Deep Learning Audio

Lecture 3

Pavel Severilov

Moscow Institute of Physics and Technology

2022

Outline

1. Automatic Speech Recognition (ASR)
2. Connectionist Temporal Classification (CTC)
3. Listen, Attend and Spell (LAS)

Outline

1. Automatic Speech Recognition (ASR)
2. Connectionist Temporal Classification (CTC)
3. Listen, Attend and Spell (LAS)

ASR: Task definition

Mapping: signal $x(t) \rightarrow$ text sequence s

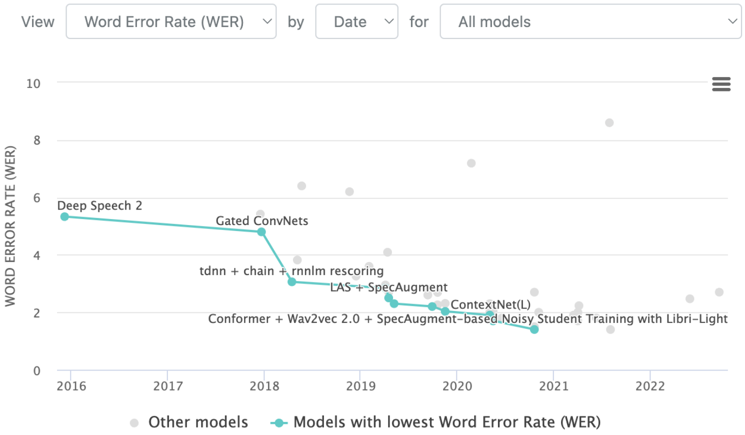


Figure: ASR progress for WER metric

ASR: Metrics

Word Error Rate

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

- S - number of substitutions,
- D - number of deletions,
- I - number of insertions,
- C - number of correct words,
- N - number of words in the reference ($N = S + D + C$).

```
True: quick brown    fox jumped over a lazy dog
Pred: quick brow  an fox jumped over   lazy dog
```

Character Error Rate: same as WER, but for characters (WER is more important)

ASR: Datasets: LibriSpeech

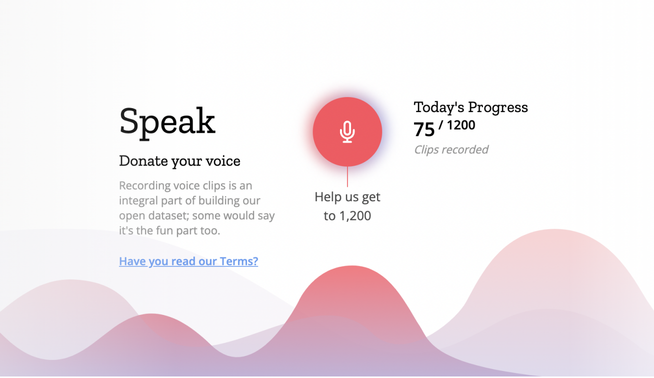
- 1,000 hours of audiobooks
- 10-20s audio, long sentences, complex language
- 'clean' (low-WER speakers) and 'other' (high-WER speakers) categories
- Human WER: test-clean: 5.83, test-other: 12.69
- Kaldi (2015) WER: test-clean: 8.01, test-other: 22.49
- Deep Speech 2 (2015) WER: test-clean: 5.15, test-other: 12.73

subset	hours	per-spk minutes	female spkrs	male spkrs	total spkrs
dev-clean	5.4	8	20	20	40
test-clean	5.4	8	20	20	40
dev-other	5.3	10	16	17	33
test-other	5.1	10	17	16	33
train-clean-100	100.6	25	125	126	251
train-clean-360	363.6	25	439	482	921
train-other-500	496.7	30	564	602	1166

Kaldi ASR by Daniel Povey

ASR: Datasets: Mozilla Common Voice

- Multiple languages
- Crowdsourced
- Simple language
- Short phrases
- Frequently updated and validated



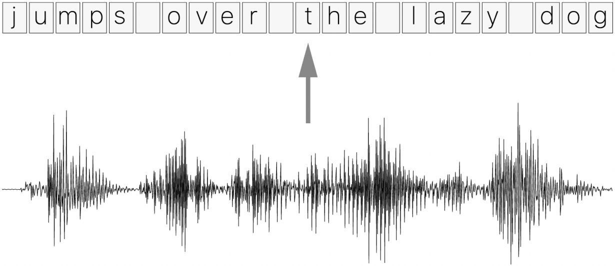
Mozilla Common Voice

Outline

1. Automatic Speech Recognition (ASR)
2. Connectionist Temporal Classification (CTC)
3. Listen, Attend and Spell (LAS)

CTC: motivation

- Variable length input
- Variable length output
- No alignment
- Want a differentiable loss function to compute $P(Y|X)$ and $\arg\max P(Y|X)$



Sequence Modeling With CTC blog post

CTC: idea

- Split input into frames
- Classify each frame into letters classes
- Merge consecutive letters

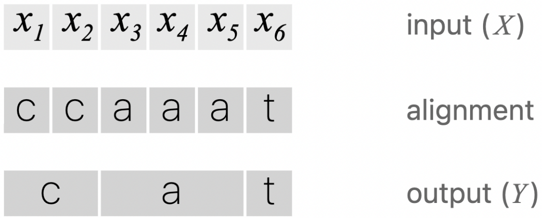


Figure: Example for [c, a ,t]

DeepSpeech 2

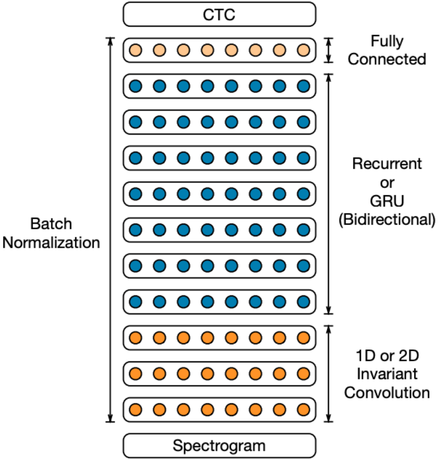


Figure: Deep Speech 2 architecture

Amodei, Dario Ananthanarayanan et al. (2015). Deep Speech 2: End-to-End Speech Recognition in English and Mandarin.

CTC: problems

Multiple consecutive letters (e.g.: hello), silence between words and letters.
Solution: add empty token ϵ

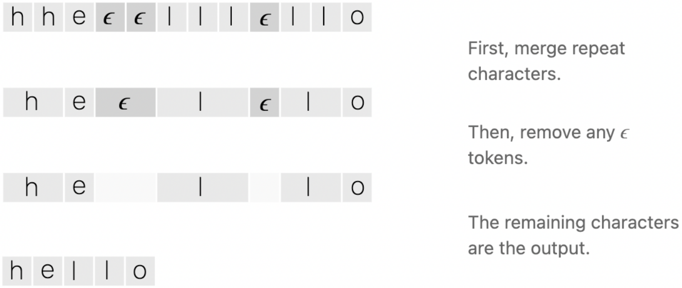


Figure: Example for [h, e, l, l, o]

CTC: Loss function

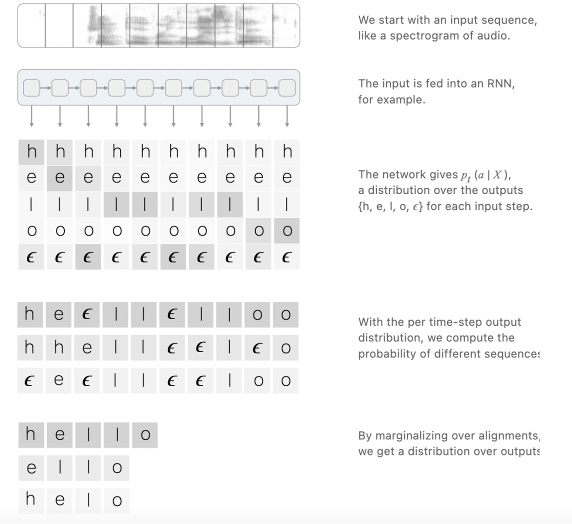


Figure: The CTC alignments give us a natural way to go from probabilities at each time-step to the probability of an output sequence.

CTC: Loss function

$$p(Y \mid X) \quad = \quad \sum_{A \in \mathcal{A}_{X,Y}} \quad \prod_{t=1}^{T} p_t(a_t \mid X)$$

The CTC conditional **probability**

marginalizes over the set of valid alignments

computing the **probability** for a single alignment step-by-step.

Figure: The CTC conditional probability

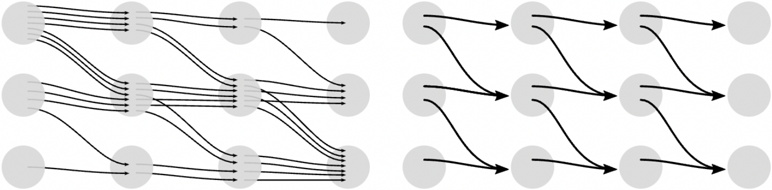


Figure: Summing over all alignments can be very expensive. Dynamic programming merges alignments, so it's much faster.

CTC: Computation

$$Z = [\epsilon,\ \overset{\text{a character}}{y_1},\ \epsilon,\ y_2,\ \ldots,\ \epsilon,\ y_U,\ \epsilon]$$

$\alpha_{s,t}$ is the CTC score of the subsequence $Z_{1:s}$ after t input steps.

Case 1

$$\alpha_{s,t} = (\alpha_{s-1,t-1} + \alpha_{s,t-1}) \cdot p_t(z_s \mid X)$$

The CTC probability of the two valid subsequences after $t-1$ input steps.

The probability of the current character at input step t.

Case 2

$$\alpha_{s,t} = (\alpha_{s-2,t-1} + \alpha_{s-1,t-1} + \alpha_{s,t-1}) \cdot p_t(z_s \mid X)$$

The CTC probability of the three valid subsequences after $t-1$ input steps.

The probability of the current character at input step t.

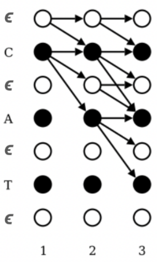


Figure: CTC computation with Dynamic programming

CTC: Properties

- Problem: Conditional Independence
- Example: "AAA". If predict 'A' as the first letter – suffix 'AA' should get much more probability than 'riple A'. If predict 't' first – the opposite.

Figure: Valid transcription could be "AAA" and "triple A".

- Advantage: Online – can be performed while the speaker is talking

CTC: Beam search

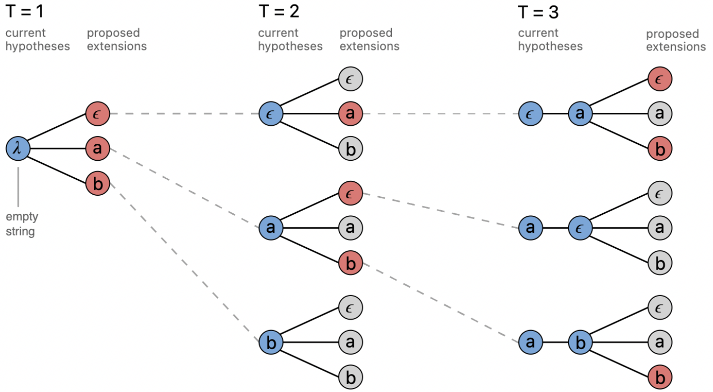


Figure: A standard beam search algorithm with an alphabet of $\{\epsilon, a, b\}$ and a beam size of 3.

Outline

1. Automatic Speech Recognition (ASR)
2. Connectionist Temporal Classification (CTC)
3. Listen, Attend and Spell (LAS)

LAS: Architecture

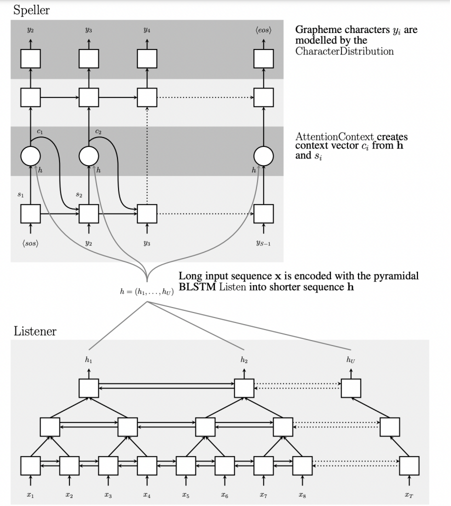


Figure: LAS model: **listener** – pyramidal BiLSTM encoding input sequence (spectrogram) into high level features h, **speller** – attention-based decoder generating the y characters from h. Train with cross entropy.

LAS: Beam search

- CTC computational cost T * beam size * **expand beam()**
- LAS computational cost T * beam size * **run decoder()**
- CTC usually uses 500 beam size, LAS – 3 beam size

WER: comparison

Method	**WER (test-clean)**	**WER (test-other)**
Human	5.83	12.69
Kaldi	8.01	22.49
Deep Speech 2	5.15	12.73
LAS	**3.2**	**9.8**