End-to-End Speech Recognition by Following my Research History

Shinji Watanabe
Center for Language and Speech Processing
Johns Hopkins University

Language Technologies Institute
Carnegie Mellon University (Jan. 2021)

@11-785 Introduction to Deep Learning
About this presentation

• This is based on my personal experience
• I re-order or re-structure several existing materials based on a chronological order
• I’m assuming people have some end-to-end neural network knowledge
Timeline

Shinji’s personal experience for end-to-end speech processing

- **2015**
  - First impression
  - No more conditional independence assumption
  - DNN tool blossom

- **2016**
  - Initial implementation
  - CTC/attention hybrid
  - Japanese e2e -> multilingual.

- **2017**
  - Open source
  - share the knowhow
  - Kaldi-style
  - Jelinek workshop

- **2018**
  - ASR+X
  - TTS
  - Speech translation

- **2019-**
  - Improvement
  - Transformer
  - Open source acceleration
Shinji’s personal experience for end-to-end speech processing

- First impression
  - No more conditional independence assumption
  - DNN tool blossom
Noisy channel model (1970s-)
Noisy channel model (1970s-)

- Automatic Speech Recognition: Mapping physical signal sequence to linguistic symbol sequence

\[ X = \{ x_l \in \mathbb{Z} | l = 1, \ldots, L \} \]

\[ L = 43263 \]

\[ W = \{ w_n \in \mathcal{V} | n = 1, \ldots, N \} \]

\[ N = 3 \]

\[ X = \{ x_t \in \mathbb{C}^D | t = 1, \ldots, T \} \]

\[ T = 268 \]

“That’s another story”
Noisy channel model (1970s-)

$$\text{arg max}_W p(W|X)$$

$X$: Speech sequence

$W$: Text sequence
Noisy channel model (1970s-)

\[ \arg \max_W p(W|X) = \arg \max_W p(X|W)p(W) \]
\[ \approx \arg \max_{W,L} p(X|L, W)p(L|W)p(W) \]

(L: Phoneme sequence)

- **Speech recognition**
  - \( p(X|L) \): Acoustic model (Hidden Markov model)
  - \( p(L|W) \): Lexicon
  - \( p(W) \): Language model (n-gram)
arg max \( p(W|X) \) = arg max \( p(X|W)p(W) \)
\[
\approx \arg \max_{W,L} p(X|L,W)p(L|W)p(W)
\]

• **Speech recognition**
  - \( p(X|L) \): Acoustic model (Hidden Markov model)
  - \( p(L|W) \): Lexicon
  - \( p(W) \): Language model (n-gram)
Noisy channel model (1970s-)

\[
\arg \max_W p(W|X) = \arg \max_W p(X|W)p(W)
\]

- **Machine translation**
  - \( p(X|W) \): Translation model
  - \( p(W) \): Language model
Noisy channel model (1970s-)

\[
\arg \max_W p(W|X) = \arg \max_W p(X|W)p(W) \\
\approx \arg \max_{W,L} p(X|L,W)p(L|W)p(W)
\]

- **Speech recognition**
  - \( p(X|L) \): Acoustic model (Hidden Markov model)
  - \( p(L|W) \): Lexicon
  - \( p(W) \): Language model (n-gram)

- Continued 40 years
arg \max_w p(W|X) = \arg \max_w p(X|W)p(W) 
\approx \arg \max_{W,L} p(X|L,W)p(L|W)p(W)

- **Speech recognition**
  - \( p(X|L) \): Acoustic model
  - \( p(L|W) \): Lexicon
  - \( p(W) \): Language model

- **Big barrier**:
  - noisy channel model
  - HMM
  - n-gram
  - etc.

- **Continued 40 years**
However,
“End-to-End” Processing
Using Sequence to Sequence

- Directly model $p(W|X)$ with a **single neural network**
  - **Integrate** acoustic $p(X|L)$, lexicon $p(L|W)$, and language $p(W)$ models
- Great success in neural machine translation
End-to-end ASR (1)

Connectionist temporal classification (CTC)


- Use bidirectional RNNs to predict frame-based labels including blanks
- Find alignments between $X$ and $Y$ using dynamic programming

CTC

Forward-Backward or Viterbi algorithm

Stacked BLSTM
End-to-end ASR (2)

Attention-based encoder decoder [Chorowski+ 2014, Chan+ 2015]

- Combine acoustic and language models in a single architecture
  - Encoder: DNN part of acoustic model
  - Decoder: language model
  - Attention: HMM part of acoustic model
First impression in -2015

• Attention based encoder decoder

\[
\arg \max_W p(W | X) = \arg \max_W \prod_j p(w_j | w_{<j}, X)
\]

• No conditional independence assumption unlike HMM/CTC
  – More precise seq-to-seq model
  – This is what I have been struggling for 15 years!

• Attention mechanism allows too flexible alignments
  – Hard to train the model from scratch
Timeline

Shinji’s personal experience for end-to-end speech processing

2016

Initial implementation

- CTC/attention hybrid
- Japanese e2e -> multilingual.
Initial implementation in 2016

- Suyoun Kim (CMU), Takaaki Hori, John Hershey, and I started an E2E project at MERL with some interns
- First, we implemented both
  - CTC
  - Attention-based encoder/decoder
- We found some pros. and cons.
Connectionist temporal classification (CTC)


- Use bidirectional RNNs to predict frame-based labels including blanks
- Find alignments between $X$ and $Y$ using dynamic programming
- Relying on conditional independence assumptions (similar to HMM)
- Output sequence is not well modeled (no language model)
Attention-based encoder decoder [Chorowski+ 2014, Chan+ 2015]

• Combine acoustic and language models in a single architecture
  – Encoder: DNN part of acoustic model
  – Decoder: language model
  – Attention: HMM part of acoustic model

• No conditional independence assumption unlike HMM/CTC
  – More precise seq-to-seq model

• Attention mechanism allows too flexible alignments
  – Hard to train the model from scratch
Input/output alignment by temporal attention

- Unlike CTC, attention model does not preserve order of inputs
- Our desired alignment in ASR task is **monotonic**
- Not regularized alignment makes the model **hard to learn** from scratch

**Example of monotonic alignment**

**Example of distorted alignment**
Timeline

Shinji’s personal experience for end-to-end speech processing

2016

Initial implementation

- CTC/attention hybrid
- Japanese e2e -> multilingual.
How to solve this unstable attention issues

It was **too unstable** to move to the next step...

• We had a lot of ideas but those were pending due to that
• Probably we should try to use **both benefits of CTC and attention**

How to combine both?

• One possible solution: RNN transducer
• Try to find another solution
• Finally came up with a simple idea (or we decided to use this simple idea)
  ➞ **Hybrid CTC/attention**
Hybrid CTC/attention network [Kim+’17]

Multitask learning: \( \mathcal{L}_{MTL} = \lambda \mathcal{L}_{CTC} + (1 - \lambda) \mathcal{L}_{Attention} \)

\( \lambda \): CTC weight

CTC guides attention alignment to be monotonic
More robust input/output alignment of attention

- Alignment of one selected utterance from CHiME4 task

**Attention Model**

Epoch 1          Epoch 3          Epoch 5          Epoch 7          Epoch 9

**Corrupted!**

**Our joint CTC/attention model**

Epoch 1          Epoch 3          Epoch 5          Epoch 7          Epoch 9

**Monotonic!**

**Faster convergence**
Joint CTC/attention decoding [Hori+’17]

Use CTC for decoding together with the attention decoder

CTC explicitly eliminates non-monotonic alignment
Experimental Results

Character Error Rate (%) in **Mandarin** Chinese Telephone Conversational (HKUST, 167 hours)

<table>
<thead>
<tr>
<th>Models</th>
<th>Dev.</th>
<th>Eval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attention model (baseline)</td>
<td>40.3</td>
<td>37.8</td>
</tr>
<tr>
<td>CTC-attention learning (MTL)</td>
<td>38.7</td>
<td>36.6</td>
</tr>
<tr>
<td>+ Joint decoding</td>
<td>35.5</td>
<td>33.9</td>
</tr>
</tbody>
</table>

Character Error Rate (%) in **Japanese** Corpus of Spontaneous (CSJ, 581 hours)

<table>
<thead>
<tr>
<th>Models</th>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attention model (baseline)</td>
<td>11.4</td>
<td>7.9</td>
<td>9.0</td>
</tr>
<tr>
<td>CTC-attention learning (MTL)</td>
<td>10.5</td>
<td>7.6</td>
<td>8.3</td>
</tr>
<tr>
<td>+ Joint decoding</td>
<td>10.0</td>
<td>7.1</td>
<td>7.6</td>
</tr>
</tbody>
</table>
Example of recovering insertion errors (HKUST)

id: (20040717_152947_A010409_B010408-A-057045-057837)

Reference
但是如果你想如果回到了过去你如果带着这个现在的记忆是不是很痛苦啊

Hybrid CTC/attention (w/o joint decoding)
Scores: (#Correctness #Substitution #Deletion #Insertion) 28 2 3 45

w/ Joint decoding
Scores: (#Correctness #Substitution #Deletion #Insertion) 31 1 1 0

HYP: 但是如果你想如果回到了过去你如果带着这个现在的机是不是很痛苦啊
Example of recovering deletion errors (CSJ)

id: (A01F0001_0844951_0854386)

Reference

Hybrid CTC/attention (w/o joint decoding)
Scores: (#Correctness #Substitution #Deletion #Insertion) 30 0 47 0

w/ Joint decoding
Scores: (#Correctness #Substitution #Deletion #Insertion) 67 9 1 0
Discussions

• Hybrid CTC/attention-based end-to-end speech recognition
  – Multi-task learning during training
  – Joint decoding during recognition
    ➔ Make use of both benefits, completely solve alignment issues
• Now we have a good end-to-end ASR tool
  ➔ Apply several challenging ASR issues

• NOTE: This can be solved by large amounts of training data and a lot of tuning. This is one solution (but quite academia friendly)
FAQ

• How to debug attention-based encoder/decoder?

• Please check
  Attention pattern!
  Learning curves!

• It gives you a lot of intuitive information!
Timeline

Shinji’s personal experience for end-to-end speech processing

2016
Initial implementation
- CTC/attention hybrid
- Japanese e2e -> multilingual.

2017
Open source
- share the knowhow
- Kaldi-style
- Jelinek workshop
Speech recognition pipeline

- Require a lot of development for an acoustic model, a pronunciation lexicon, a language model, and finite-state-transducer decoding
- Require linguistic resources
- Difficult to build ASR systems for non-experts

“I want to go to Johns Hopkins campus”
Speech recognition pipeline

- Require **a lot of development** for an acoustic model, a pronunciation lexicon, a language model, and finite-state-transducer decoding.
- Require linguistic resources.
- Difficult to build ASR systems for non-experts.

Pronunciation lexicon:

- A AH
- A'S EY Z
- A(2) EY
- A. EY
- A.'S EY Z
- A.S EY Z
- AAA T R I H P AH L EY
- AABERG AA B ER G
- AACHEN AA K AH N
- AACHENER AA K AH N ER
- AAKER AA K ER
- AALSETH AA L S EH TH
- AAMODT AA M AH T
- AANCOR AA N K AO R
- AARDEMA AA R D EH M AH
- AARDVARK AA R D V AA R K
- AARON EH R AH N
- AARON'S EH R AH N Z
- AARONS EH R AH N Z
- ...
Speech recognition pipeline

- Require a lot of development for an acoustic model, a pronunciation lexicon, a language model, and finite-state-transducer decoding
- Require linguistic resources
- Difficult to build ASR systems for non-experts

“I want to go to Johns Hopkins campus”
From pipeline to integrated architecture

• Train a deep network that directly maps speech signal to the target letter/word sequence
• Greatly simplify the complicated model-building/decoding process
• Easy to build ASR systems for new tasks without expert knowledge
• Potential to outperform conventional ASR by optimizing the entire network with a single objective function

“I want to go to Johns Hopkins campus”
Japanese is a very ASR unfriendly language

“二つ目の要因は計算機資源・音声データの増加及びKaldiやTensorflowなどのオープンソースソフトウェアの普及である”

• No word boundary
• Mix of 4 scripts (Hiragana, Katakana, Kanji, Roman alphabet)
• Frequent many to many pronunciations
  – A lot of homonym (same pronunciations but different chars.)
  – A lot of multiple pronunciations for each char
• Very different phoneme lengths per character
  – “ン”: /n/, …. “侍”: /s/ /a/ /m/ /u/ /r/ /a/ /i/ (from 1 to 7 phonemes per character!)

We need very accurate tokenizer (chasen, mecab) to solve the above problems jointly
My attempt (2016)

- Japanese NLP/ASR: always go through NAIST Matsumoto lab’s tokenizer

- My goal: remove the tokenizer
- Directly predict Japanese text only from audio
- Surprisingly working very well. Our initial attempt reached Kaldi state-of-the-art with a tokenizer (CER~10% (2016) cf. ~5% (2020))
- This was the first Japanese ASR without using tokenizer (one of my dreams)
Multilingual e2e ASR

• Given the Japanese ASR experience, I thought that e2e ASR can handle mixed languages with a single architecture
  ➡ Multilingual e2e ASR (2017)
  ➡ Multilingual code-switching e2e ASR (2018)
Speech recognition pipeline

```
\text{GO}_W\text{T}_{UW}
\text{GO}_W\text{Z}_{T}_{UW}
```

“I want to \text{go to} Johns Hopkins campus”

```
\begin{align*}
    p(X|L) & \quad p(L|W) & \quad p(W)
\end{align*}
```

- Feature extraction
- Acoustic modeling
- Lexicon
- Language modeling

“go to”
“go two”
“go too”
“goes to”
“goes two”
“goes too”
Multilingual speech recognition pipeline

Feature extraction → Acoustic modeling → Lexicon → Language modeling

- Feature extraction
- Acoustic modeling
- Lexicon
- Language modeling

"I want to go to Johns Hopkins campus"

- Feature extraction
- Acoustic modeling
- Lexicon
- Language modeling

"ジョンズホプキンスのキャンパスに行きたいです"

- Feature extraction
- Acoustic modeling
- Lexicon
- Language modeling

"Ich möchte gehen Johns Hopkins Campus"

\[ p(X|L) \] \[ p(L|W) \] \[ p(W) \]
Multilingual speech recognition pipeline

"I want to go to Johns Hopkins campus"

“ジョンズホプキンスのキャンパスに行きたいです”

“Ich möchte gehen Johns Hopkins Campus”

End-to-End Neural Network
Multi-speaker multilingual speech recognition pipeline

"I want to go to Johns Hopkins campus"

"今天下雨了"

$p(X|L)$ $p(L|W)$ $p(W)$
Multi-speaker multilingual speech recognition pipeline

End-to-End Neural Network

“I want to go to Johns Hopkins campus”

“今天下雨了”
Multi-lingual end-to-end speech recognition

[Watanabe+’17, Seki+’18]

- Learn a single model with multi-language data (10 languages)
- **Integrates** language identification and 10-language speech recognition systems
- **No pronunciation lexicons**

Include all language characters and language ID for final softmax to accept all target languages
ASR performance for 10 languages

- Comparison with language dependent systems
- Language-independent single end-to-end ASR works well!

![Graph showing character error rate for different languages.](image)

- **CN**
- **EN**
- **JP**
- **DE**
- **ES**
- **FR**
- **IT**
- **NL**
- **RU**
- **PT**
- **Ave.**

**Character Error Rate [%]**

- **Language dependent**
- **Language independent**

你好
Hello
こんにちは
Hallo
Hola
Bonjour
Ciao
Hallo
Privet
Olá
## Language recognition performance

<table>
<thead>
<tr>
<th></th>
<th>CH</th>
<th>EN</th>
<th>JP</th>
<th>DE</th>
<th>ES</th>
<th>FR</th>
<th>IT</th>
<th>NL</th>
<th>RU</th>
<th>PT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH</td>
<td>train_dev</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>dev</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>EN</td>
<td>test_eval92</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>test_dev93</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>eval1_jpn</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>eval2_jpn</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>eval3_jpn</td>
<td>0.0</td>
<td>0.0</td>
<td>99.9</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>JP</td>
<td>et_de</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>99.7</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>dt de</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>99.7</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>dt es</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>67.9</td>
<td>0.0</td>
<td>31.9</td>
<td>0.0</td>
<td>0.0</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>et es</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>91.1</td>
<td>0.0</td>
<td>8.4</td>
<td>0.1</td>
<td>0.0</td>
<td>0.2</td>
</tr>
<tr>
<td>ES</td>
<td>dt fr</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>99.4</td>
<td>0.0</td>
<td>2.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>et fr</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>99.5</td>
<td>0.0</td>
<td>2.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>dt it</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>99.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
<td>0.0</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>et it</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>98.3</td>
<td>0.2</td>
<td>0.1</td>
<td>0.7</td>
<td>0.0</td>
<td>0.3</td>
</tr>
<tr>
<td>IT</td>
<td>dt nl</td>
<td>0.0</td>
<td>0.0</td>
<td>1.3</td>
<td>0.4</td>
<td>0.4</td>
<td>4.3</td>
<td>19.7</td>
<td>0.0</td>
<td>1.3</td>
</tr>
<tr>
<td></td>
<td>et nl</td>
<td>0.0</td>
<td>0.0</td>
<td>10.0</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>97.9</td>
<td>0.0</td>
<td>0.9</td>
</tr>
<tr>
<td>NL</td>
<td>dt ru</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.2</td>
<td>0.2</td>
<td>0.3</td>
<td>4.3</td>
<td>94.7</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>et ru</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.2</td>
<td>0.2</td>
<td>0.3</td>
<td>4.3</td>
<td>94.7</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>dt pt</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
<td>0.3</td>
<td>2.6</td>
<td>1.7</td>
<td>3.4</td>
<td>0.6</td>
</tr>
<tr>
<td>PT</td>
<td>et pt</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>3.9</td>
<td>3.6</td>
<td>0.3</td>
<td>91.2</td>
<td>0.0</td>
<td>91.5</td>
</tr>
</tbody>
</table>
ASR performance for low-resource 10 languages

• Comparison with language dependent systems

![Graph showing the comparison between language dependent and independent systems for 10 languages: Bangali, Cantonese, Georgian, Haitian, Kurmanji, Pashto, Tamil, Tok Pisin, Turkish, and Vietnamese. The x-axis represents the languages, and the y-axis represents the Character Error Rate [%]. The graph compares the performance of language dependent and independent systems, with the average error rate for each language displayed.]
ASR performance for low-resource 10 languages

- Comparison with language dependent systems

![Bar chart showing comparison between language dependent and independent systems for character error rate across 10 languages.]

~100 languages with CMU Wilderness Multilingual Speech Dataset
[Adams+(2019)]
Actually it was one of the easiest studies in my work

Q. How many people were involved in the development?
   A. 1 person

Q. How long did it take to build a system?
   A. Totally ~1 or 2 day efforts with bash and python scripting (no change of main e2e ASR source code), then I waited 10 days to finish training

Q. What kind of linguistic knowledge did you require?
   A. Unicode (because python2 Unicode treatment is tricky. If I used python3, I would not even have to consider it)

ASRU’17 best paper candidate (not best paper 😞)
## Multi-lingual ASR

(Supporting 10 languages: CN, EN, JP, DE, ES, FR, IT, NL, RU, PT)

<table>
<thead>
<tr>
<th>ID</th>
<th>csj-eval:s00m0070-0242356-0244956:voxforge-et-fr:mirage59-20120206-njp-fr-sb-570</th>
</tr>
</thead>
<tbody>
<tr>
<td>REF:</td>
<td>[JP] 日本でもニュースになったと思いますが [FR] le conseil supérieur de la magistrature est présidé par le président de la république</td>
</tr>
<tr>
<td>ASR:</td>
<td>[JP] 日本でもニュースになったと思いますが [FR] le conseil supérieur de la magistrature est présidée par le président de la république</td>
</tr>
</tbody>
</table>

|----|--------------------------------------------------------------------------|
Timeline

Shinji’s personal experience for end-to-end speech processing

2017

Open source

- share the knowhow
- Kaldi-style
- Jelinek workshop
ESPnet: End-to-end speech processing toolkit

Shinji Watanabe
Center for Language and Speech Processing
Johns Hopkins University

Joint work with Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, Tsubasa Ochiai,

and more and more
ESPnet

- Open source (Apache2.0) end-to-end speech processing toolkit developed at Frederick Jelinek Memorial Summer Workshop 2018
- >3000 GitHub stars, ~100 contributors
- Major concept

Reproducible end-to-end speech processing studies for speech researchers

Keep simplicity

- Follows the Kaldi style
  - Data processing, feature extraction/format
  - Recipes to provide a complete setup for speech processing experiments

I personally don’t like pre-training fine-tuning strategies (but I’m changing my mind)
Functionalities

• Kaldi style data preprocessing
  1) fairly comparable to the performance obtained by Kaldi hybrid DNN systems
  2) easily porting the Kaldi recipe to the ESPnet recipe
• Attention-based encoder-decoder
  • Subsampled BLSTM and/or VGG-like encoder and location-based attention (+10 attentions)
  • beam search decoding
• CTC
  • WarpCTC, beam search (label-synchronous) decoding
• Hybrid CTC/attention
  • Multitask learning
  • Joint decoding with label-synchronous hybrid CTC/attention decoding (solve monotonic alignment issues)
• RNN transducer
  • Warptransducer, beam search (label-synchronous) decoding
• Use of language models
  • Combination of RNNLM/n-gram trained with external text data (shallow fusion)
Timeline

Shinji’s personal experience for end-to-end speech processing

2018

ASR+X

- TTS
- Speech translation
- Speech enhancement + ASR
ASR+X

• This toolkit (ASR+X) covers the following topics complementally

  ESPnet

  Speech translation  TTS

  ASR  Speech enhancement

• Why we can support such wide-ranges of applications?
High-level benefit of e2e neural network

- **Unified** views of multiple speech processing applications based on end-to-end neural architecture
- **Integration** of these applications in a single network
- **Implementation** of such applications and their integrations based on an open source toolkit like ESPnet, nemo, espresso, ctc++, fairseq, opennmt, py, lingvo, etc. etc., in an unified manner
Automatic speech recognition (ASR)

- Mapping speech sequence to character sequence

\[ X = (x(l) \in \mathbb{Z}|l = 1, \ldots, L) \]
\[ L = 43263 \]

\[ X = (x_t \in \mathbb{R}^D|t = 1, \ldots, T) \]
\[ T = 268 \]

“That’s another story”

\[ W = (w_n \in \mathcal{V}|n = 1, \ldots, N) \]
\[ N = 18 \]
Speech to text translation (ST)

- Mapping **speech** sequence in a **source** language to **character** sequence in a **target** language

"Das ist eine andere Geschichte"

That’s another story

\[ X = (x(l) \in \mathbb{Z} | l = 1, \ldots, L) \]
\[ L = 43263 \]

\[ X = (x_t \in \mathbb{R}^D | t = 1, \ldots, T) \]
\[ T = 268 \]

\[ W = (w_n \in \mathcal{V} | n = 1, \ldots, N) \]
\[ N = 31 \]
Text to speech (TTS)

- Mapping **character** sequence to **speech** sequence

"That’s another story"

\[
W = (w_n \in \mathcal{V} | n = 1, \ldots, N) \\
N = 18
\]

\[
X = (x(l) \in \mathbb{Z} | l = 1, \ldots, L) \\
L = 43263
\]

\[
X = (x_t \in \mathbb{R}^D | t = 1, \ldots, T) \\
T = 268
\]
Speech enhancement (SE)

- Mapping *noisy* speech sequence to *clean* speech sequence

\[
X = (x_t \in \mathbb{R}^D | t = 1, \ldots, T) \\
T = 268
\]

\[
X' = (x'_t \in \mathbb{R}^D | t = 1, \ldots, T) \\
T = 268
\]
All of the problems

\[ X = (x_1, x_2, \cdots, x_T) \xrightarrow{f} Y = (y_1, y_2, \cdots, y_N) \]
Unified view with sequence to sequence

- All the above problems: find a mapping function from sequence to sequence (unification)

\[ X = (x_1, x_2, \ldots, x_T) \xrightarrow{f} Y = (y_1, y_2, \ldots, y_N) \]

- ASR: \( X = \text{Speech}, \ Y = \text{Text} \)
- TTS: \( X = \text{Text}, \ Y = \text{Speech} \)
- ST: \( X = \text{Speech (EN)}, \ Y = \text{Text (JP)} \)
- Speech Enhancement: \( X = \text{Noisy speech}, \ Y = \text{Clean speech} \)

- Mapping function \( f(\cdot) \)
  - Sequence to sequence (seq2seq) function
  - ASR as an example
Seq2seq end-to-end ASR

\[ X = (x_1, x_2, \cdots, x_T) \xrightarrow{f} Y = (y_1, y_2, \cdots, y_N) \]

Mapping seq2seq function \( f(\cdot) \)

1. Connectionist temporal classification (CTC)
2. Attention-based encoder decoder
3. Joint CTC/attention (Joint C/A)
4. RNN transducer (RNN-T)
5. Transformer
Unified view

- Target speech processing problems: find a mapping function from sequence to sequence (unification)

\[ X = (x_1, x_2, \cdots, x_T) \xrightarrow{f} Y = (y_1, y_2, \cdots, y_N) \]

- ASR: X = Speech, Y = Text
- TTS: X = Text, Y = Speech
- ...

- Mapping function \((f)\)
  - Attention based encoder decoder
  - Transformer
  - ...

Seq2seq TTS (e.g., Tacotron2) [Shen+ 2018]

- Use seq2seq generate a spectrogram feature sequence
- We can use either attention-based encoder decoder or transformer
Unified view → Unified software design

We design a new speech processing toolkit based on

\[ X = (x_1, x_2, \cdots, x_T) \xrightarrow{f} Y = (y_1, y_2, \cdots, y_N) \]
Unified view \rightarrow Unified software design

We design a new speech processing toolkit based on

\[ X = (x_1, x_2, \cdots, x_T) \]

\[ Y = (y_1, y_2, \cdots, y_N) \]

ESPnet: End-to-end speech processing toolkit
We design a new speech processing toolkit based on ESPnet: End-to-end speech processing toolkit.
Unified view → Unified software design

We design a new speech processing toolkit based on ESPnet: End-to-end speech processing toolkit.

\[ X = (x_1, x_2, \cdots, x_T) \]

\[ Y = (y_1, y_2, \cdots, y_N) \]

CTC
Attention
Joint C/A
RNN-T
Transformer
We design a new speech processing toolkit based on

\[ X = (x_1, x_2, \cdots, x_T) \]

\[ Y = (y_1, y_2, \cdots, y_N) \]

**ESPnet: End-to-end speech processing toolkit**

- Many speech processing applications can be **unified** based on seq2seq
- Again, **Espresso, Nemo, Fairseq, Lingvo** and other toolkits also fully make use of these functions.
Timeline

Shinji’s personal experience for end-to-end speech processing

2018

ASR+X

- TTS
- Speech translation
- Speech enhancement + ASR
Examples of integrations
Dereverberation + beamforming + ASR

- Multichannel end-to-end ASR framework
  - integrates entire process of speech dereverberation (SD), beamforming (SB) and speech recognition (SR), by single neural-network-based architecture
  
  \[ \Downarrow \]

  - SD : DNN-based weighted prediction error (DNN-WPE) [Kinoshita et al., 2016]
  - SB : Mask-based neural beamformer [Erdogan et al., 2016]
  - SR : Attention-based encoder-decoder network [Chorowski et al., 2014]

https://github.com/nttcslab-sp/dnn_wpe, [Subramanian’19]
Beamforming + separation + ASR
[Xuankai Chang., 2019, ASRU]

- Multi-channel (MI) multi-speaker (MO) end-to-end architecture
  - Extend our previous model to **multispeaker end-to-end network**
  - Integrate the **beamforming-based speech enhancement and separation networks** inside the neural network

We call it **MIMO speech**

![Diagram of Multi-channel multi-speaker end-to-end ASR](image)
ASR + TTS feedback loop
→ Unpaired data training

Only audio data to train both ASR and TTS
We do not need a pair data!!!
Timeline

Shinji’s personal experience for end-to-end speech processing

2019-

Improvement

- Transformer
- Open source acceleration
Experiments (~1000 hours)  
Librispeech (Audio book)  

<table>
<thead>
<tr>
<th>Toolkit</th>
<th>dev_clean</th>
<th>dev_other</th>
<th>test_clean</th>
<th>test_other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook wav2letter++</td>
<td>3.1</td>
<td>10.1</td>
<td>3.4</td>
<td>11.2</td>
</tr>
<tr>
<td>RWTH RASR</td>
<td>2.9</td>
<td>8.8</td>
<td>3.1</td>
<td>9.8</td>
</tr>
<tr>
<td>Nvidia Jasper</td>
<td>2.6</td>
<td>7.6</td>
<td>2.8</td>
<td>7.8</td>
</tr>
<tr>
<td>Google SpecAug.</td>
<td>N/A</td>
<td>N/A</td>
<td>2.5</td>
<td>5.8</td>
</tr>
</tbody>
</table>

• Very impressive results by Google
Experiments (~**1000** hours)

Librispeech

<table>
<thead>
<tr>
<th>Toolkit</th>
<th>dev_clean</th>
<th>dev_other</th>
<th>test_clean</th>
<th>test_other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook wav2letter++</td>
<td>3.1</td>
<td>10.1</td>
<td>3.4</td>
<td>11.2</td>
</tr>
<tr>
<td>RWTH RASR</td>
<td>2.9</td>
<td>8.8</td>
<td>3.1</td>
<td>9.8</td>
</tr>
<tr>
<td>Nvidia Jasper</td>
<td>2.6</td>
<td>7.6</td>
<td>2.8</td>
<td>7.8</td>
</tr>
<tr>
<td>Google SpecAug.</td>
<td>N/A</td>
<td>N/A</td>
<td><strong>2.5</strong></td>
<td>5.8</td>
</tr>
<tr>
<td>ESPnet</td>
<td><strong>2.2</strong></td>
<td><strong>5.6</strong></td>
<td>2.6</td>
<td><strong>5.7</strong></td>
</tr>
</tbody>
</table>

- Reached Google’s best performance by community-driven efforts (on September 2019)
GAFAM
Good example of “Collapetition”
= Collaboration + Competition
Experiments (~1000 hours)  
Librispeech

<table>
<thead>
<tr>
<th>Toolkit</th>
<th>dev_clean</th>
<th>dev_other</th>
<th>test_clean</th>
<th>test_other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook wav2letter++</td>
<td>3.1</td>
<td>10.1</td>
<td>3.4</td>
<td>11.2</td>
</tr>
<tr>
<td>RWTH RASR</td>
<td>2.9</td>
<td>8.8</td>
<td>3.1</td>
<td>9.8</td>
</tr>
<tr>
<td>Nvidia Jasper</td>
<td>2.6</td>
<td>7.6</td>
<td>2.8</td>
<td>7.8</td>
</tr>
<tr>
<td>Google SpecAug.</td>
<td>N/A</td>
<td>N/A</td>
<td>2.5</td>
<td>5.8</td>
</tr>
<tr>
<td><strong>ESPnet</strong></td>
<td><strong>2.2</strong></td>
<td><strong>5.6</strong></td>
<td><strong>2.6</strong></td>
<td><strong>5.7</strong></td>
</tr>
<tr>
<td>MS Semantic Mask (ESPnet)</td>
<td><strong>2.1</strong></td>
<td><strong>5.3</strong></td>
<td><strong>2.4</strong></td>
<td><strong>5.4</strong></td>
</tr>
<tr>
<td>Facebook wav2letter Transformer</td>
<td><strong>2.1</strong></td>
<td><strong>5.3</strong></td>
<td><strong>2.3</strong></td>
<td><strong>5.6</strong></td>
</tr>
</tbody>
</table>
Experiments (~**1000** hours)
Librispeech

<table>
<thead>
<tr>
<th>Toolkit</th>
<th>dev_clean</th>
<th>dev_other</th>
<th>test_clean</th>
<th>test_other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook wav2letter++</td>
<td>3.1</td>
<td>10.1</td>
<td>3.4</td>
<td>11.2</td>
</tr>
<tr>
<td>RWTH RASR</td>
<td>2.9</td>
<td>8.8</td>
<td>3.1</td>
<td>9.8</td>
</tr>
<tr>
<td>Nvidia Jasper</td>
<td>2.6</td>
<td>7.6</td>
<td>2.8</td>
<td>7.8</td>
</tr>
<tr>
<td>Google SpecAug.</td>
<td>N/A</td>
<td>N/A</td>
<td>2.5</td>
<td>5.8</td>
</tr>
<tr>
<td><strong>ESPnet</strong></td>
<td>2.2</td>
<td>5.6</td>
<td>2.6</td>
<td>5.7</td>
</tr>
<tr>
<td>MS Semantic Mask (ESPnet)</td>
<td>2.1</td>
<td><strong>5.3</strong></td>
<td>2.4</td>
<td><strong>5.4</strong></td>
</tr>
<tr>
<td>Facebook wav2letter Transformer</td>
<td>2.1</td>
<td><strong>5.3</strong></td>
<td>2.3</td>
<td>5.6</td>
</tr>
<tr>
<td>Kaldi (Pipeline) by ASAPP</td>
<td><strong>1.8</strong></td>
<td>5.8</td>
<td><strong>2.2</strong></td>
<td>5.8</td>
</tr>
</tbody>
</table>

*(January 2020)*
Transformer is powerful for multilingual ASR

One of the most stable and biggest gains compared with other multilingual ASR techniques
By Philipp Koehn
Self-Attentive End-to-End Diarization [Fujita+(2019)]

Audio Feature

- SAD
- Speaker embedding
- Scoring transform

Result

- Model-wise training
- Unsupervised
- Cannot handle speech overlap

SAD neural network
Speaker embedding neural network
Same/Diff covariance matrices
Self-Attentive End-to-End Diarization
[Fujita+(2019)]

Audio Feature

- Multi-label classification with permutation-free loss

EEND neural network

- Only one network to be trained
- Fully-supervised
- Can handle speech overlap

Result

[Fujita, Interspeech 2019]
Self-Attentive End-to-End Diarization
[Fujita+(2019)]

- Only one network to be trained
- Fully-supervised
- Can handle speech overlap

Multi-label classification with permutation-free loss

Audio Feature

Result

[Fujita, Interspeech 2019]

<table>
<thead>
<tr>
<th></th>
<th>CALL HOME DER (%)</th>
<th>CSJ EDR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-vector</td>
<td>11.53</td>
<td>22.96</td>
</tr>
<tr>
<td>EEND BLSTM</td>
<td>23.07</td>
<td>25.37</td>
</tr>
<tr>
<td>EEND Self-attention</td>
<td>9.54</td>
<td>20.48</td>
</tr>
</tbody>
</table>

- Outperform the state-of-the-art x-vector system!
- Check https://github.com/hitachi-speech/EEND
FAQ (before transformer)

• How to debug attention-based encoder/decoder?

• Please check
  Attention pattern!
  Learning curves!

• It gives you a lot of intuitive information!
FAQ (after transformer)

• How to debug attention-based encoder/decoder?
  
  • Please check  
    Attention pattern (including self attention)!
    Learning curves!
  
  • It gives you a lot of intuitive information!
  
  • Tune optimizers!
Shinji’s personal experience for end-to-end speech processing

- **2015**
  - First impression
    - No more conditional independence assumption
    - DNN tool blossom

- **2016**
  - Initial implementation
    - CTC/attention hybrid
    - Japanese e2e -> multilingual.

- **2017**
  - Open source
    - share the knowhow
    - Kaldi-style
    - Jelinek workshop

- **2018**
  - ASR+X
    - TTS
    - Speech translation
    - Speech enhancement + ASR

- **2019**
  - Improvement
    - Transformer
    - Open source acceleration

- **2020**
What’s next?

• Non autoregressive ASR

• New architecture
  • Conformer

• Time-domain processing (real end-to-end including feature extraction and speech enhancement)

• Differentiable WFST
Thanks!