.0	65	.60

Example Study

Image from Jain et al., "RNN-T For Latency Controlled ASR With Imoroved Beam Search"

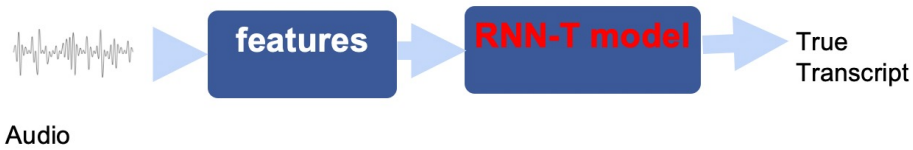
Components Of RNN-T ASR System

- Audio Encoder: To encode sequence of audio features into audio embeddings. Long short-term memory (LSTM), B-LSTM (bi-directional LSTM), Transformer are commonly used audio encoders. Acts as acoustic model.
- Text Predictor: To encode transcript produced so far into text embedding. Typically a LSTM. Acts as language model.
- Joiner combines output of audio encoder and text predictor.
- A Linear Layer followed by Softmax produces probability distribution over output units. $\Pr(k|t,u)$($P^{k}_{(t,u)}$ is probability of emitting “k” from (t, u).
- How is ∅ different from other symbols: If emitted output symbols is blank (∅) then move to next time frame else stay in the same time frame. ∅ also indicates “emit nothing”.

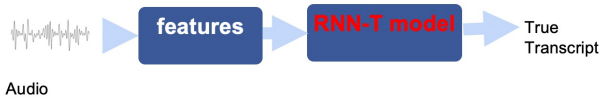


President Bara

Training



Training Objective



- Training utterance:
  “Input Audio” -> “BEE” (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- We don’t know alignment i.e. which portion of audio aligns to what output unit (A path taken in lattice)

- Training Objective

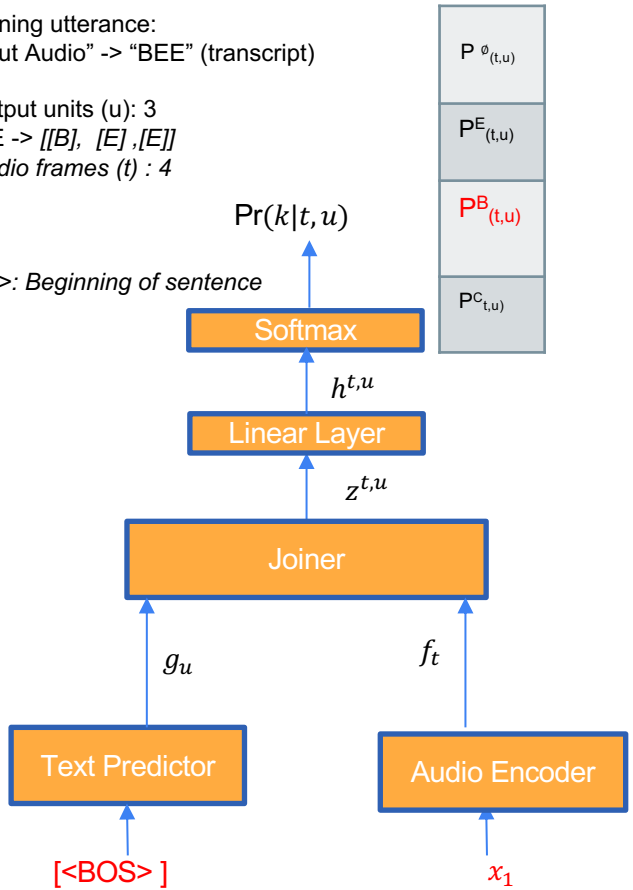
$$P(BEE|X) = \sum_{alignment} P(alignment, BEE|X)$$

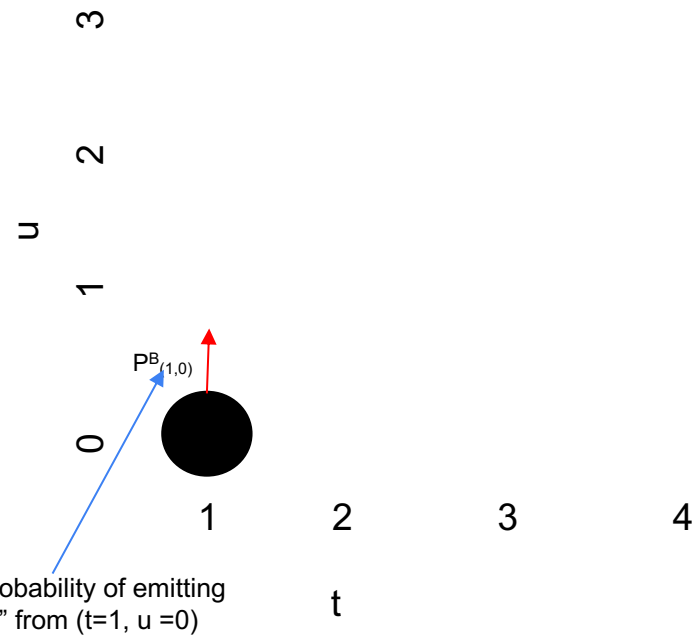
$$P(BEE|alignment, X) = 1 \quad \frac{\sum_{alignment} P(BEE|alignment, X)P(alignment|X)}{\sum_{alignment} P(alignment|X)}$$

Graves et al., “Sequence transduction with recurrent neural networks”

An Alignment During Training

- Training utterance:
  "Input Audio" -> "BEE" (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- *<BOS>: Beginning of sentence*

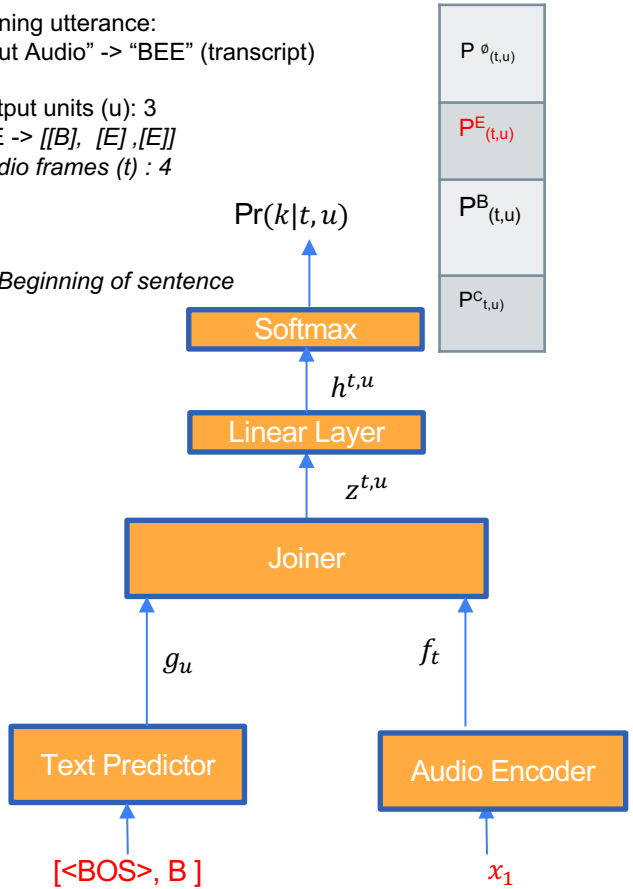


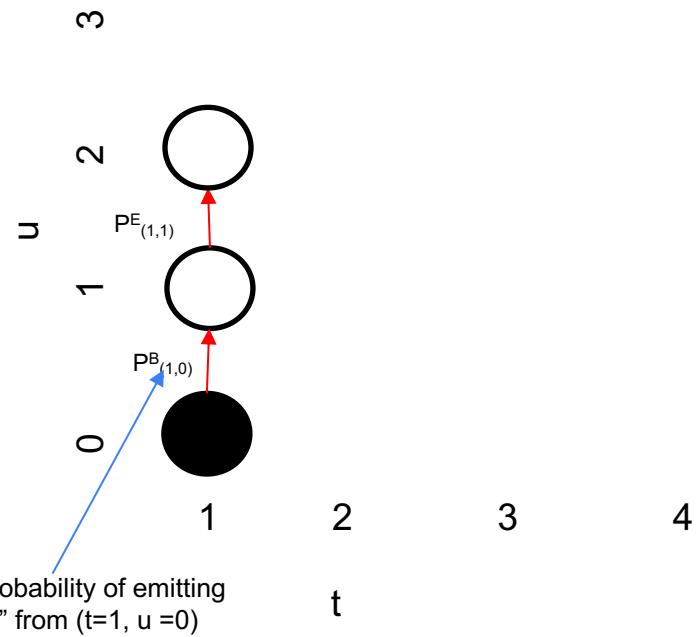


Graves et al., "Sequence transduction with recurrent neural networks"

An Alignment During Training

- Training utterance:
  "Input Audio" -> "BEE" (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- *BOS: Beginning of sentence*

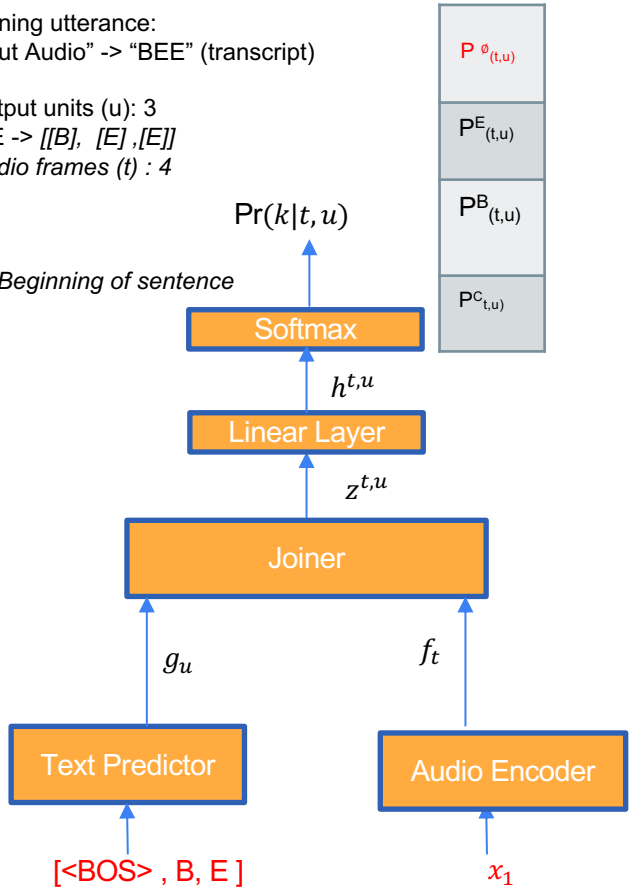


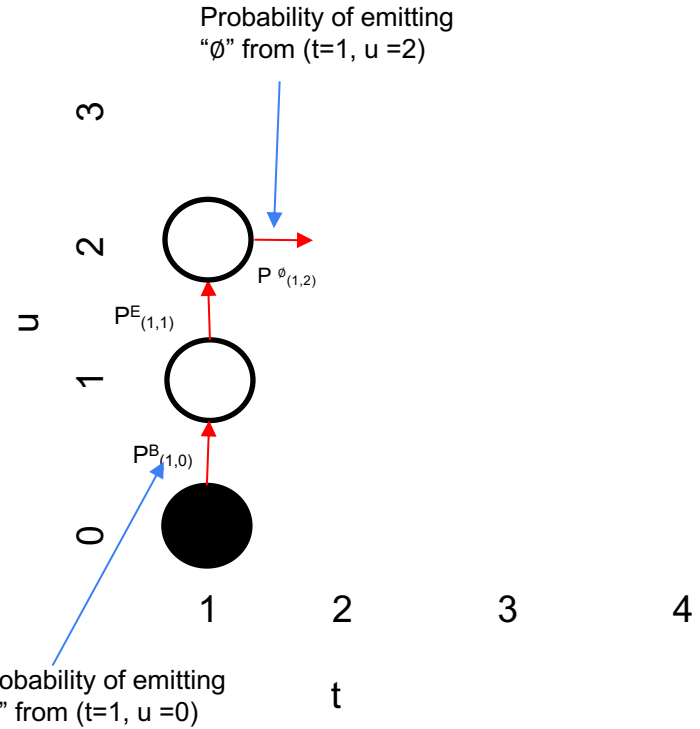


Graves et al., "Sequence transduction with recurrent neural networks"

An Alignment During Training

- Training utterance:
  "Input Audio" -> "BEE" (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- *BOS: Beginning of sentence*

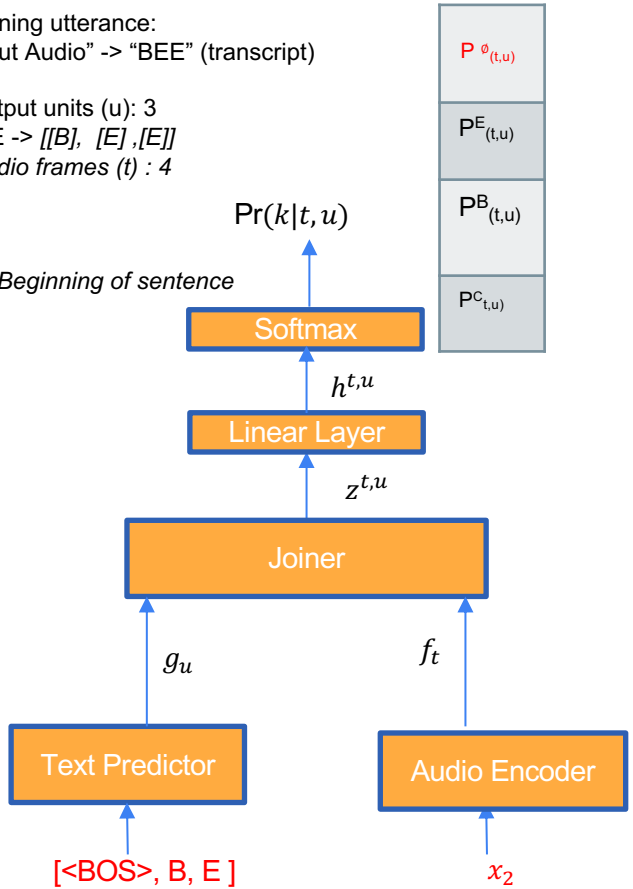


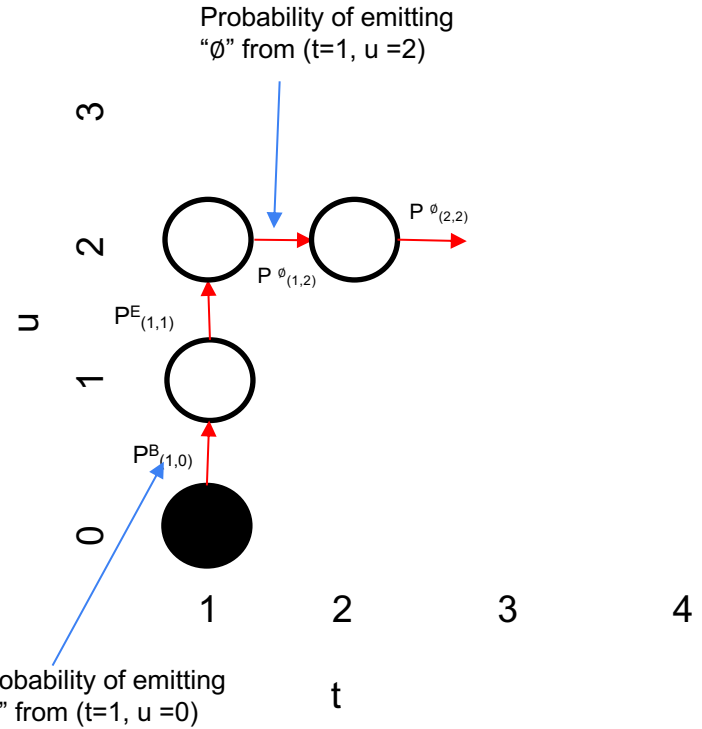


Graves et al., "Sequence transduction with recurrent neural networks"

An Alignment During Training

- Training utterance:
  "Input Audio" -> "BEE" (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- *BOS: Beginning of sentence*

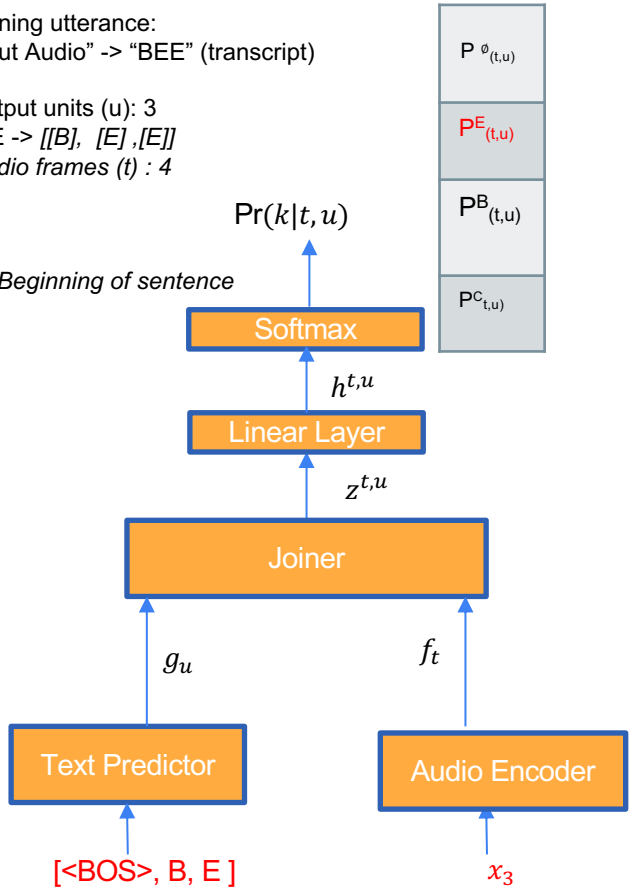


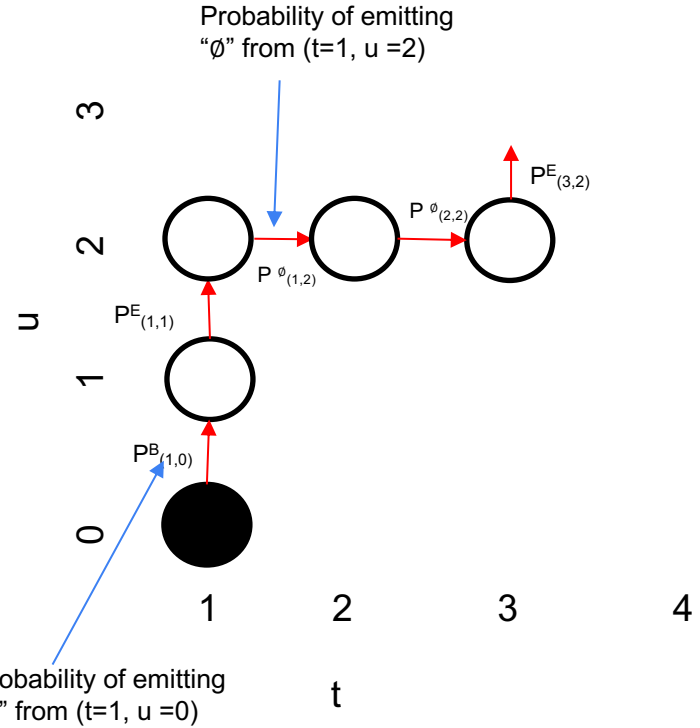


Graves et al., "Sequence transduction with recurrent neural networks"

An Alignment During Training

- Training utterance:
  "Input Audio" -> "BEE" (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- *BOS: Beginning of sentence*

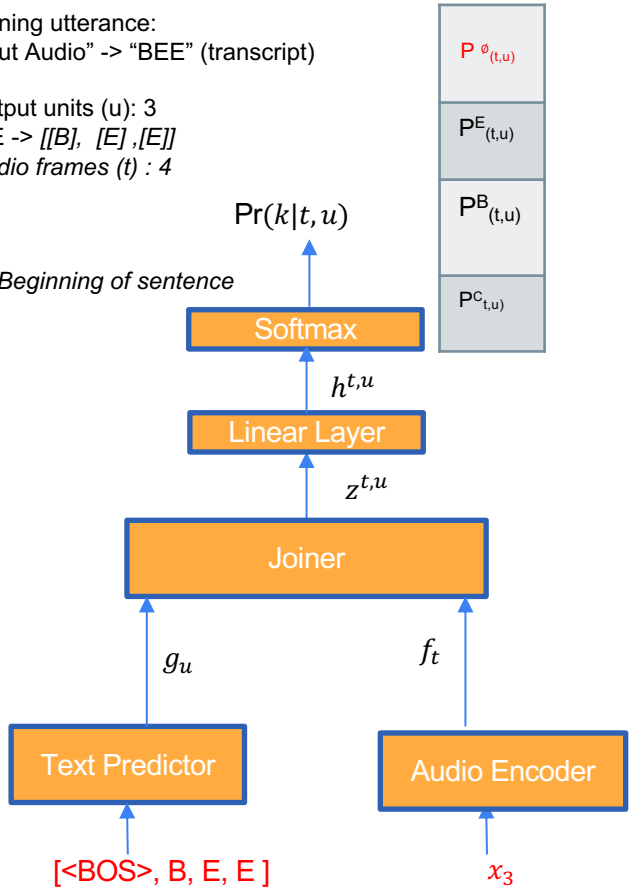


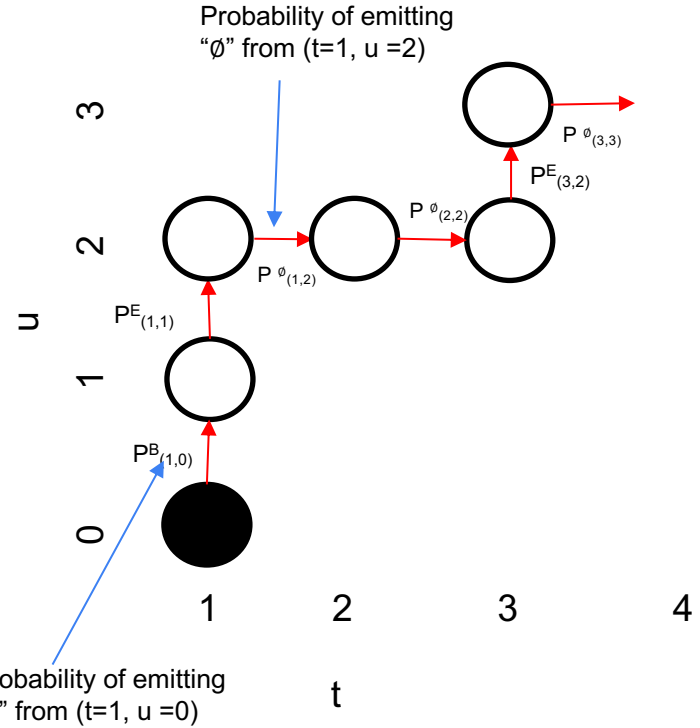


Graves et al., "Sequence transduction with recurrent neural networks"

An Alignment During Training

- Training utterance:
  "Input Audio" -> "BEE" (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- *BOS: Beginning of sentence*

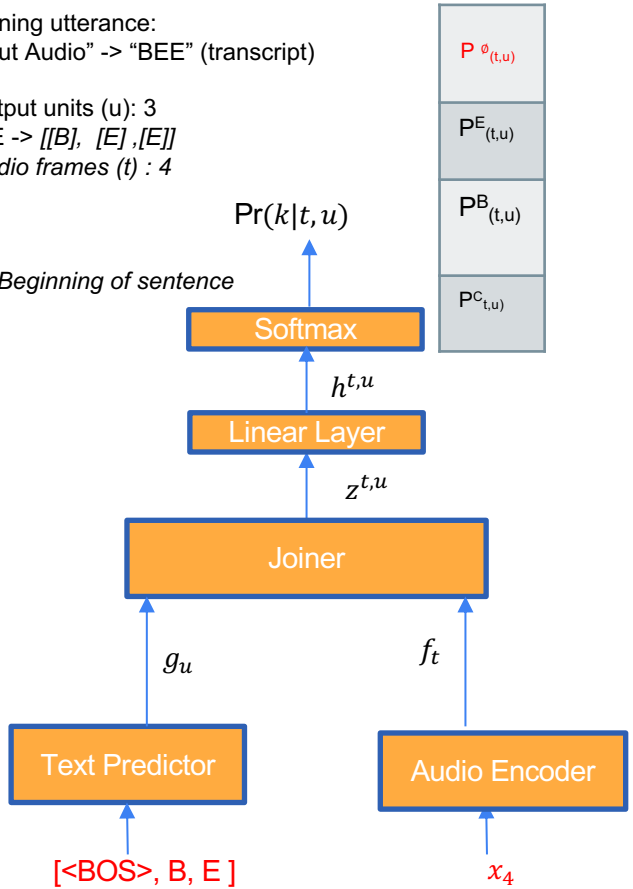


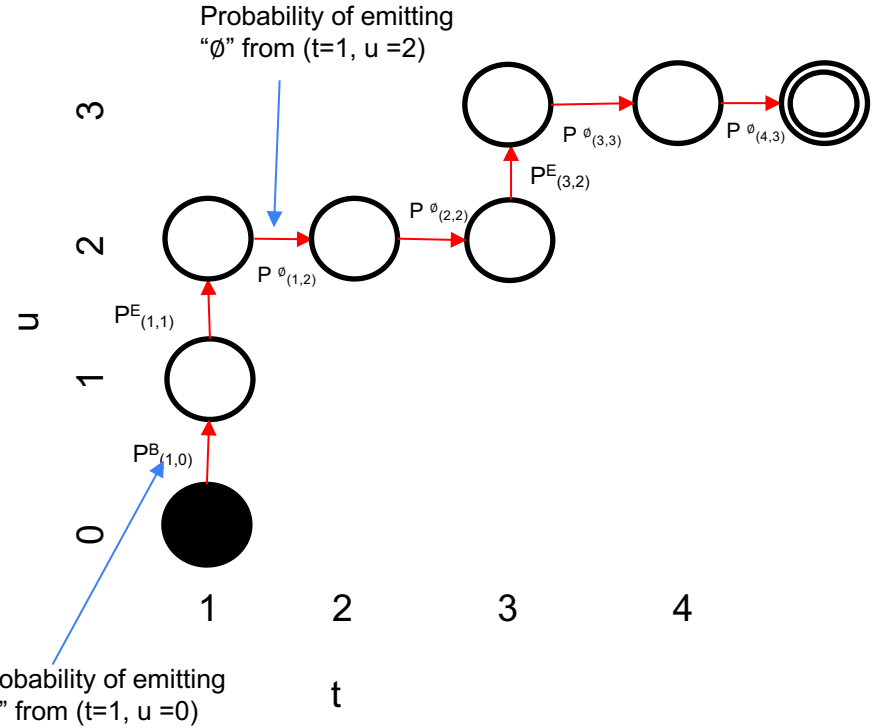


Graves et al., "Sequence transduction with recurrent neural networks"

An Alignment During Training

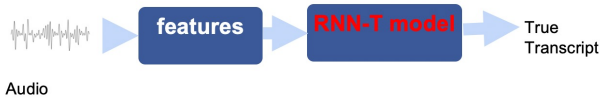
- Training utterance:
  "Input Audio" -> "BEE" (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- *BOS: Beginning of sentence*



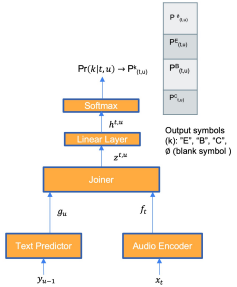


Graves et al., "Sequence transduction with recurrent neural networks"

RNN-T Lattice And Training



- Training utterance:
  "Input Audio" -> "BEE" (transcript)
- #output units (u): 3
  BEE -> *[[B], [E] ,[E]]*
- *#audio frames (t) : 4*
- We don't know alignment i.e. which portion of audio aligns to what output unit



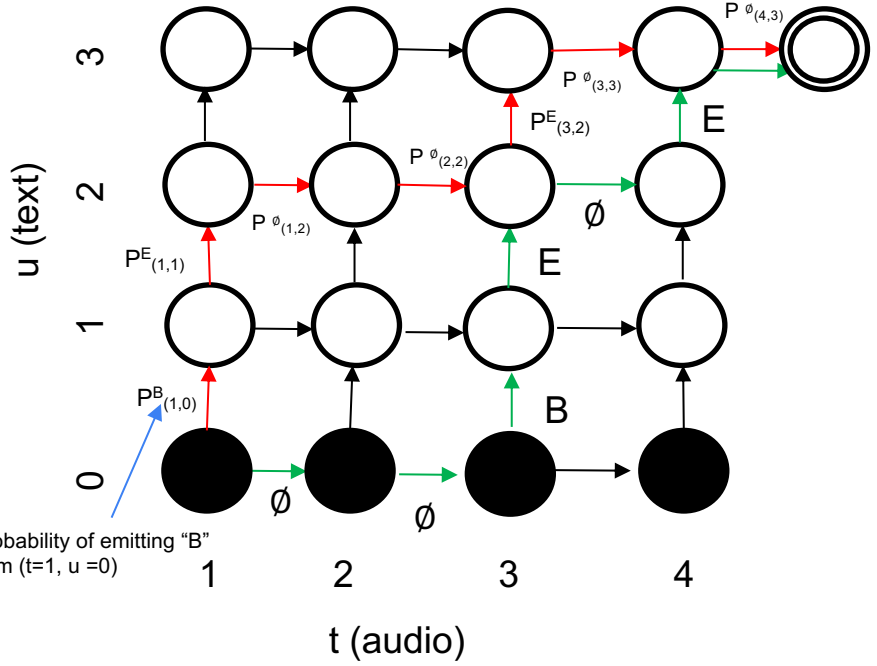
- Probability of alignment is multiplication of probabilities assigned along the path of alignment
- Training

$$P(BEE|X) = \sum_{alignment} P(alignment, BEE|X)$$

$$P(BEE|alignment, X) = 1$$

$$\sum_{alignment} P(BEE|alignment, X)P(alignment|X)$$

$$\sum_{alignment} P(alignment|X)$$



- Lattice contains all valid alignment paths(traversals). **During training, we change (optimize) neural network parameters to maximize sum of probabilities of all alignment paths**
- **Computation is done efficiently through dynamic programming (slide 54 to 58)**

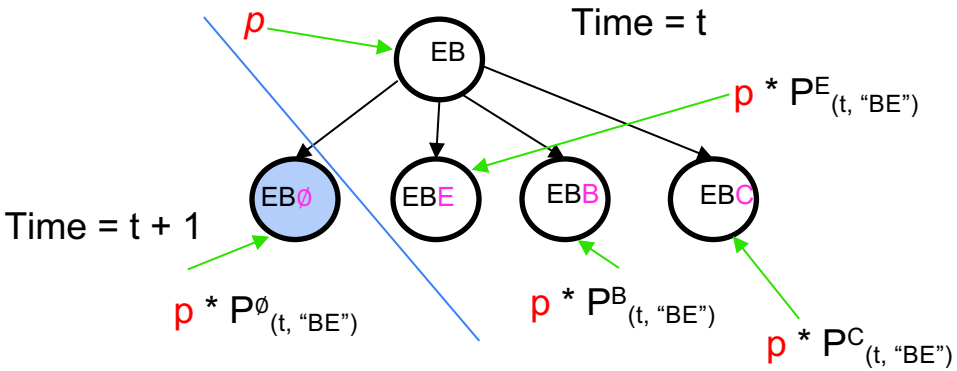
Graves et al., "Sequence transduction with recurrent neural networks"

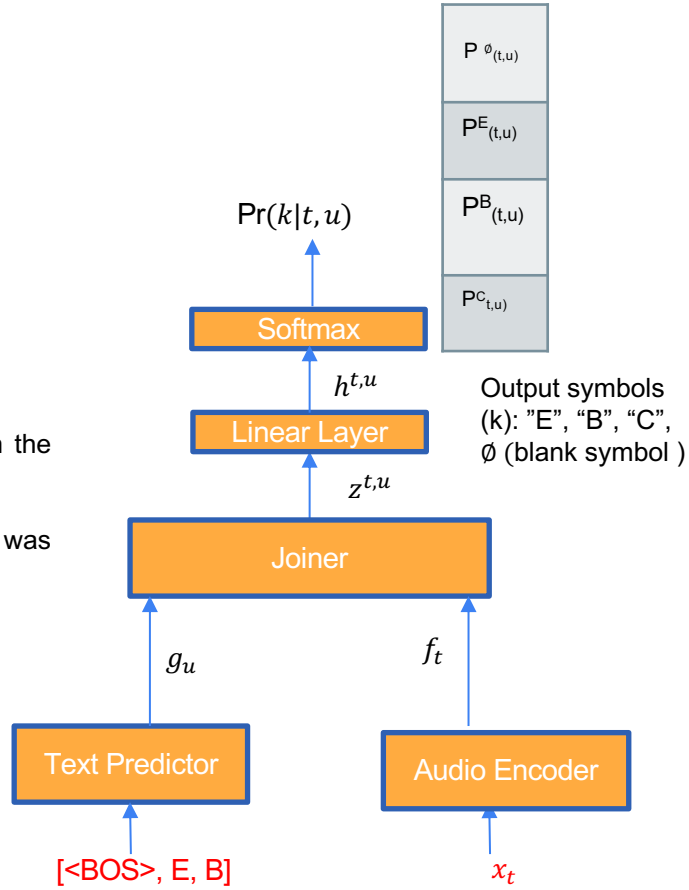
Inference: Find a Transcript given an audio and Trained RNN-T model

Goal: Find candidate transcript of a given audio during test time.
Challenge: There are infinitely many alignments that can be assigned to a test audio.

Operation: Extend Hypothesis

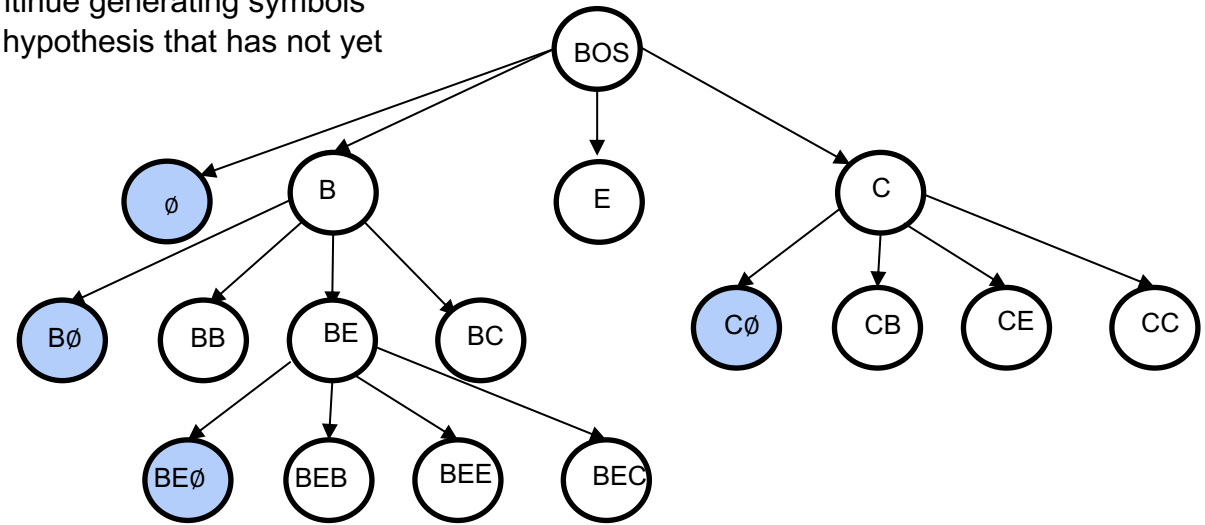
- Hypothesis is defined as a candidate output sequence(including ∅) during search
- Example Hypothesis: “EB” at time frame “t”
- Output symbols: ”E”, “B”, “C”, “∅”
- Extend Hypothesis: Append hypothesis with each of the output symbols (k) and ∅
- The ∅ extension goes to next time frame (t + 1) and non-blank extensions remain in the same time frame (t)
- Every extended hypothesis has lower probability compared to the hypothesis it was extended from

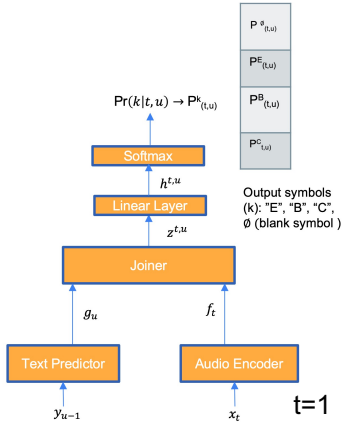


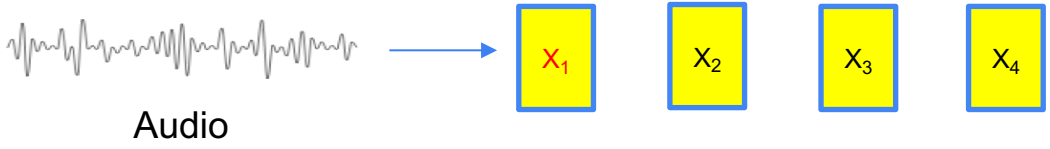


Subset Of Hypotheses At t = 1 During Decoding

- We can continue generating symbols from every hypothesis that has not yet emitted ∅

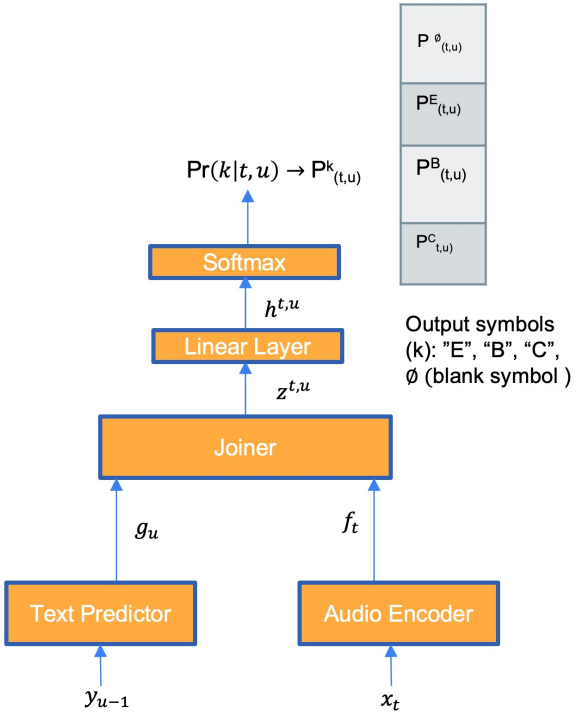






Greedy Decoding

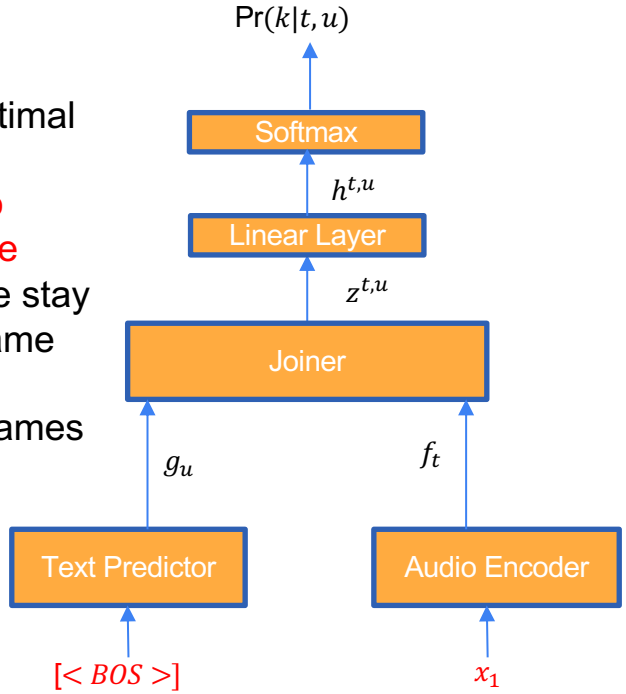
- Goal: Get a candidate final transcript by making local optimal choice at each distinct “t” and “u”



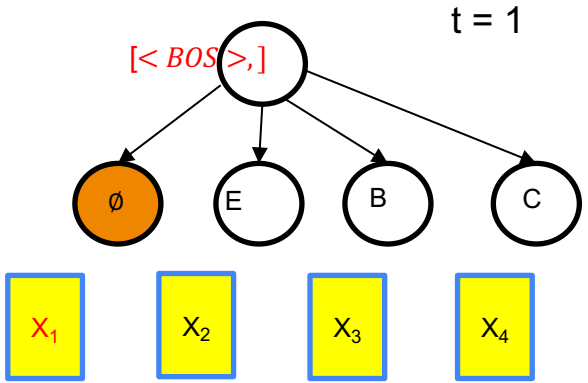
Greedy Decoding

- Lets make a local optimal choice: If highest probability is given to blank (∅), move to the next audio frame else stay in the same audio frame
- Do it until all audio frames are processed

k	probability
∅	0.4
E	0.2
B	0.3
C	0.1



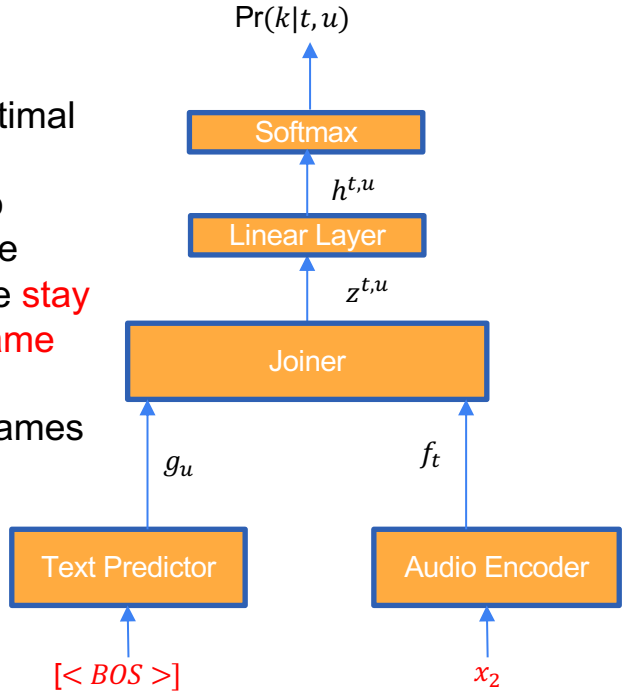
Audio



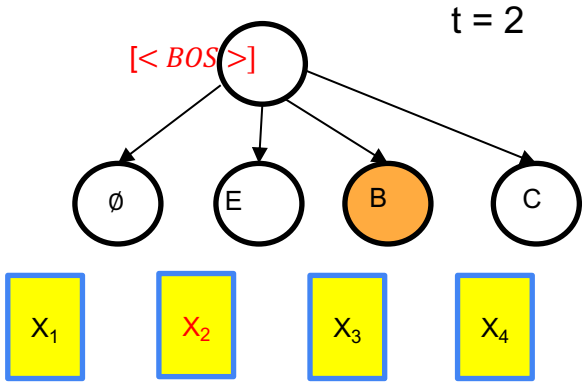
Greedy Decoding

- Lets make a local optimal choice: If highest probability is given to blank (∅), move to the next audio frame else stay in the same audio frame
- Do it until all audio frames are processed

k	probability
∅	0.04
E	0.04
B	0.3
C	0.01



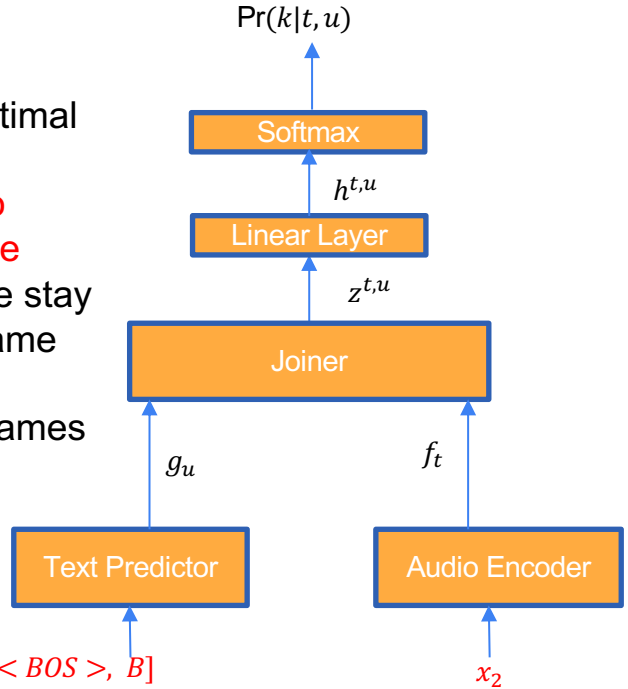
Audio

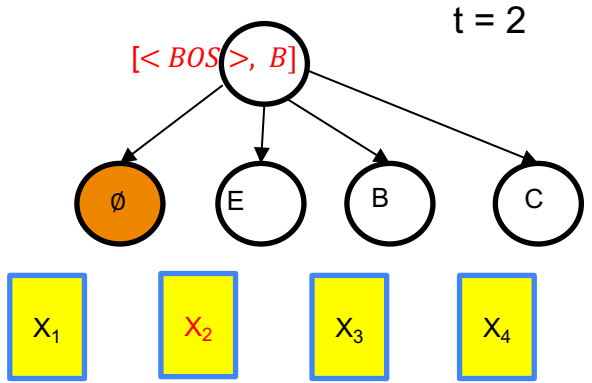


Greedy Decoding

- Lets make a local optimal choice: If highest probability is given to blank (∅), move to the next audio frame else stay in the same audio frame
- Do it until all audio frames are processed

k	probability
∅	0.4
E	0.1
B	0.2
C	0.3

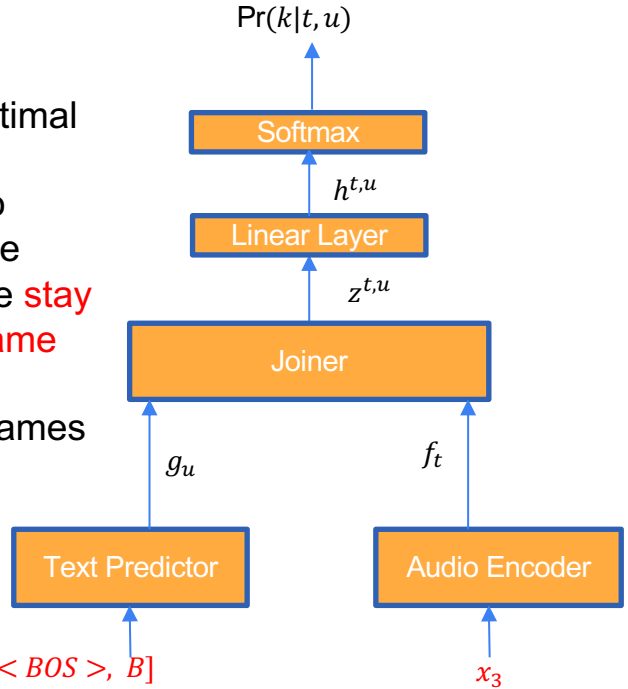


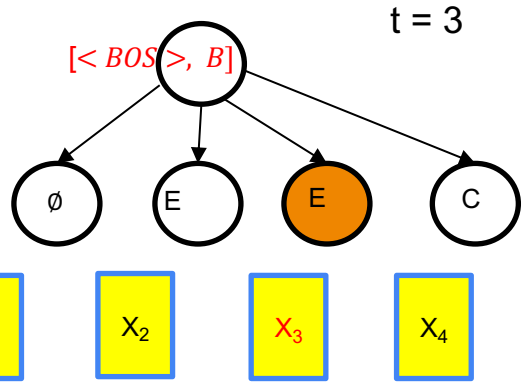


Greedy Decoding

- Lets make a local optimal choice: If highest probability is given to blank (∅), move to the next audio frame else stay in the same audio frame
- Do it until all audio frames are processed

k	probability
∅	0.3
E	0.35
B	0.3
C	0.15



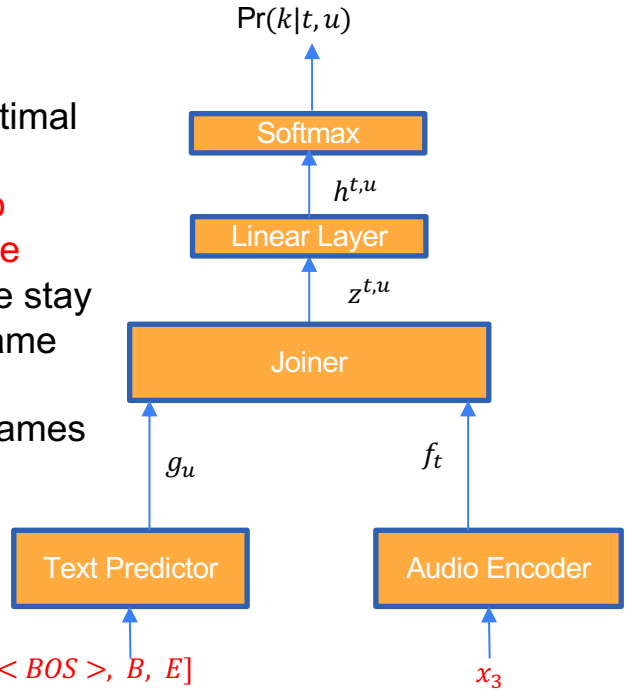


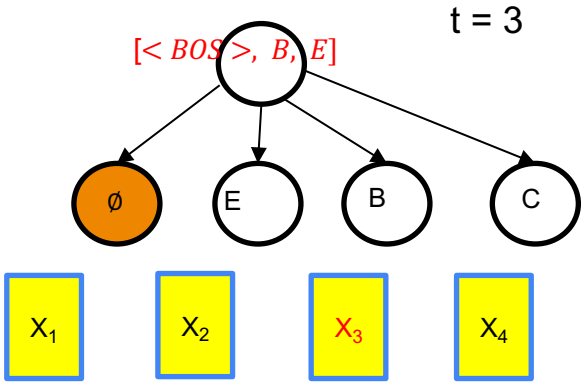
Audio

Greedy Decoding

- Lets make a local optimal choice: If highest probability is given to blank (∅), move to the next audio frame else stay in the same audio frame
- Do it until all audio frames are processed

k	probability
∅	0.34
E	0.26
B	0.25
C	0.25



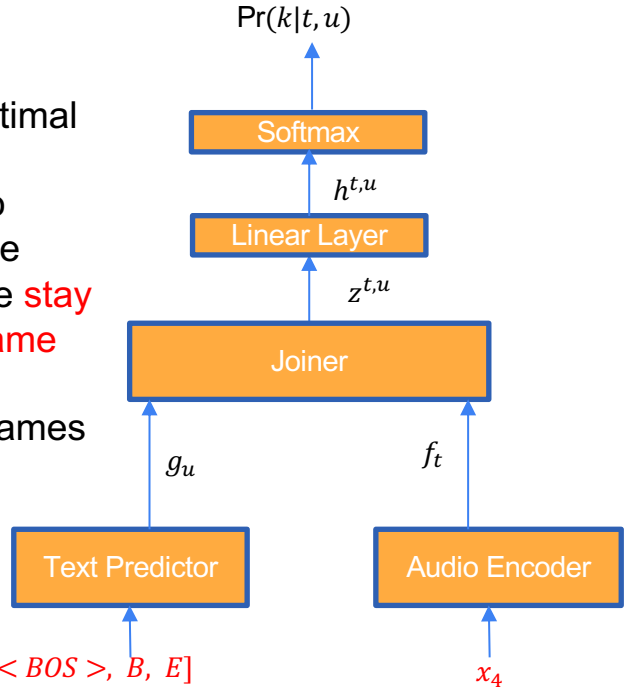


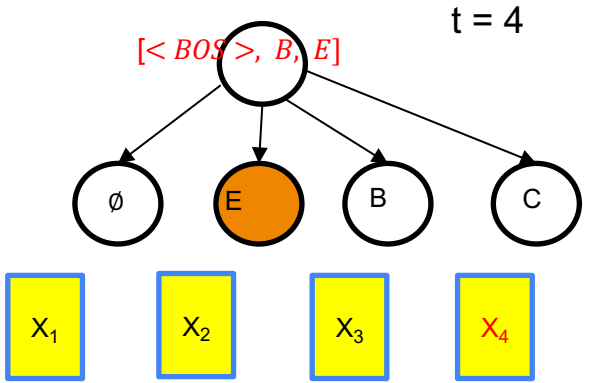
Audio

Greedy Decoding

- Lets make a local optimal choice: If highest probability is given to blank (∅), move to the next audio frame else stay in the same audio frame
- Do it until all audio frames are processed

k	probability
∅	0.3
E	0.4
B	0.28
C	0.02



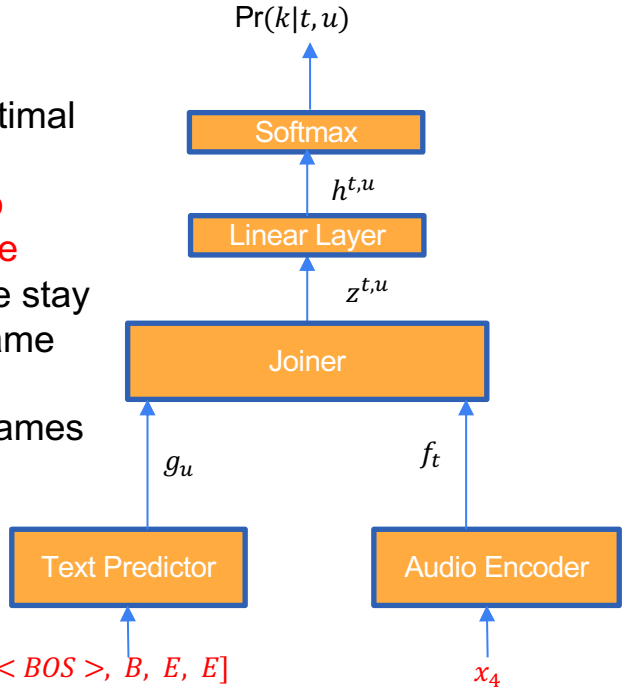


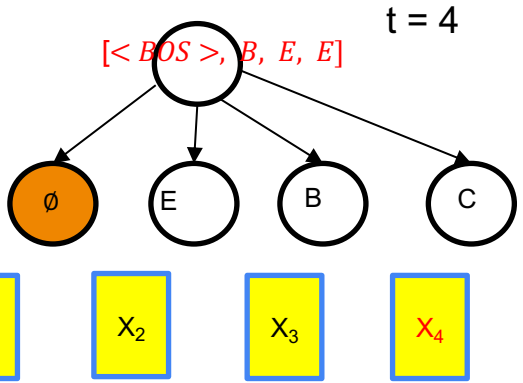
Audio

Greedy Decoding

- Lets make a local optimal choice: If highest probability is given to blank (∅), move to the next audio frame else stay in the same audio frame
- Do it until all audio frames are processed

k	probability
∅	0.4
E	0.25
B	0.3
C	0.05

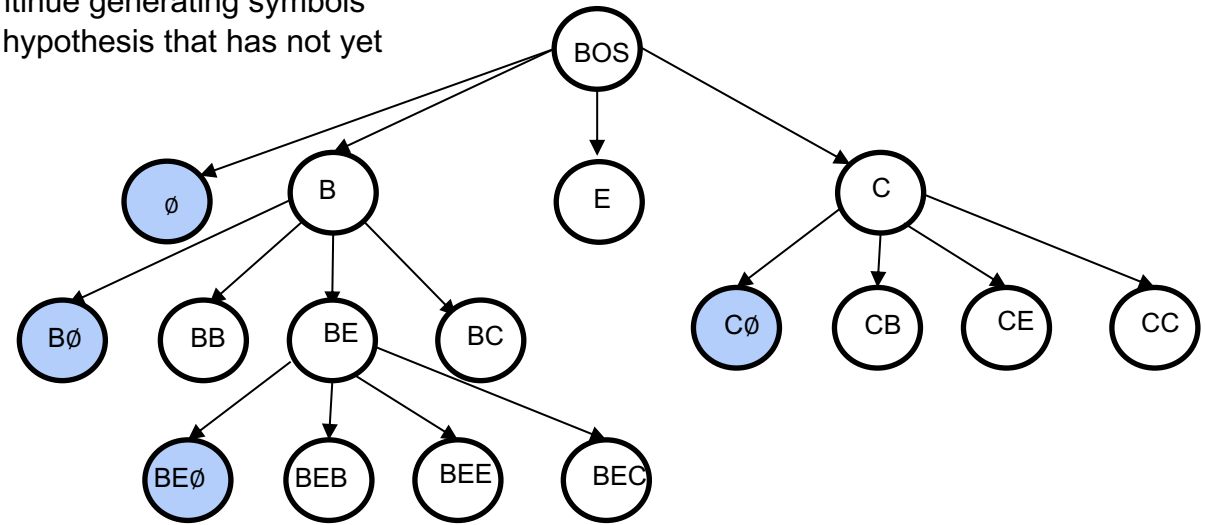


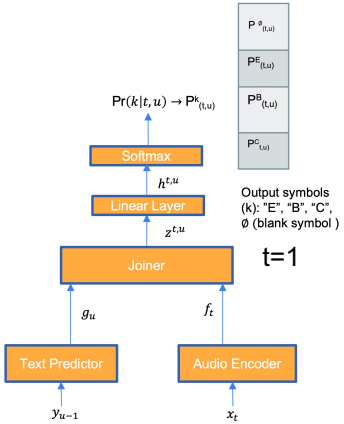


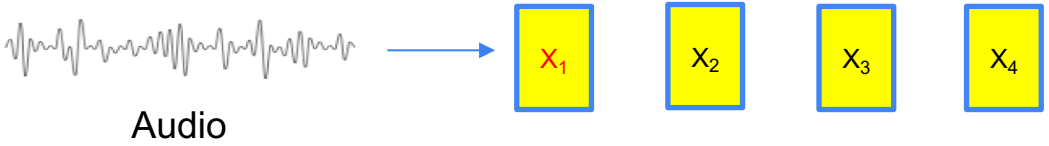
Audio

Subset Of Hypotheses At t = 1 During Decoding

- We can continue generating symbols from every hypothesis that has not yet emitted ∅







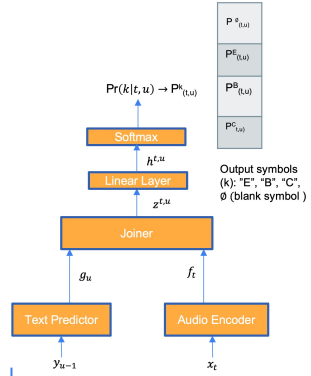
Beam Search

- We don’t want to stay in time frame “t” forever. But we do not know when to move to “t+1” as there are infinitely many candidates that can be explored in time frame “t”.

- Goal: We would like to obtain *n* candidate hypotheses at time frame “t+1” before exiting to decode at “t” which would be better than all possible future extensions at “t”.

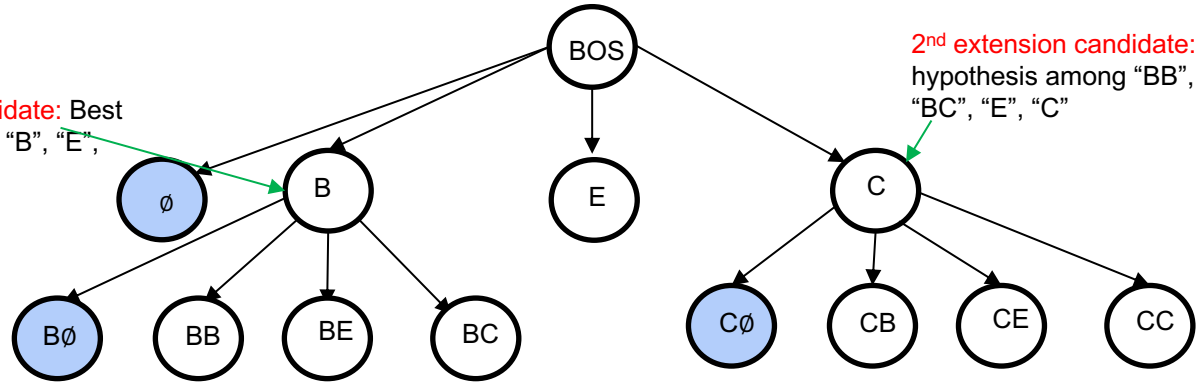
- *n* is hyper parameter

Beam Search: Expand the best hypothesis among candidate hypotheses



1st extension candidate: Best hypothesis among “B”, “E”, “C”

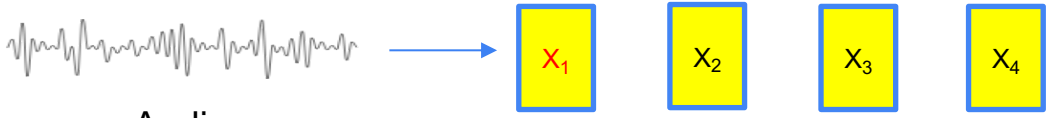
2nd extension candidate: Best hypothesis among “BB”, “BE”, “BC”, “E”, “C”



Bø
Cø
ø

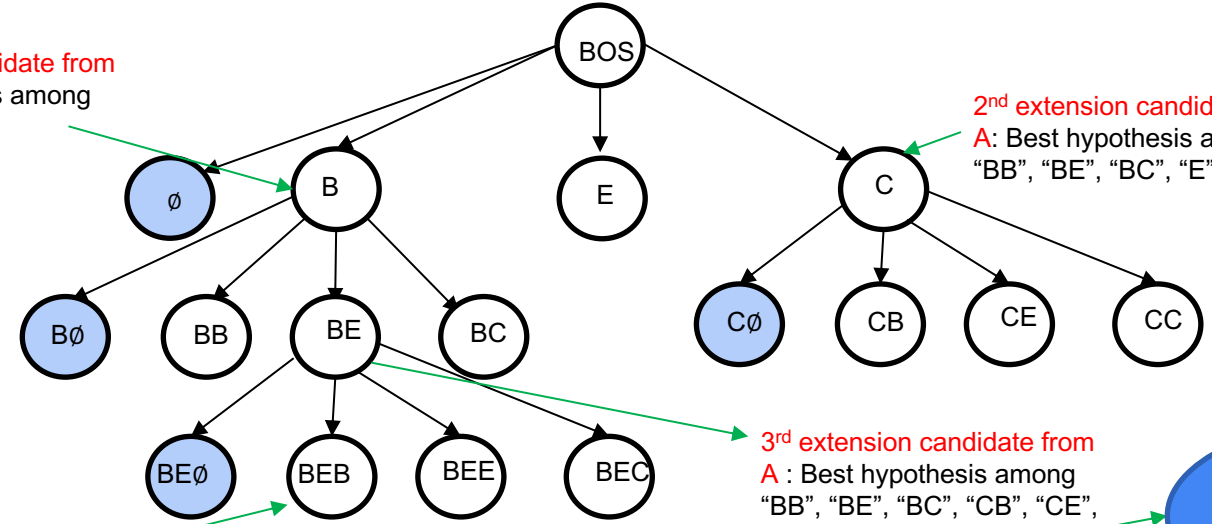
Hypothesis that has emitted ø and hence gone to “t+1”. Beam Size (n) “3”

Is beam size (n) number of extensions enough? No.



Beam Search: When to exit search at time frame t

- Set B: Hypotheses that has emitted ø from frame "t" and are now in "t+1"
- Set A: Hypotheses that has not yet emitted ø from frame "t" and so can continue emitting more symbols from "t"



1st extension candidate from A: Best hypothesis among "B", "E", "C"

2nd extension candidate from A: Best hypothesis among "BB", "BE", "BC", "E", "C"

3rd extension candidate from A : Best hypothesis among "BB", "BE", "BC", "CB", "CE", "CC"

4th extension candidate from A : Best hypothesis among "BB", "BEB", "BEE", "BEC", "BC", "CB", "CE", "CC"

nth best hypothesis from B

Bø
BEø
Cø
ø

Hypothesis that has emitted ø and hence gone to "t+1". Beam Size (n) "3"

Is the nth best hypothesis in B better than the best hypothesis in A

yes

Exit from t and move to t+ 1

no

Stay at t and expand more

Recap: Every extended hypothesis has lower probability compared to hypothesis it was extended from.

Thanks

- Many thanks to Rohit Prabhavalkar, Mark Tygert, Michael Picheny, Nayan Singhal, Kritika Singh, Xiaohui Zhang, Duc Le, Yuan Shangguan, Paco Guzmán, Yatharth Saraf and Mike Seltzer for brainstorming with ideas to improve content of this presentation.