Language Independent End-to-End Architecture For Joint Language and Speech Recognition (2017)

Watanabe, S.; Hori, T.; Hershey, J.R.

Motivation / Goal

Recognize multiple languages at the same time

- Use a single model for 10 languages (EN, JP, CH, DE, ES, FR, IT, NL, PT, RU)
- Find out if transfer learning between languages work
- Two tasks: identify language AND recognize speech (simultaneously)
- End to end: Directly train sequence to sequence, no lexicon, phoneme pronounciation maps, or manual alignment

Problems

- How to input audio?

  $\rightarrow$ Spectral features of audio frames (e.g. in 10ms segments)

Problems

- How to input audio?

  $\rightarrow$ Spectral features of audio frames (e.g. in 10ms segments)

- How to output text?
  (a) word embeddings (word2vec) (would need fixed dictionary)
  (b) characters (one-hot)
    - Different char sets for languages (abc, äàąå, 漢字,   , ひらがな)
    - Just unify all character sets (5500 total)

Problems

- How to input audio?

  → Spectral features of audio frames (e.g. in 10ms segments)

- How to output text?
  (a) word embeddings (word2vec) (would need fixed dictionary)
  (b) characters (one-hot)
    - Different char sets for languages (abc, äàąå, 漢字, , ひらがな)
    - Just unify all character sets (5500 total)

- How to output language id?
  (a) separate one-hot output
  (b) as a special character: "`[EN]Hello`" or "`[CH]` 你好"

Related Work

Related Work

- *Multilingual Speech Recognition With A Single End-To-End Model* (Shubham Toshniwal)
  - separate output for language id
  - only on 9 indian languages, hard to compare
- *Hybrid CTC/Attention Architecture for End-to-End Speech Recognition* (Watanabe et al. 2017)
  - Same as this paper except only one language and more detailed

Model overview

Model overview

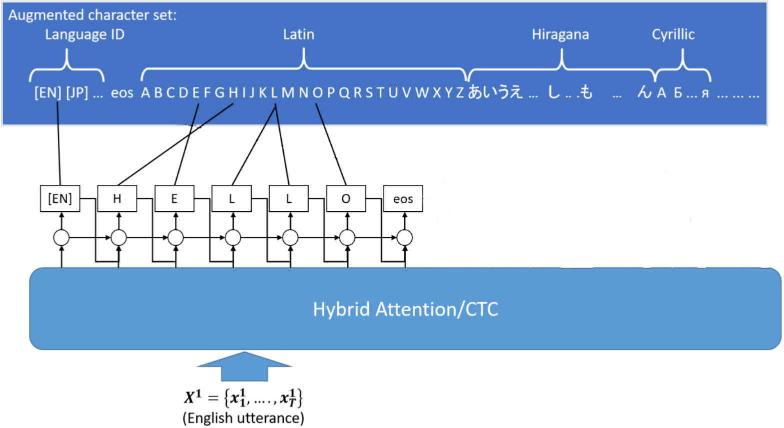


Figure 1: Model overview (from the paper)

Model overview

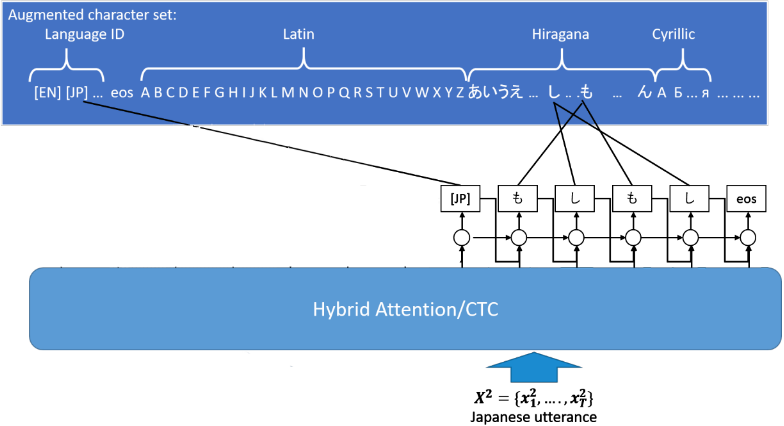


Figure 2: Model overview (from the paper)

Simple Model overview

1. Input: Basically a spectrogram as a 2D image
2. Encoder
   2.1 VGGNet Convolutional NN (first 6 layers)
   2.2 One bidirectional LSTM layer
3. Decoder (Attention + one directional LSTM)
   3.1 Soft Attention for each input frame to each output character
   3.2 LSTM Layer
   3.3 Fully connected layer (per time step)
4. Output
   - N characters from union of all languages (one-hot / softmax)

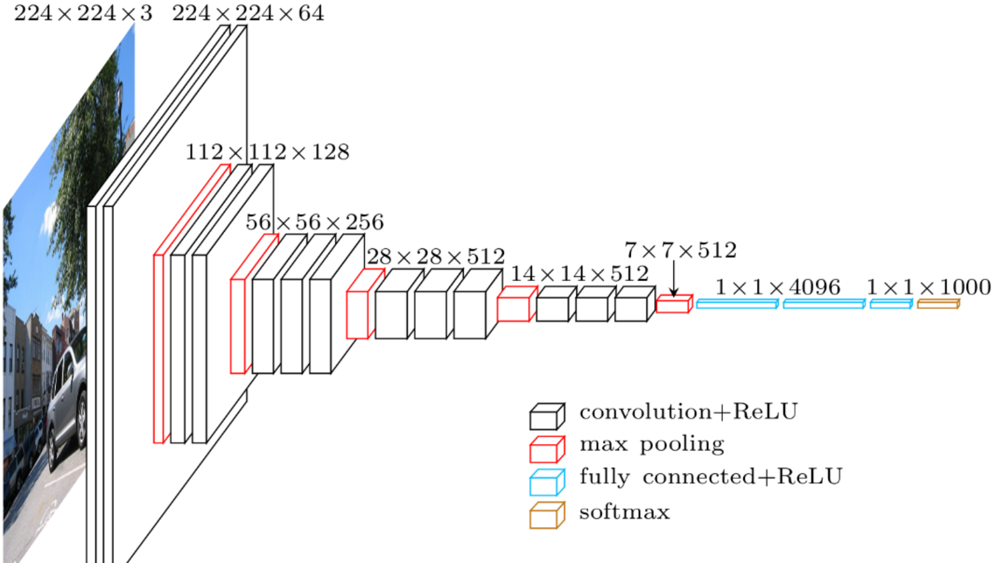
Input

(Ab)use of image processing pipeline - input formatted like a RGB image

x=time, y=feature index

- first channel: spectral features
- second channel: delta spectral features
- third channel: deltadelta spectral features

Encoder - VGG Net Architecture



Encoder - VGG Net Architecture - First six layers

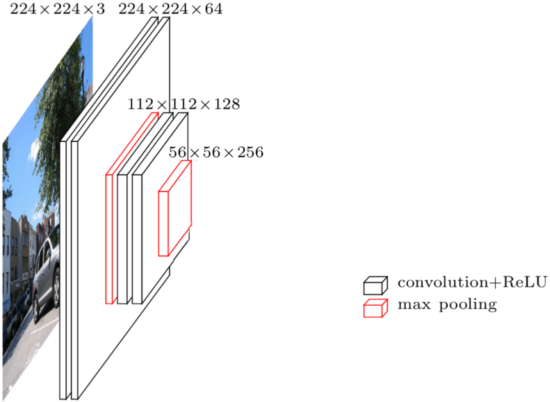


Figure 4: VGG Net - first 6 layers

Encoder - Bidirectional LSTM layer

320 cells for each direction $\rightarrow$ 640 outputs per time step ($\mathbf{h}_t$)

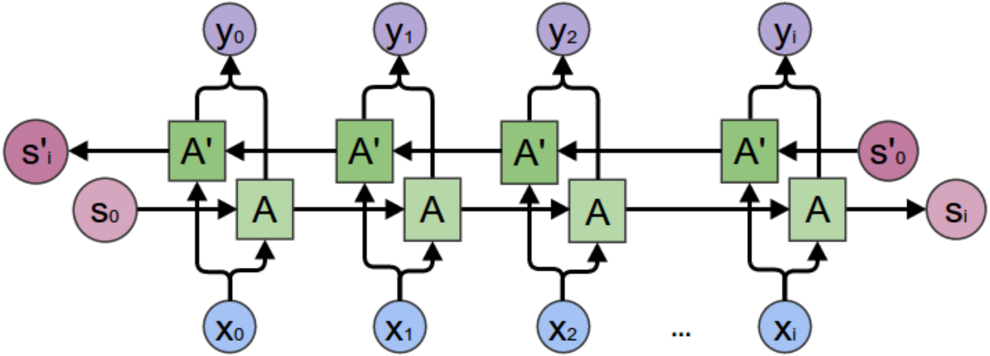


Figure 5: Bidirectional LSTM

http://colah.github.io/posts/2015-09-NN-Types-FP/

Decoder (Attention-based)

Input: $\mathbf{x}_1, \ldots, \mathbf{x}_t$

Output: $c_1, \ldots, c_l$

1. Encode whole sequence to $\mathbf{h}_1, \ldots, \mathbf{h}_t$ (via VGG+BLSTM)
2. Calculate soft attention weights a_{lt}, based on
   (a) $a_{(l-1)t}$ (attention on same input for previous output)
   (b) current encoded state $\mathbf{h}_t$
   (c) previous hidden state $\mathbf{q}_{l-1}$

Decoder (Attention-based)

Input: $\mathbf{x}_1, \ldots, \mathbf{x}_t$

Output: $c_1, \ldots, c_l$

1. Encode whole sequence to $\mathbf{h}_1, \ldots, \mathbf{h}_t$ (via VGG+BLSTM)
2. Calculate soft attention weights a_{lt}, based on
   (a) $a_{(l-1)t}$ (attention on same input for previous output)
   (b) current encoded state $\mathbf{h}_t$
   (c) previous hidden state $\mathbf{q}_{l-1}$
3. Sum encoded state with soft alignment: $\mathbf{r}_l = \sum_t a_{lt}\mathbf{h}_t$
4. Decoder = $\text{Softmax}(\text{FC}(\text{LSTM}(\mathbf{r}_l, \mathbf{q}_{l-1}, c_{l-1})))$

Problems with this simple model

- Pure temporal attention too flexible, allows nonsensical alignments
  - Intuition: In MT word order can change, in ASR it can not
- Languages must be implicitly modeled

Additions to the simple model

Problem 1: "Pure temporal attention too flexible"

Add a second, Parallel Decoder with CTC

1. Input (same as before)
2. Encoder (same as before)
3. Decoder

   fully connected softmax layer per time stemp (converts 640 outputs from BLSTM $\rightarrow$ N characters)
4. $\rightarrow$ One output character per input frame, using CTC Loss

CTC Crash Course

Problem: output sequence shorter than input sequence

- First, add blank character "-" to set. e.g. HELLO $\rightarrow \{H, E, L, O, -\}$

CTC Crash Course

Problem: output sequence shorter than input sequence

- First, add blank character "-" to set. e.g. HELLO $\rightarrow \{H, E, L, O, -\}$
- Inference: Remove duplicates: HHHH-EEEEEEEE-LL-LLL----OOOOOO $\rightarrow$ H-E-L-L-O $\rightarrow$ HELLO

CTC Crash Course

Problem: output sequence shorter than input sequence

- First, add blank character "-" to set. e.g. HELLO $\rightarrow \{H, E, L, O, -\}$
- Inference: Remove duplicates: HHHH-EEEEEEEE-LL-LLL----OOOOOO $\rightarrow$ H-E-L-L-O $\rightarrow$ HELLO
- Training: HELLO $\rightarrow$ H-E-L-L-O $\rightarrow$ all combinations of char duplications are ok

CTC Crash Course

Problem: output sequence shorter than input sequence

- First, add blank character "-" to set. e.g. HELLO $\rightarrow \{H, E, L, O, -\}$
- Inference: Remove duplicates: HHHH-EEEEEEEE-LL-LLL----OOOOOO $\rightarrow$ H-E-L-L-O $\rightarrow$ HELLO
- Training: HELLO $\rightarrow$ H-E-L-L-O $\rightarrow$ all combinations of char duplications are ok

$\rightarrow$ Enforces monotonic alignment

- Efficient computation using Viterbi / forward-backward algorithm
- Loss = negative log of GT probability

https://towardsdatascience.com/intuitively-understanding-connectionist-temporal-classification-3797e43a86c

Problem 2: “Languages must be implicitly modeled”

Add a RNN-LM

- Model distribution of character sequences in languages (ignores input speech)
- Trained seperately

Combine both decoders + RNN-LM

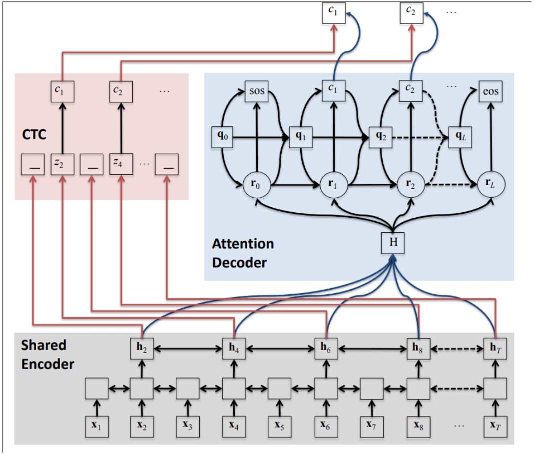


Figure 6: Hybrid CTC/attention-based end-to-end architecture (RNN-LM not shown)

Final loss function

$$\mathcal{L}_{\mathsf{MTL}} = \lambda \log p_{\mathsf{ctc}}(C|X) + (1-\lambda) \log p_{\mathsf{att}}(C|X) + \gamma \log p_{\mathsf{rnnlm}}(C)$$

$\lambda = 0.5$, $\gamma = 0.1$

Training / Inference

- AdaDelta optimization, 15 epochs
- Inference via beam search on attention output weighted by loss function

Results

Results

	Language-dependent 4BLSTM	**7lang** 4BLSTM	**7lang** CNN-7BLSTM	**7lang** CNN-7BLSTM RNN-LM	**10lang** CNN-7BLSTM RNN-LM
Avg. 7 langs	22.7	20.3	18.9	18.3	**16.6**

Figure 7: Character Error Rates (abbrev.)

- VGG-CNN improves it (by 7%)
- RNN-LM improves it (by 3%)
- Adding data in other languages improves it (by 9%)

Language Confusion Matrix

		CH	EN	JP	DE	ES	FR	IT	NL	RU	PT
CH	train_dev	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	dev	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
EN	test_eval92	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	test_dev93	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
JP	eval1_jpn	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	eval2_jpn	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	eval3_jpn	0.0	0.0	99.9	0.0	0.0	0.0	0.1	0.0	0.0	0.0
DE	et_de	0.0	0.0	0.0	99.7	0.0	0.0	0.0	0.3	0.0	0.0
	dt_de	0.0	0.0	0.0	99.7	0.0	0.0	0.0	0.3	0.0	0.0
ES	dt_es	0.0	0.0	0.0	0.0	67.9	0.0	31.9	0.0	0.0	0.2
	et_es	0.0	0.0	0.0	0.1	91.1	0.0	8.4	0.1	0.0	0.2
FR	dt_fr	0.0	0.0	0.0	0.1	0.0	99.4	0.0	0.2	0.0	0.3
	et_fr	0.0	0.0	0.0	0.1	0.0	99.5	0.0	0.1	0.0	0.3
IT	dt_it	0.0	0.0	0.0	0.0	0.3	0.4	99.1	0.0	0.0	0.3
	et_it	0.0	0.0	0.0	0.0	0.4	0.4	98.3	0.2	0.1	0.7
NL	dt_nl	0.0	0.0	0.0	1.3	0.0	0.1	0.1	97.2	0.0	1.3
	et_nl	0.0	0.0	0.0	1.0	0.0	0.2	0.2	97.6	0.0	0.9
RU	dt_ru	0.2	0.0	0.0	0.0	0.2	0.6	0.5	0.0	97.9	0.8
	et_ru	0.0	0.0	0.0	0.2	0.2	0.3	4.3	0.0	94.7	0.3
PT	dt_pt	0.0	0.0	0.0	0.3	0.3	2.6	1.7	3.4	0.6	91.2
	et_pt	0.0	0.3	0.0	0.3	0.0	0.0	3.9	3.6	0.3	91.5

Figure 8: Language identification (LID) accuracies/error rates (%). The diagonal elements correspond to the LID accuracies while the offdiagonal elements correspond to the LID error rates

Potential problems / future work?

- Only fed with a single language utterance at a time
  - maybe we want to allow switching? (append utterances from different languages)

Potential problems / future work?

- Only fed with a single language utterance at a time
  - maybe we want to allow switching? (append utterances from different languages)
- Uniform random parameter initialization with [-0.1, 0.1] seems statistically unsound? (use Xavier / Hu)

Potential problems / future work?

- Only fed with a single language utterance at a time
  - maybe we want to allow switching? (append utterances from different languages)
- Uniform random parameter initialization with [-0.1, 0.1] seems statistically unsound? (use Xavier / Hu)
- Input feature convolution is weird
  - *[...] we used 40-dimensional filterbank features with 3-dimensional pitch features*
  - redundancy (delta, deltadelta)

Potential problems / future work?

- Only fed with a single language utterance at a time
  - maybe we want to allow switching? (append utterances from different languages)
- Uniform random parameter initialization with [-0.1, 0.1] seems statistically unsound? (use Xavier / Hu)
- Input feature convolution is weird
  - *[...] we used 40-dimensional filterbank features with 3-dimensional pitch features*
  - redundancy (delta, deltadelta)
- Unbalanced language sets (500h CH, 2.9h PR)
- Same latin characters are used for multiple languages, while others (RU, CH, JP) get their own character set
  - Try transliterating them to Latin?

Future Work?

- Does not work in realtime (without complete input utterance)
  - Bidirectional LSTM in encoder
    - Could try one directional, but Language ID would completely break
    - aggregate limited number of future frames (e.g. add 500ms latency between input and output)
  - Does CTC work in real time?
  - Attention does not work in realtime

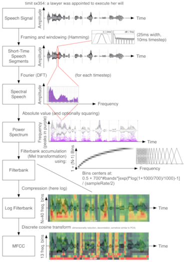
Thank you for your *attention*

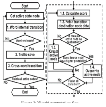
Full Results Table

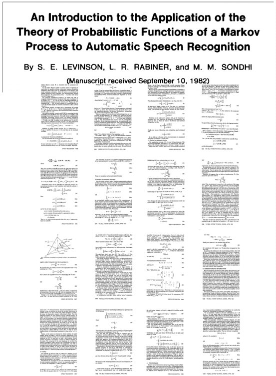
			Language-dependent 4BLSTM	7lang 4BLSTM	7lang CNN-7BLSTM	7lang CNN-7BLSTM RNN-LM	10lang CNN-7BLSTM RNN-LM
HKUST	CH	train_dev	40.1	43.9	40.5	40.2	32.0
		dev	40.4	43.6	40.5	40.0	31.0
WSJ	EN	dev93	9.4	9.6	7.7	7.0	9.7
		eval92	7.4	7.3	5.6	5.1	7.4
CSJ	JP	eval1	13.5	14.3	12.4	11.9	10.2
		eval2	10.8	10.8	9.0	8.5	7.2
		eval3	23.2	24.9	22.0	21.4	8.7
Voxforge	DE	dev	6.6	7.4	5.7	5.4	7.3
		eval	5.2	7.4	5.8	5.5	7.3
	ES	dev	50.9	28.1	31.9	31.5	25.8
		eval	50.8	29.6	34.7	34.4	26.7
	FR	dev	27.7	25.0	22.0	21.0	24.1
		eval	26.5	23.5	21.2	20.3	23.2
	IT	dev	14.3	14.3	11.8	11.1	13.8
		eval	14.3	14.4	12.0	11.2	14.1
	NL	dev	27.0				23.2
		eval	25.5				22.4
	RU	dev	47.8				45.0
		eval	49.4				43.2
	PT	dev	56.9				35.5
		eval	52.2				31.9
Avg.	7 langs		22.7	20.3	18.9	18.3	**16.6**
Avg.	10 langs		27.4				**21.4**

WHO WOULD WIN?

decades of research on Feature extraction, Dynamic time warping, HMMs, Language modeling







An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition

By S. E. LEVINSON, L. R. RABINER, and M. M. SONDHI

(Manuscript received September 10, 1982)

one deepy boi

Solving universal speech recognition

By Random Author, Big Company, Random other Guy

we literally just throw an LSTM at it.