# Introduction of ESPnet, End-to-End Speech Processing Toolkit

**Shinji Watanabe**
Carnegie Mellon University
**Pengcheng Guo**
Northwestern Polytechnical University
**Sathvik Udupa**
Indian Institute of Science

MUCS 2021: MUltilingual and Code-Switching ASR Challenges for Low Resource Indian Languages
**12-13 August 2021**

# Overview of today's tutorial

- **5pm to 6pm:** part I presentation by Shinji
  - Introduction of end-to-end ASR and ESPnet
- **6pm to 6:30 pm:** Q&A for part I and break
- **6:30pm to 7pm:** part II presentation by Pengcheng
  - Advanced techniques in ESPnet
- **7pm to 7:15 pm:** part II espnet mucs recipe by Sathvik
  - espnet mucs recipe, and demo
- **7:15pm to 7:30pm:** summary and Q&A by Shinji

# Introduction of ESPnet, End-to-End Speech Processing Toolkit

## Part I: Introduction of end-to-end ASR and ESPnet


**Shinji Watanabe**
Carnegie Mellon University
**Pengcheng Guo**
Northwestern Polytechnical University
**Sathvik Udupa**
Indian Institute of Science


MUCS 2021: MUltilingual and Code-Switching ASR Challenges for Low Resource Indian Languages

**12-13 August 2021**

# 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

# Timeline

## Shinji's personal experience for end-to-end speech processing

**-2015**

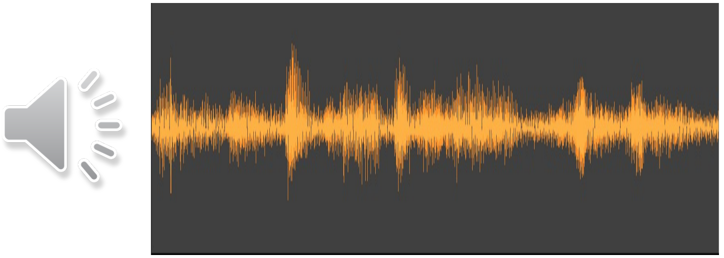
**First impression**

- No more conditional independence assumption
- DNN tool blossom

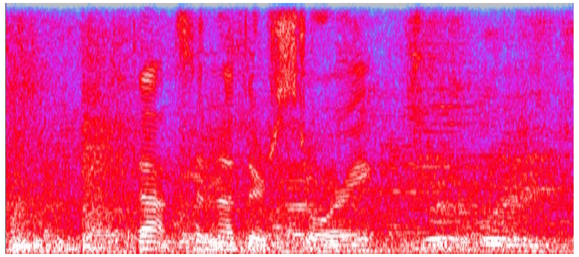
# Noisy channel model (1970s-)

# Noisy channel model (1970s-)

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



$$X = \{x_l \in \mathbb{Z} | l = 1, \dots, L\}$$
$$L = 43263$$



$$X = \{\mathbf{x}_t \in \mathbb{C}^D | t = 1, \dots, T\}$$
$$T = 268$$

"That's another story"

$$W = \{w_n \in \mathcal{V} | n = 1, \dots, N\}$$
$$N = 3$$

# Noisy channel model (1970s-)

$$\arg\max_{\mathrm{W}} p(W|X)$$

$X$: Speech sequence
$W$: Text sequence

# Noisy channel model (1970s-)

$L$: Phoneme sequence

$$\arg\max_{\mathrm{W}} p(W|X) = \arg\max_{\mathrm{W}} p(X|W)p(W)$$
$$\approx \arg\max_{\mathrm{W,L}} p(X|L, \cancel{W})p(L|W)p(W)$$

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

# Noisy channel model (1970s-)

$$\arg\max_{\mathrm{W}} p(W|X) = \arg\max_{\mathrm{W}} p(X|W)p(W)$$
$$\approx \arg\max_{\mathrm{W,L}} p(X|L,\cancel{W})p(L|W)p(W)$$

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

- Factorization
- Conditional independence (Markov) assumptions

# Noisy channel model (1970s-)

$$\arg\max_{\mathrm{W}} p(W|X) = \arg\max_{\mathrm{W}} p(X|W)p(W)$$

- **Machine translation**
  - $p(X|W)$: Translation model
  - $p(W)$: Language model

# Noisy channel model (1970s-)

$$\arg\max_{\mathrm{W}} p(W|X) = \arg\max_{\mathrm{W}} p(X|W)p(W)$$
$$\approx \arg\max_{\mathrm{W,L}} p(X|L, \cancel{W})p(L|W)p(W)$$

- **Speech recognition**
  - $p(X|L)$: Acoustic model (Hidden Markov model)
  - $p(L|W)$: Lexicon
  - $p(W)$: Language model (n-gram)
- Continued 40 years

# Noisy channel model (1970s-)

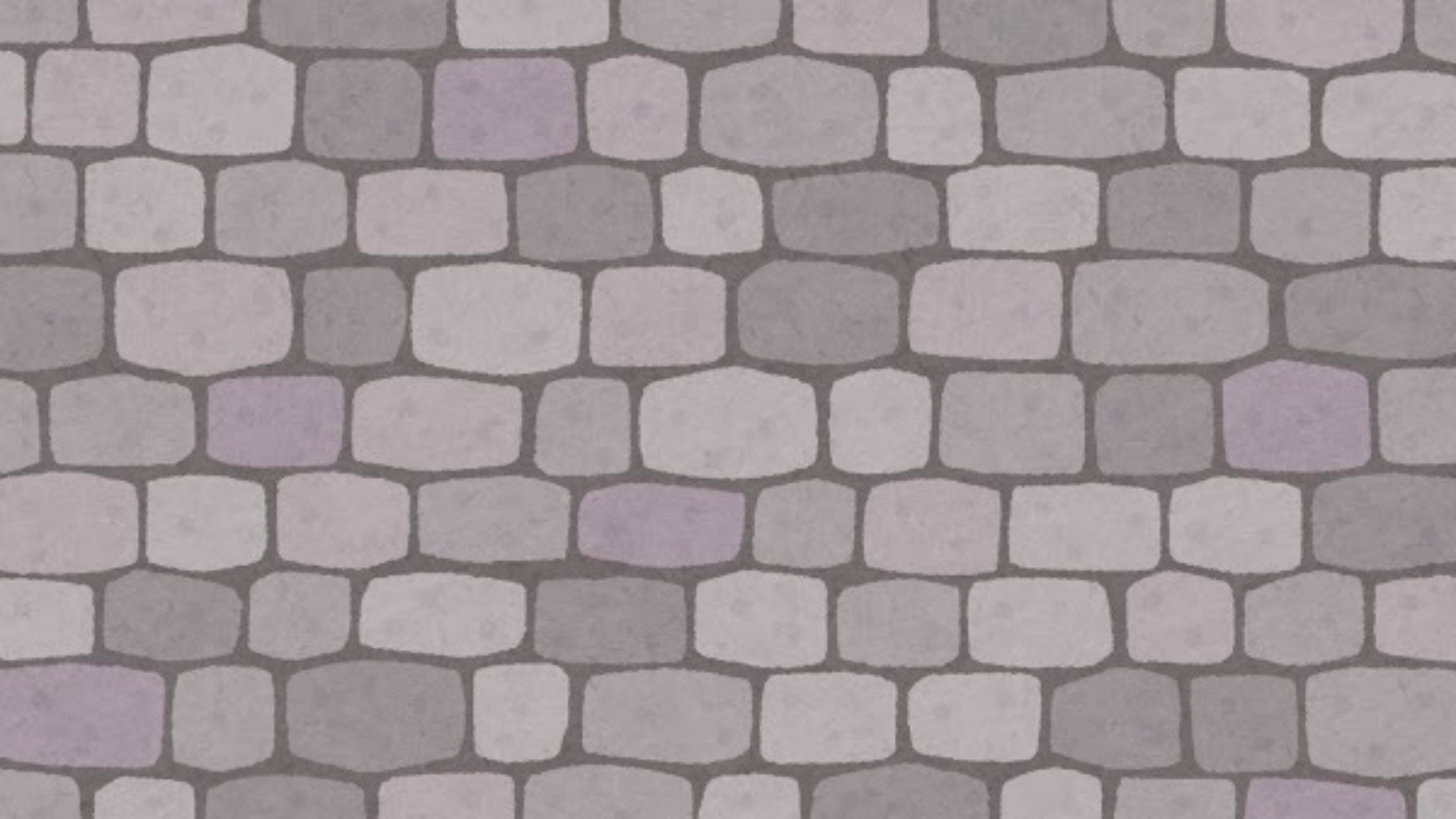
$$\arg\max_{\mathrm{W}} p(W|X) = \arg\max_{\mathrm{W}} p(X|W)p(W)$$
$$\approx \arg\max_{\mathrm{W,L}} p(X|L,\cancel{W})p(L|W)p(W)$$

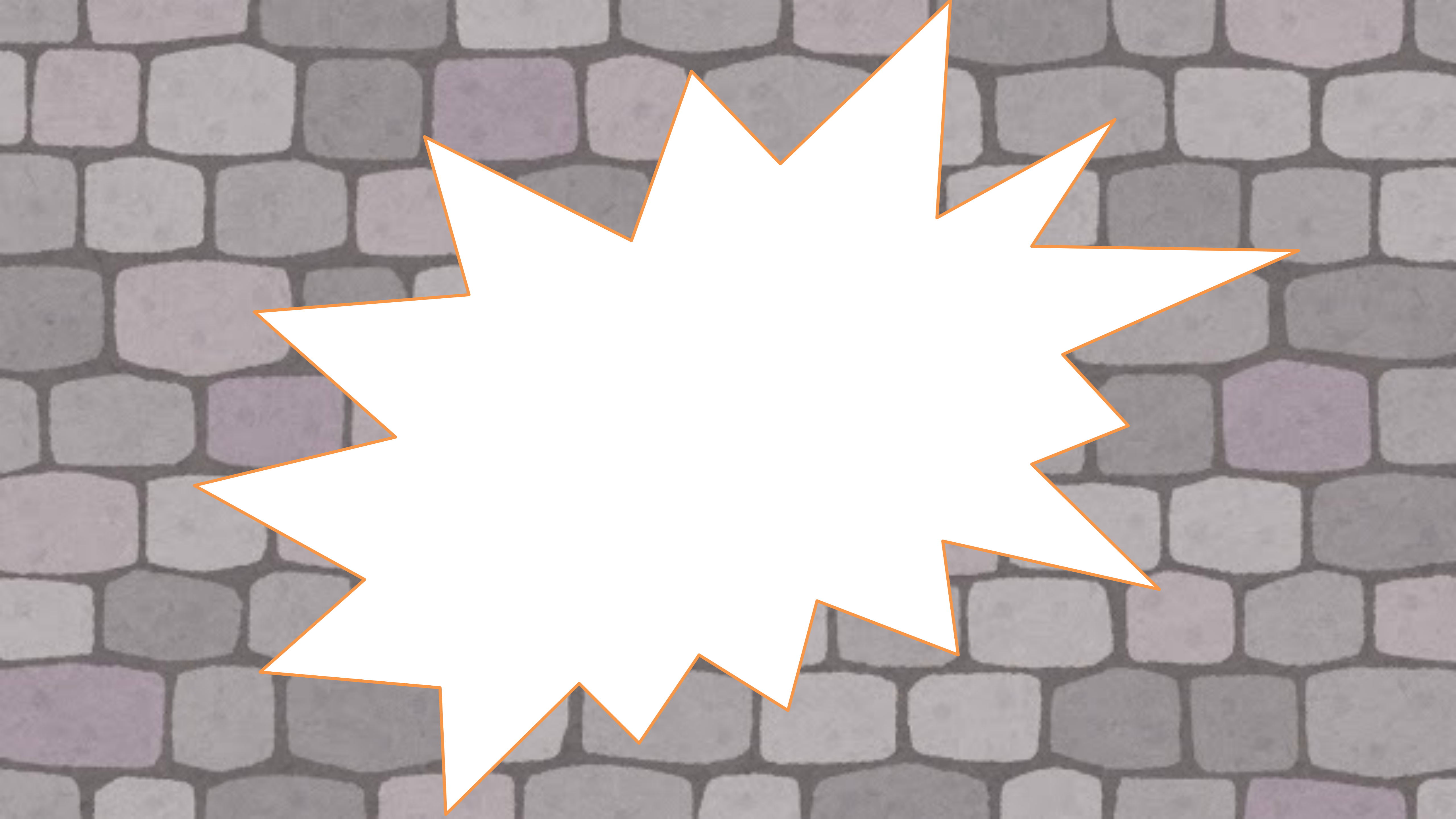
- **Speech recognition**
  - $p(X|L)$: Acoustic model
  - $p(L|W)$: Lexicon
  - $p(W)$: Language model
- Continued 40 years

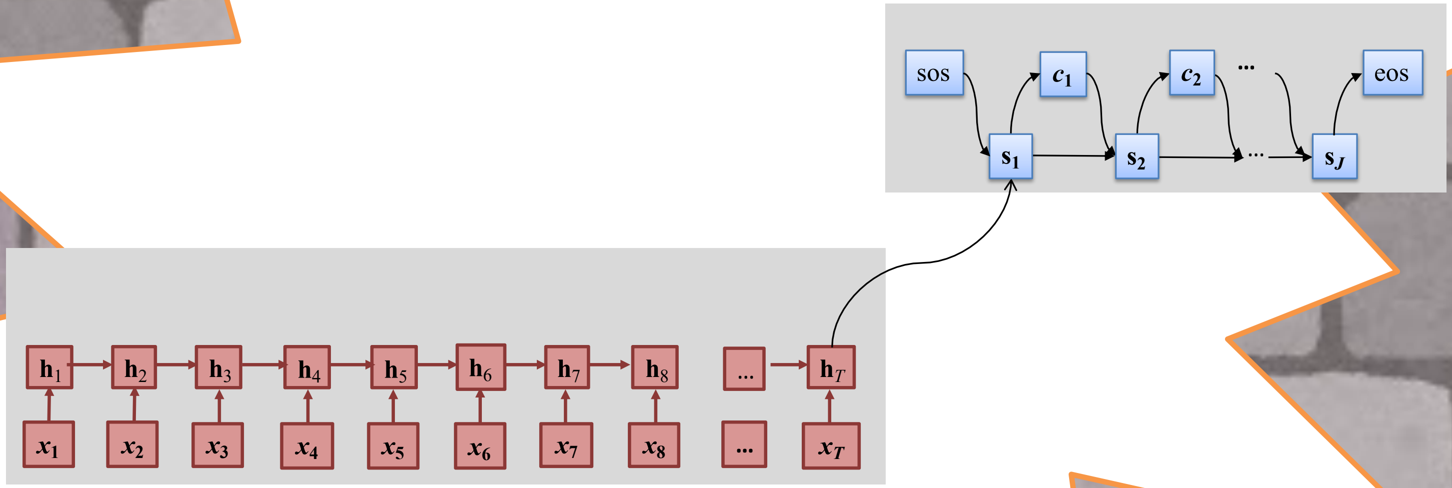
**Big barrier**:

noisy channel model
HMM
n-gram
etc.

# However,

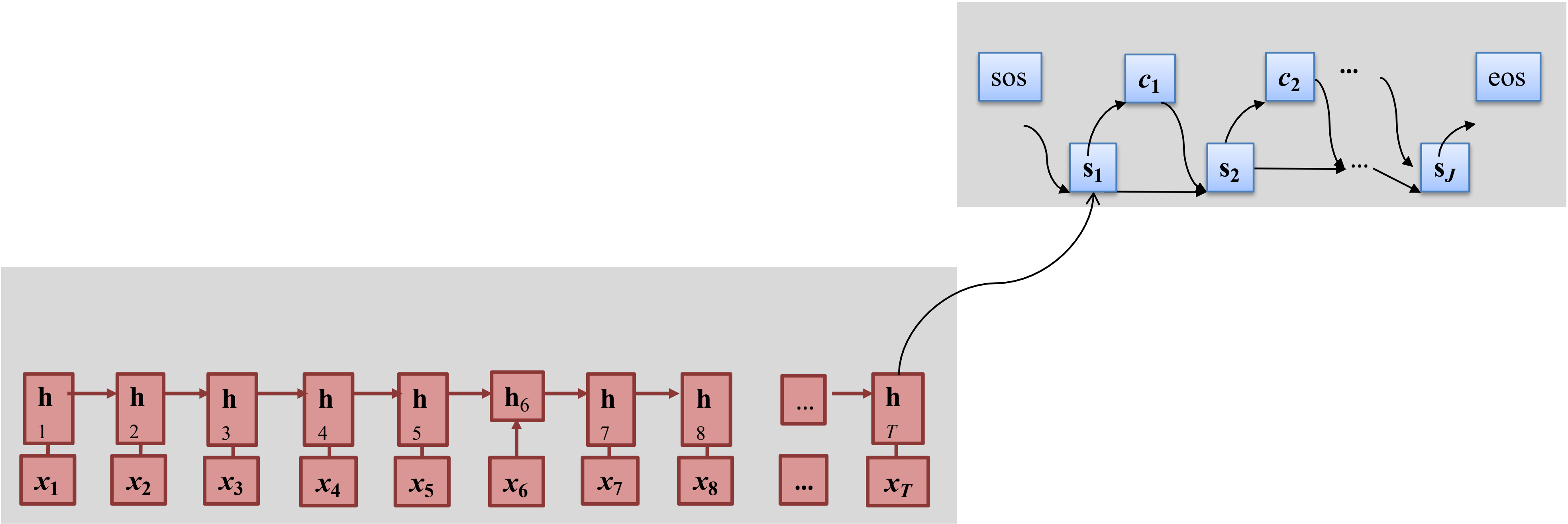






sos
$c_1$
$c_2$
...
eos
$\mathbf{s}_1$
$\mathbf{s}_2$
...
$\mathbf{s}_J$
$\mathbf{h}_1$
$\mathbf{h}_2$
$\mathbf{h}_3$
$\mathbf{h}_4$
$\mathbf{h}_5$
$\mathbf{h}_6$
$\mathbf{h}_7$
$\mathbf{h}_8$
...
$\mathbf{h}_T$
$x_1$
$x_2$
$x_3$
$x_4$
$x_5$
$x_6$
$x_7$
$x_8$
...
$x_T$

# "End-to-End" Processing Using Sequence to Sequence



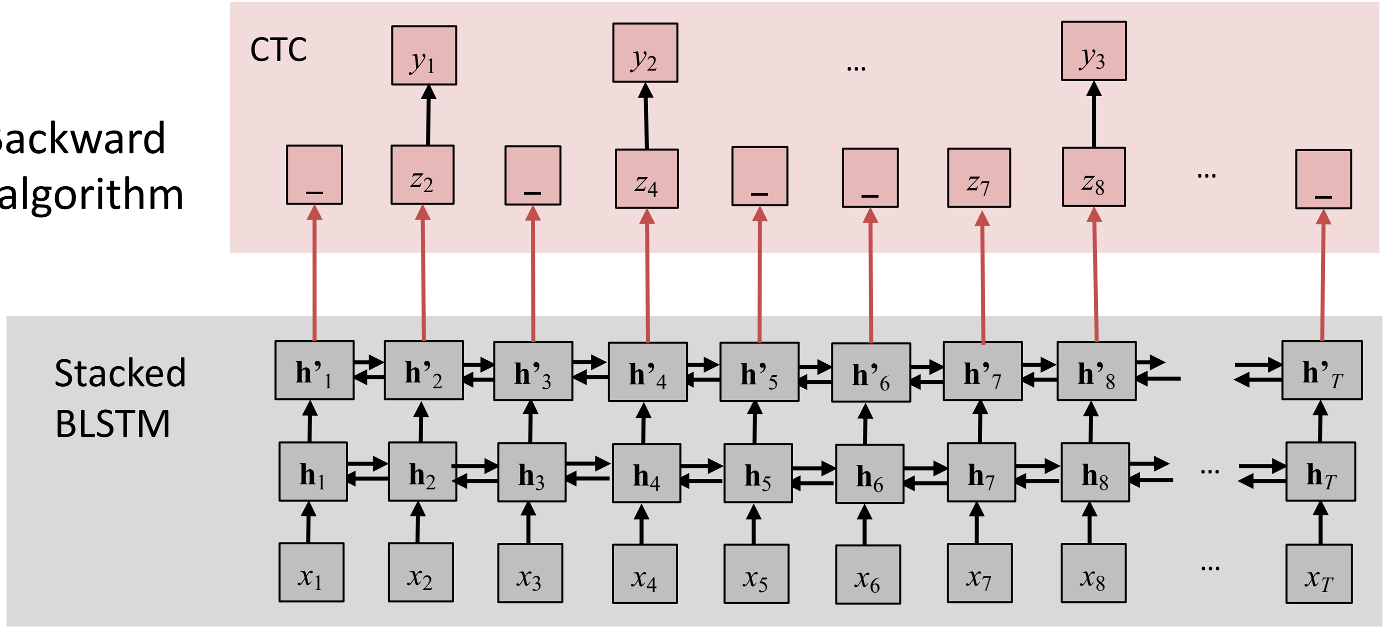

- Directly model $p(W|X)$ with a **single neural network**
  - **Integrate** acoustic $p(X|L)$, lexicon $p(L|W)$, and language $p(W)$ models
- Great success in neural machine translation

# Connectionist temporal classification (CTC)

[Graves+ 2006, Graves+ 2014, Miao+ 2015]

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

Forward-Backward or Viterbi algorithm

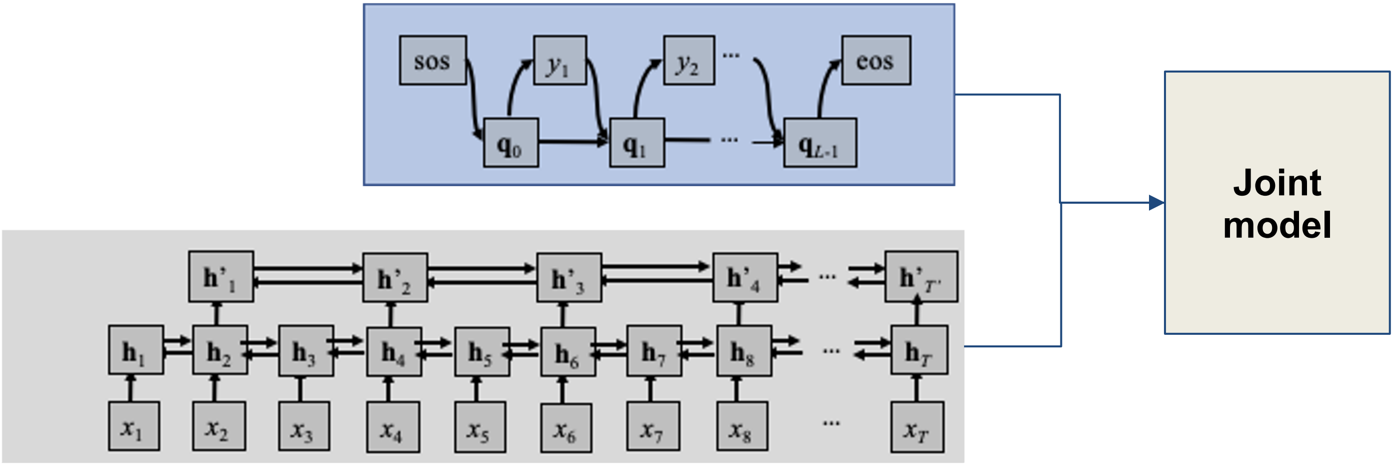


End-to-end ASR (1)

# RNN transducer

[Graves+ 2006, Graves+ 2014, Miao+ 2015]

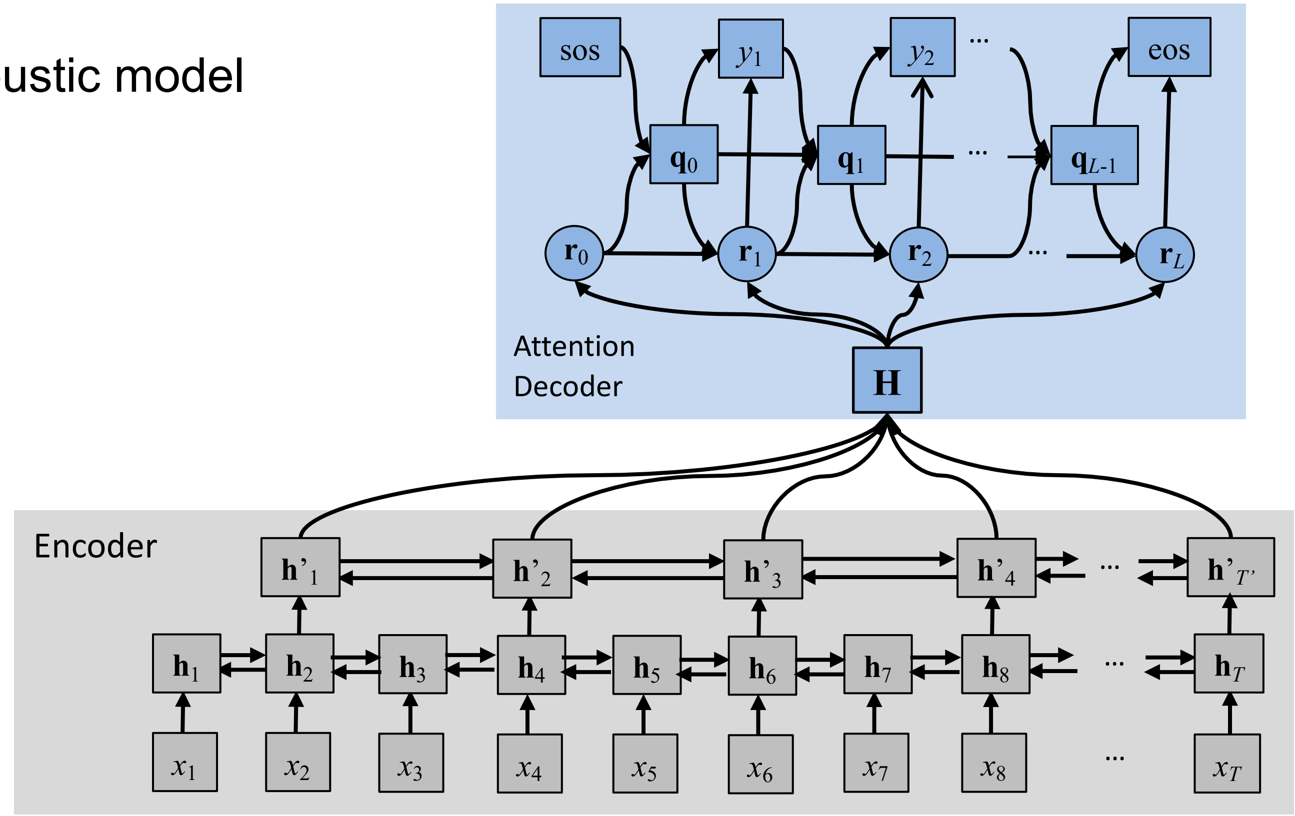
- Encoder to capture the acoustic information
- Label prediction similar to an LM
- Joint model to integrate both information




$$p(y_n | \mathbf{x}_t, y_{n-1})$$

# 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

- Attentio 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

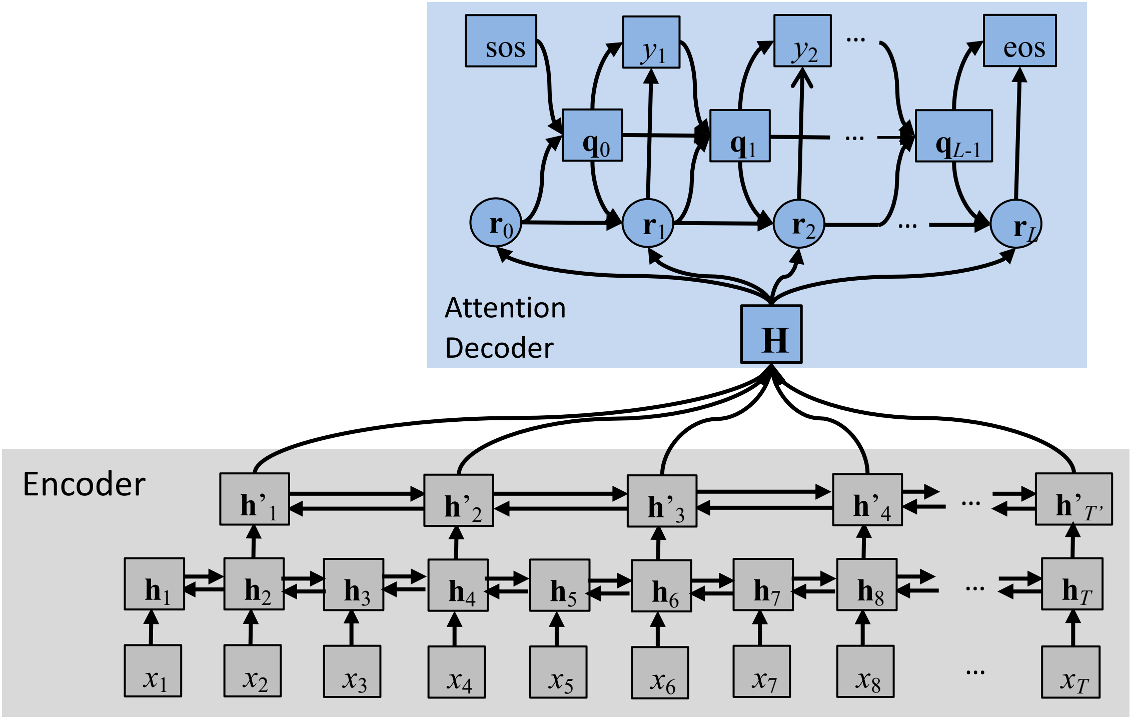
**2016**

**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.



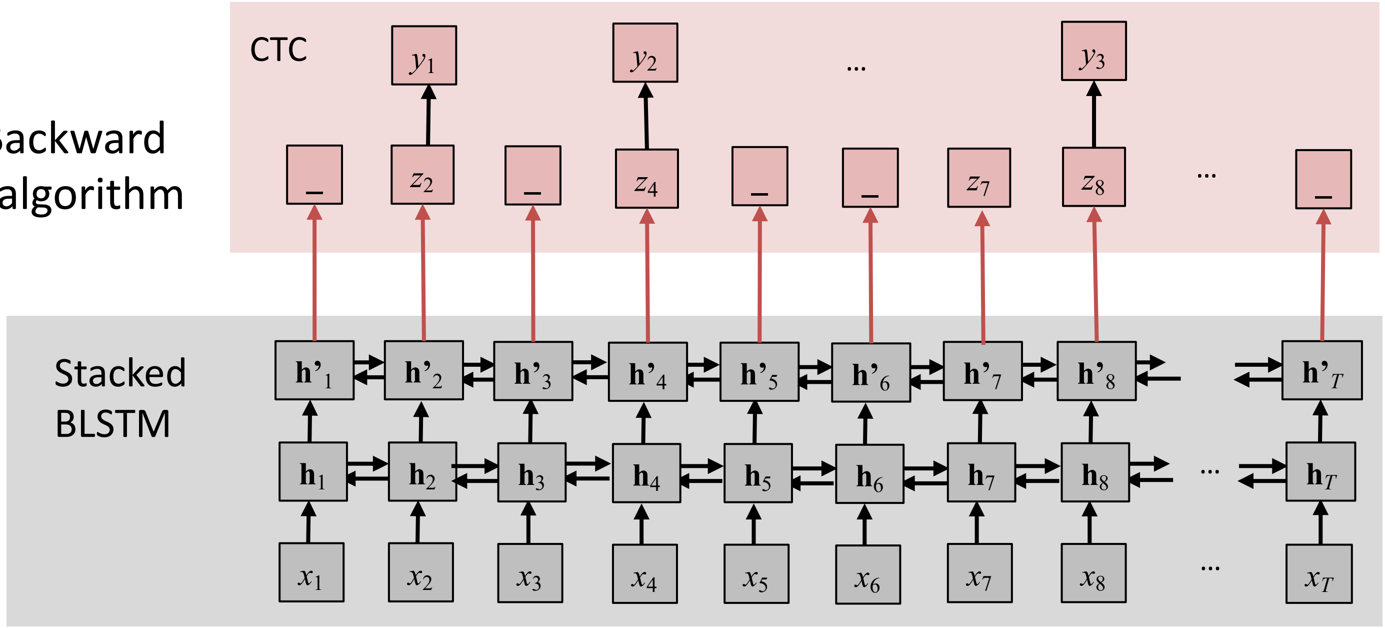
End-to-end ASR (1)

# Connectionist temporal classification (CTC)

[Graves+ 2006, Graves+ 2014, Miao+ 2015]

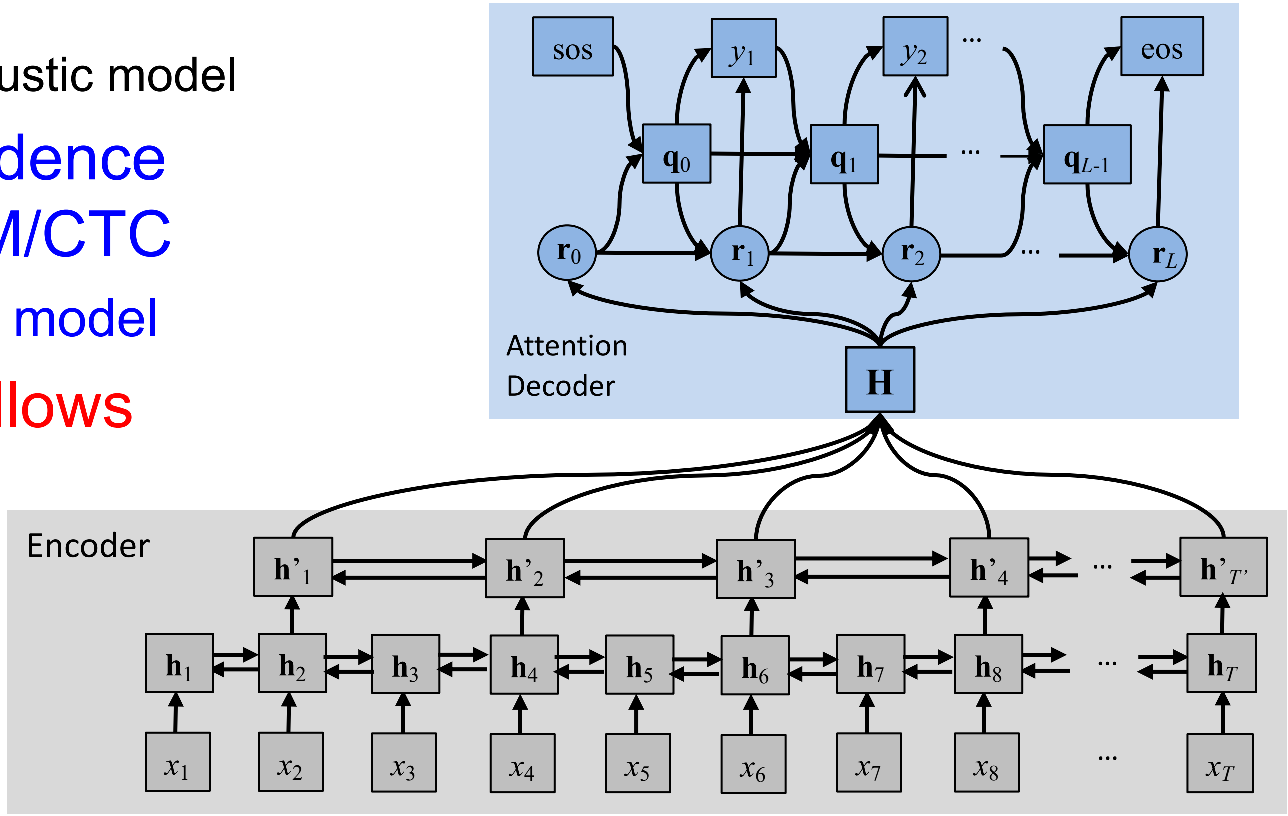
- Use bidirectional RNNs to predict frame-based labels including blanks
- Find alignments between $X$ and $Y$ using dynamic programming
- Relying on conditional independence assumptions (similar to HMM)
- Output sequence is not well modeled (no language model)

Forward-Backward
or Viterbi algorithm



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

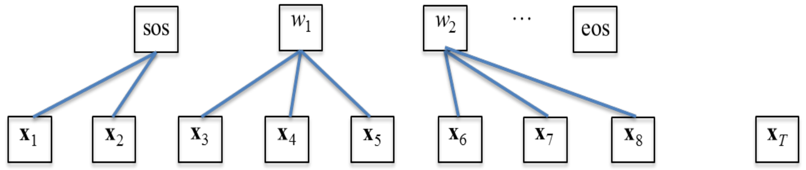
- Combine acoustic and language models in a single architecture
  - Encoder: DNN part of acoustic model
  - Decoder: language model
  - Attention: HMM part of acoustic model
- No conditional independence assumption unlike HMM/CTC
  - More precise seq-to-seq model
- Attention mechanism allows too flexible alignments
  - Hard to train the model from scratch



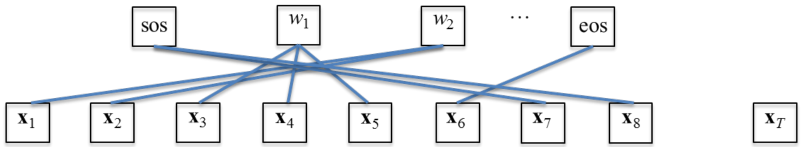
# Input/output alignment by temporal attention

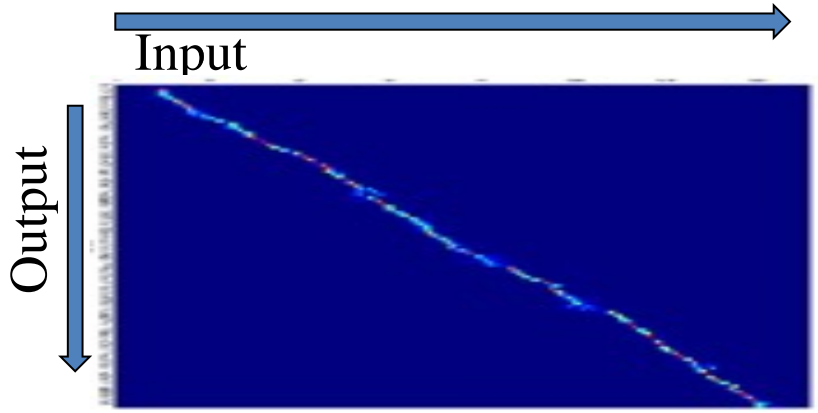
- Unlike CTC, attention model does not preserve order of inputs
- Our desired alignment in ASR task is **monotonic**
- Not regularized alignment makes the model **hard to learn** from scratch

HMM or CTC case

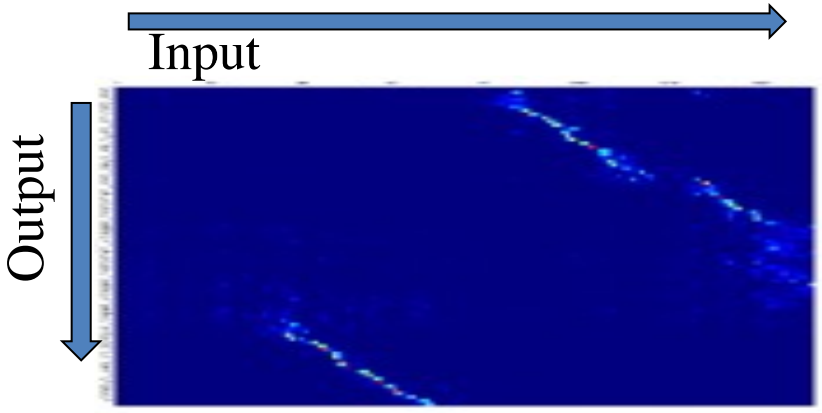



Attention model case




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**

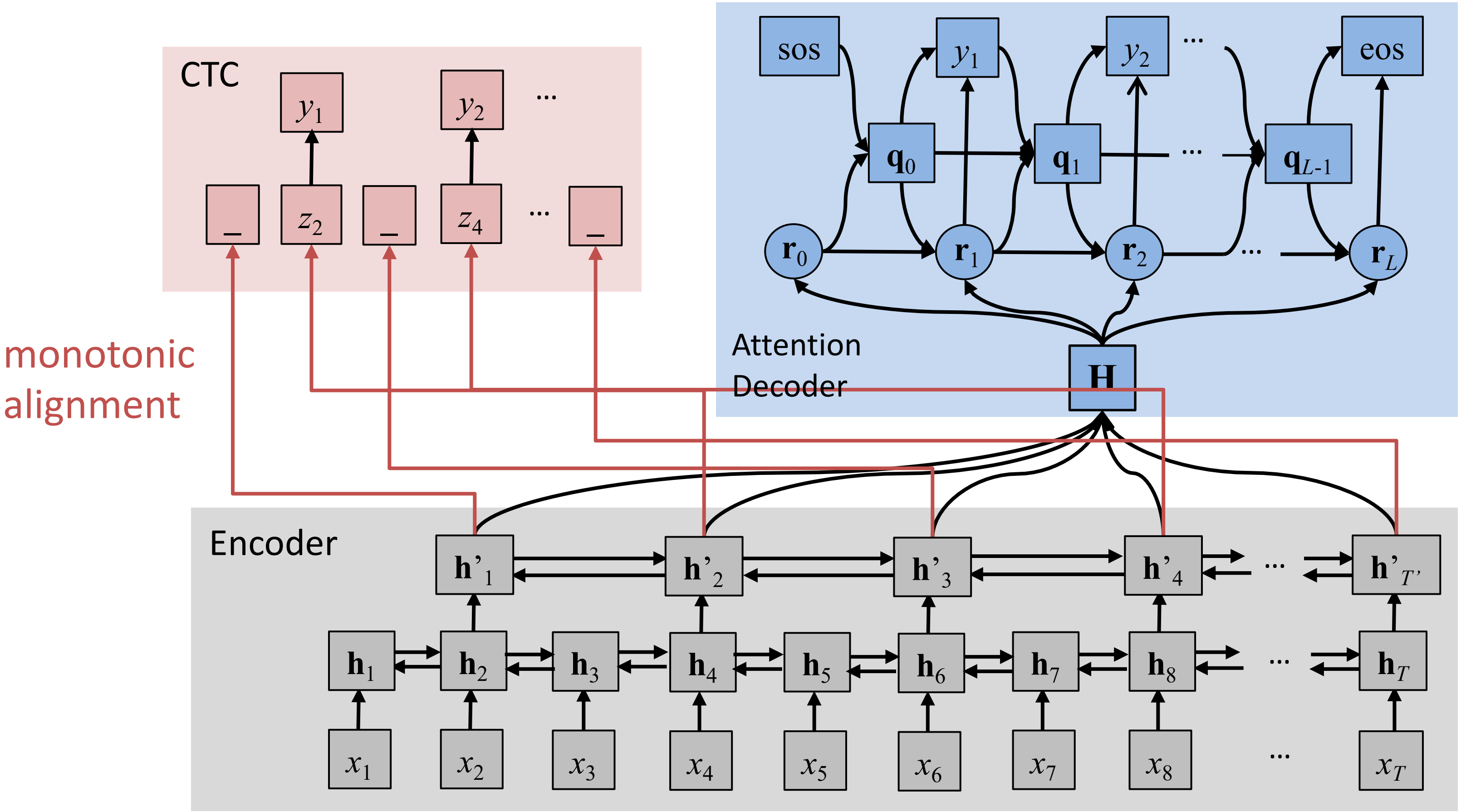
How to combine both?

- 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]

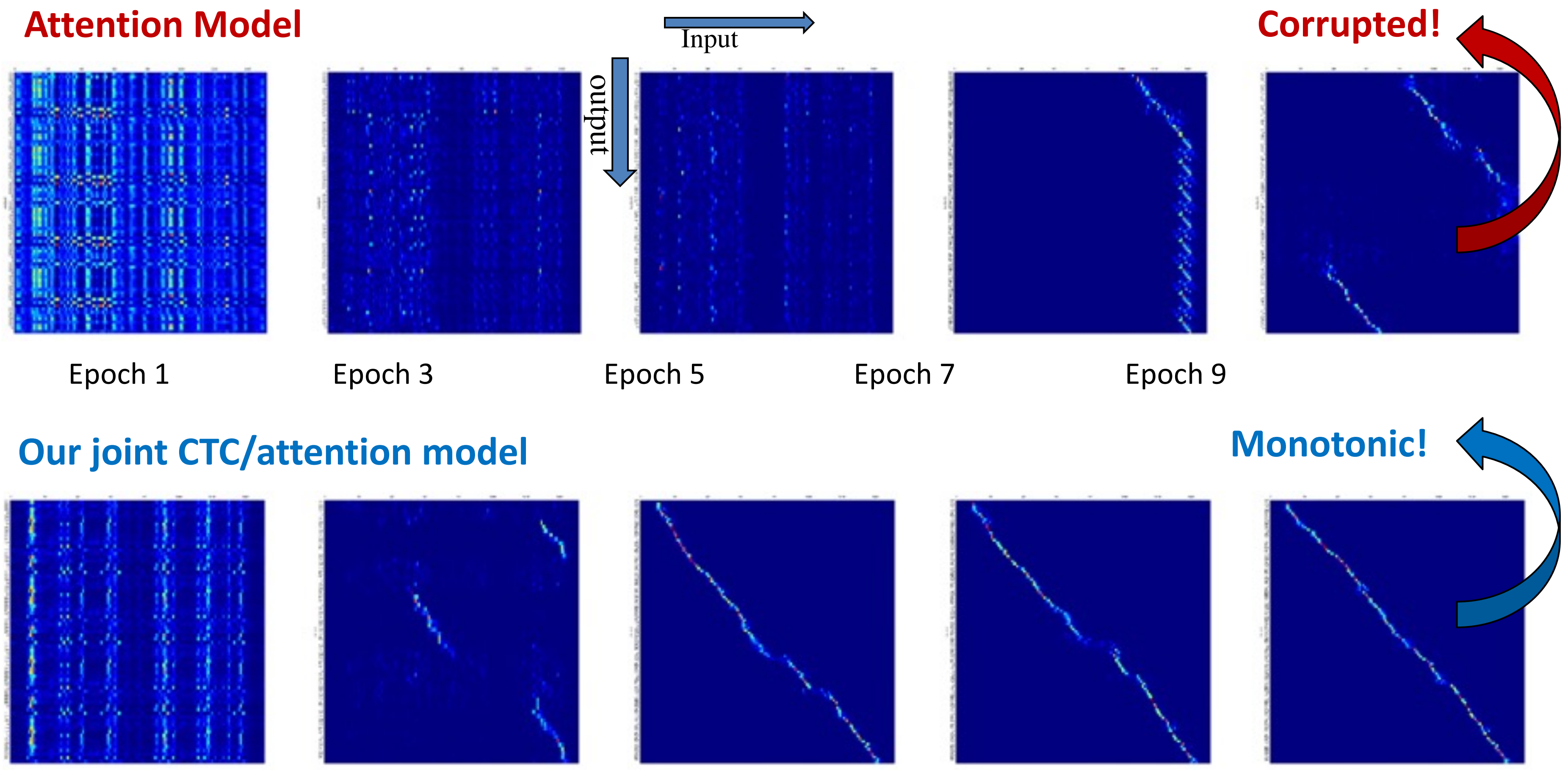
Multitask learning: $\mathcal{L}_{\mathrm{MTL}} = \lambda \mathcal{L}_{\mathrm{CTC}} + (1-\lambda)\mathcal{L}_{\mathrm{Attention}}$ λ: CTC weight




CTC guides attention alignment to be monotonic

# More robust input/output alignment of attention

- Alignment of one selected utterance from CHiME4 task

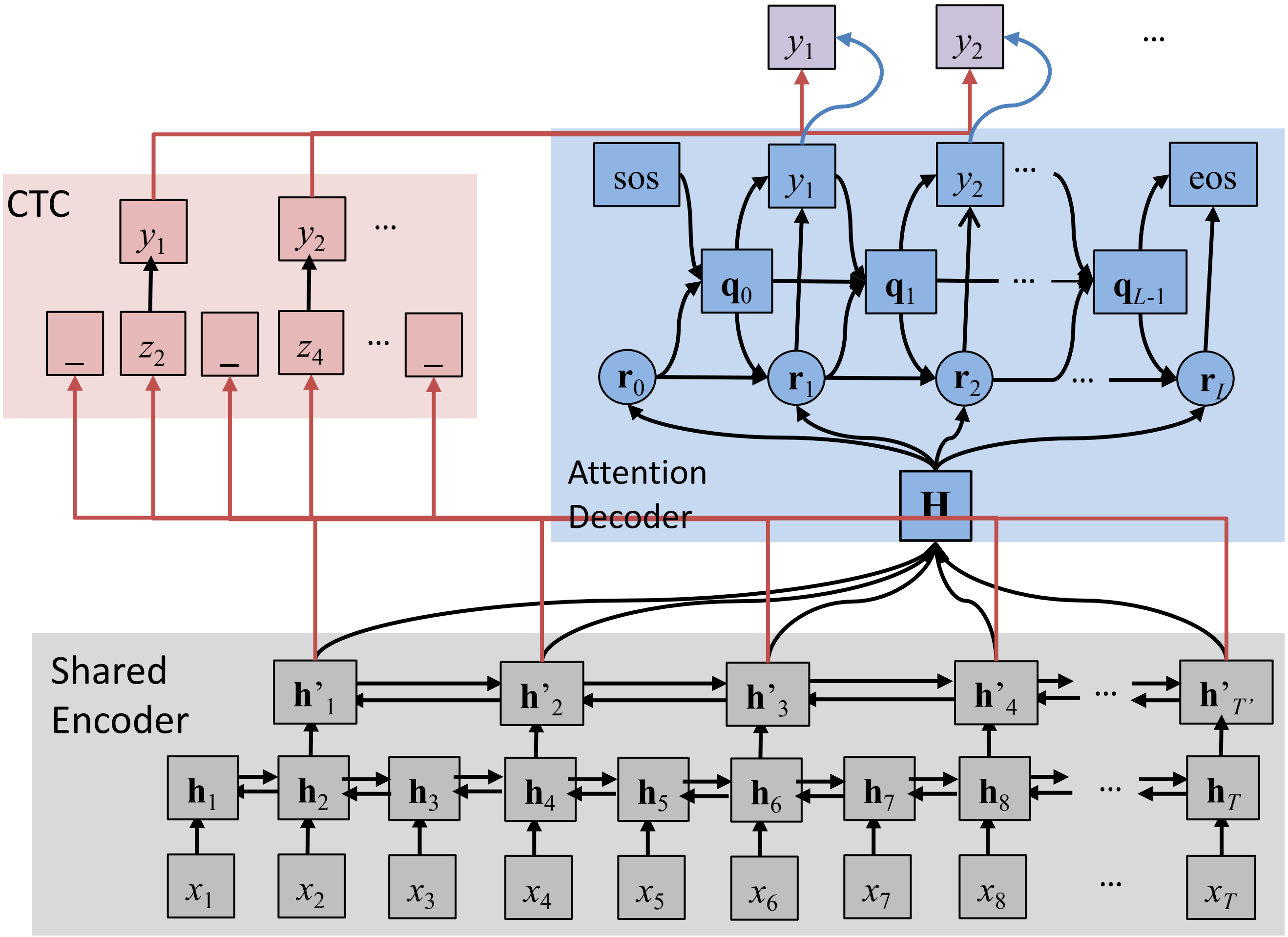



Faster convergence

# 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)

| Models | Dev. | Eval |
|---|---|---|
| Attention model (baseline) | 40.3 | 37.8 |
| CTC-attention learning (MTL) | 38.7 | 36.6 |
| + Joint decoding | **35.5** | **33.9** |

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

| Models | Task 1 | Task 2 | Task 3 |
|---|---|---|---|
| Attention model (baseline) | 11.4 | 7.9 | 9.0 |
| CTC-attention learning (MTL) | 10.5 | 7.6 | 8.3 |
| + Joint decoding | **10.0** | **7.1** | **7.6** |

# Example of recovering insertion errors (HKUST)

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

**Reference**

但是如果你想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)**

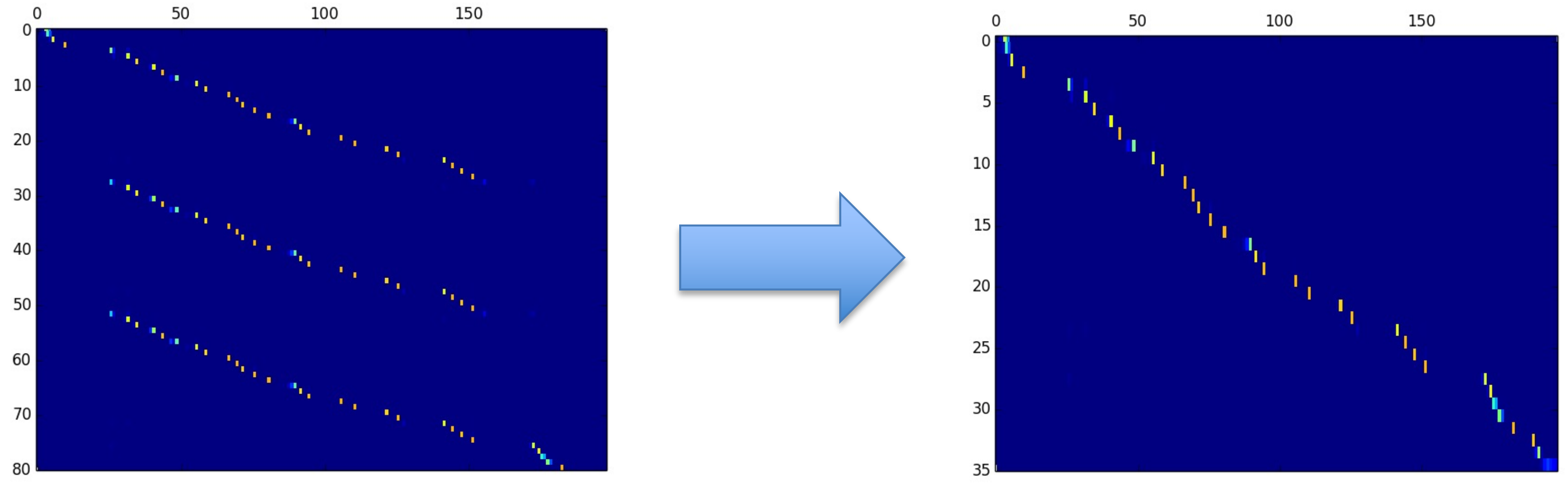
Scores: (#Correctness #Substitution #Deletion #Insertion) 28 2 3 45

但是如果你想想如果回到了过去你如果带着这个现在的节如果你想想如果回到了过去你如果带着这个现在的节如果你想想如果回到了过去你如果带着这个现在的机是不是很・・・

**w/ Joint decoding**

Scores: (#Correctness #Substitution #Deletion #Insertion) 31 1 1 0

HYP: 但是如果你想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**

ま た え 飛 行 時 の エ コ ー ロ ケ ー シ ョ ン 機 能 を よ り 詳 細 に 解 明 す る 為 に 超 小 型 マ イ
ク ロ ホ ン お よ び 生 体 ア ン プ を コ ウ モ リ に 搭 載 す る こ と を 考 え て お り ま す そ う す
る こ と に よ っ て

**Hybrid CTC/attention (w/o joint decoding)**

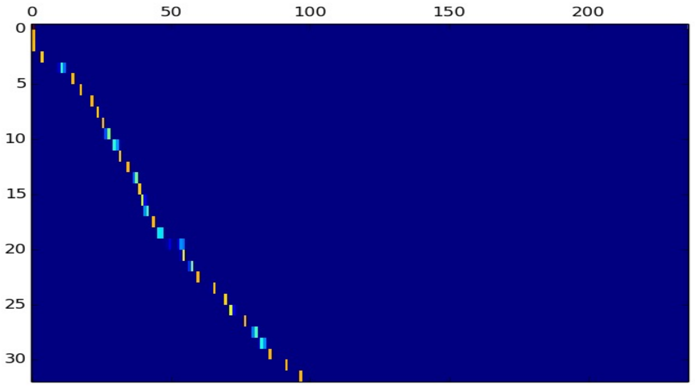
Scores: (#Correctness #Substitution #Deletion #Insertion) 30 0 47 0

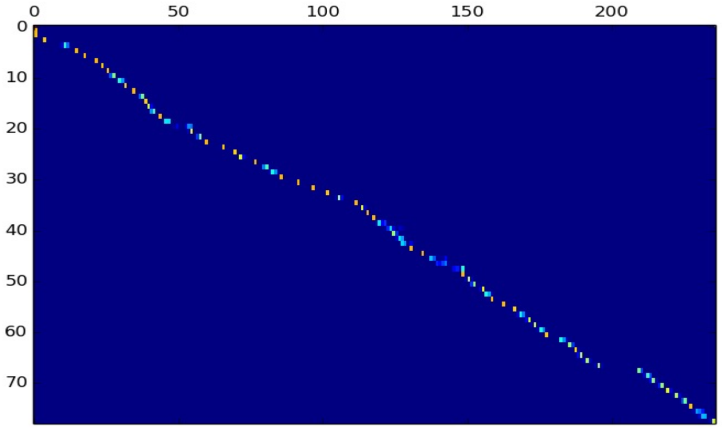
ま た え 飛 行 時 の エ コ ー ロ ケ ー シ ョ ン 機 能 を よ り 詳 細 に 解 明 す る
為 ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・
・ ・ ・ ・ ・ ・ ・ ・ ・ ・ に ・ ・ ・

**w/ Joint decoding**

Scores: (#Correctness #Substitution #Deletion #Insertion) 67 9 1 0

ま た え 飛 行 時 の エ コ ー ロ ケ ー シ ョ ン 機 能 を よ り 詳 細 に 解 明 す る 為 に 長 国 型 マ イ
ク ロ ホ ン お ・ い く 声 単 位 方 を コ ウ モ リ に 登 載 す る こ と を 考 え て お り ま す そ う す
る こ と に よ っ て



# 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)

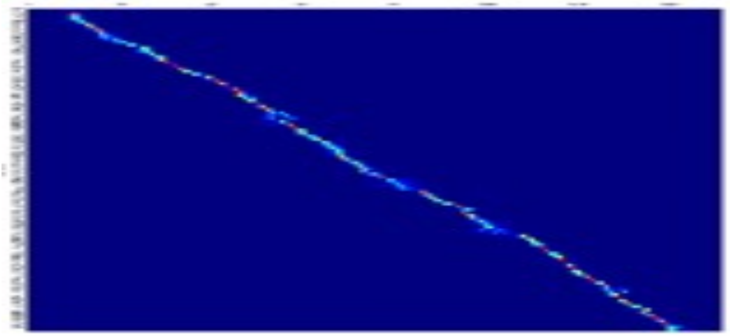
# FAQ

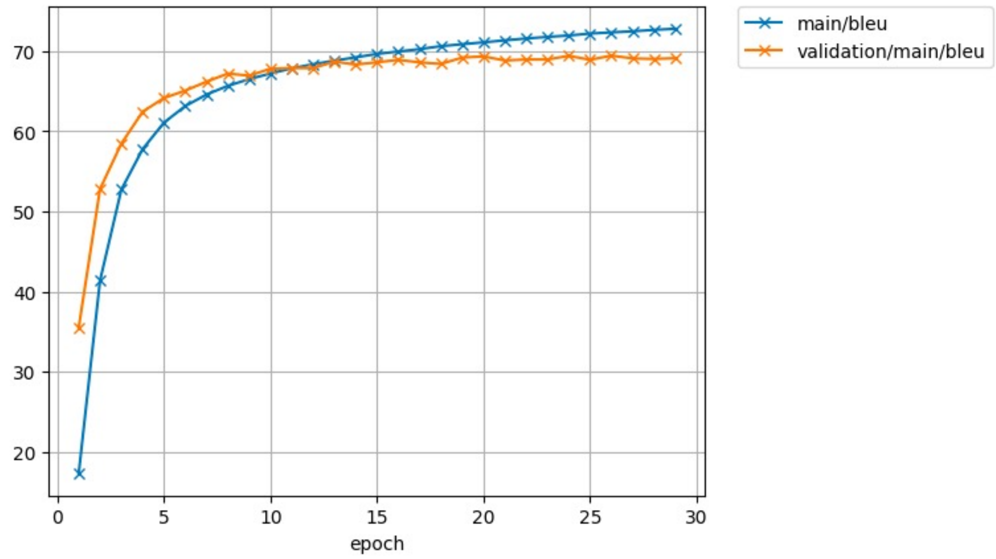
- How to debug attention-based encoder/decoder?
- Please check

  **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**

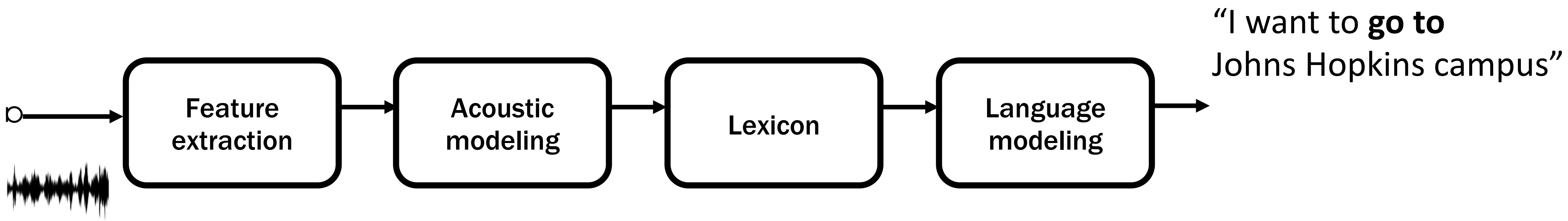
- CTC/attention hybrid
- Japanese e2e -> multilingual.

**2017**

**Open source**

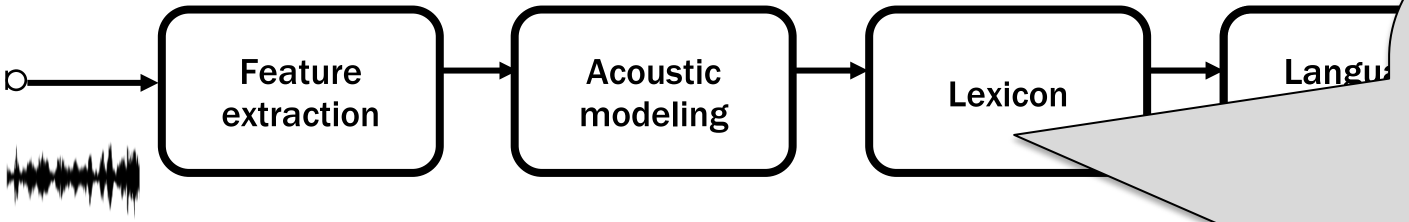
- share the knowhow
- Kaldi-style
- Jelinek workshop

# 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

# Speech recognition pipeline



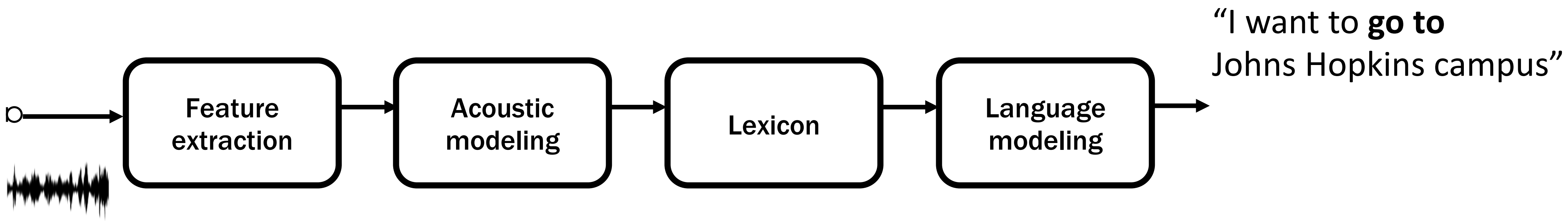

**Pronunciation lexion**

| | |
|---|---|
| A | AH |
| A'S | EY Z |
| A(2) | EY |
| A. | EY |
| A.'S | EY Z |
| A.S | EY Z |
| AAA | T R IH 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 |
| … | |

100K~1M words!

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

# 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**

# 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** **(Example by Sathvik)**
- Potential to outperform conventional ASR by **optimizing the entire network** with a single objective function

# Japanese is **not** an ASR friendly 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**

# Japanese is **not** an ASR friendly 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**

# Japanese is **not** an ASR friendly 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**

# Japanese is **not** an ASR friendly 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**

# Japanese is **not** an ASR friendly 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 a tokenizer
  - Additional tool
  - Require a dictionary

# MeCab/Unidic Demonstration

Enter Japanese sentence:

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

Run Reset

# MeCab/Unidic Demonstration

### Input Text

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

### MeCab Segmentation

| | | | | |
|---|---|---|---|---|
| 二 | フタ | フタ | 二 | 名詞-数詞 |
| つ | ツ | ツ | つ | 接尾辞-名詞的-助数詞 |
| 目 | メ | メ | 目 | 接尾辞-名詞的-一般 |
| の | ノ | ノ | の | 助詞-格助詞 |
| 要因 | ヨーイン | ヨウイン | 要因 | 名詞-普通名詞-一般 |
| は | ワ | ハ | は | 助詞-係助詞 |
| 計算 | ケーサン | ケイサン | 計算 | 名詞-普通名詞-サ変可能 |
| 機 | キ | キ | 機 | 名詞-普通名詞-助数詞可能 |
| 資源 | シゲン | シゲン | 資源 | 名詞-普通名詞-一般 |
| ・ | | | ・ | 補助記号-一般 |
| 音声 | オンセー | オンセイ | 音声 | 名詞-普通名詞-一般 |
| データ | データ | データ | データ-data | 名詞-普通名詞-一般 |
| の | ノ | ノ | の | 助詞-格助詞 |
| 増加 | ゾーカ | ゾウカ | 増加 | 名詞-普通名詞-サ変可能 |
| 及び | オヨビ | オヨビ | 及び | 接続詞 |
| Kaldi | Kaldi | Kaldi | Kaldi | 名詞-普通名詞-一般 |

# My attempt (2016)

- Japanese NLP/ASR: always go through a tokenizer
  - Additional tool
  - Require a dictionary
- **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)

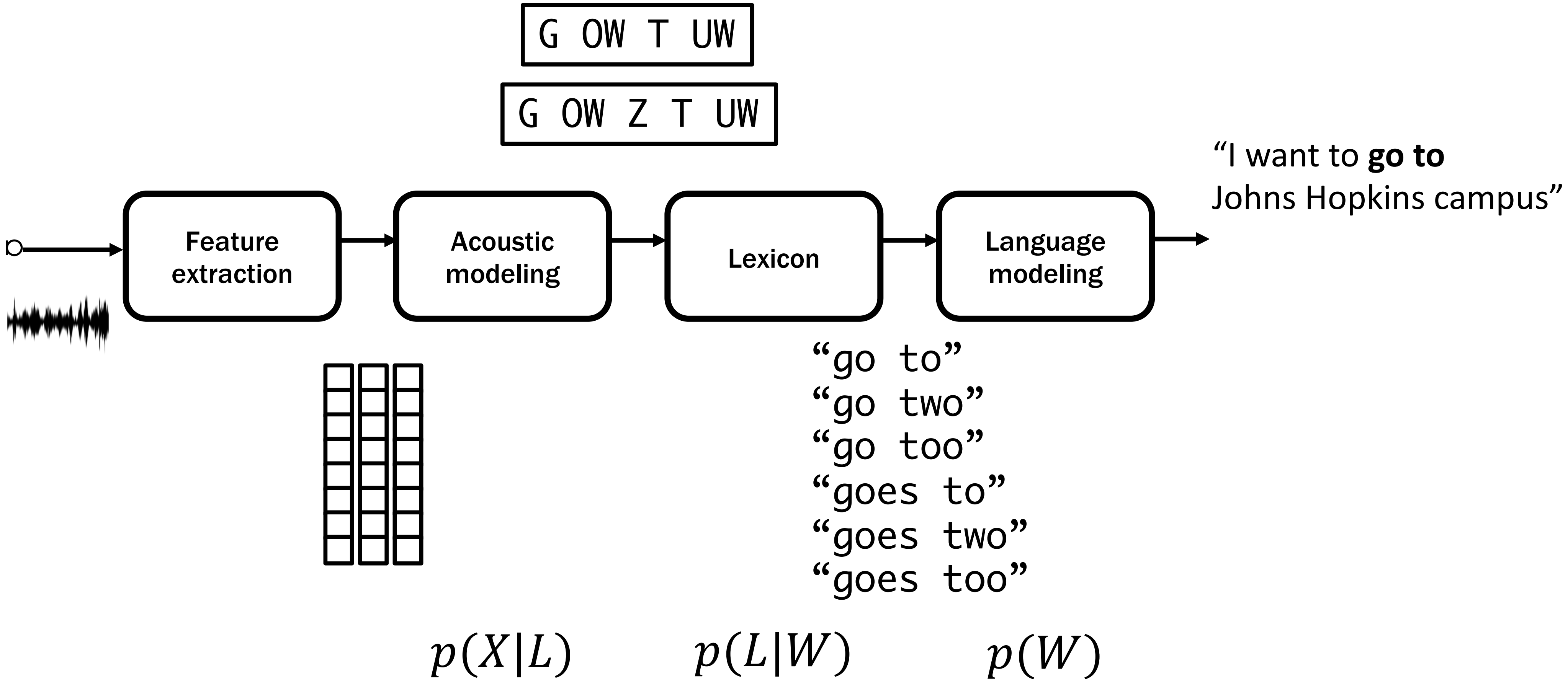
# Multilingual e2e ASR

- Given the Japanese ASR experience, I thought that e2e ASR can handle mixed languages with a single architecture

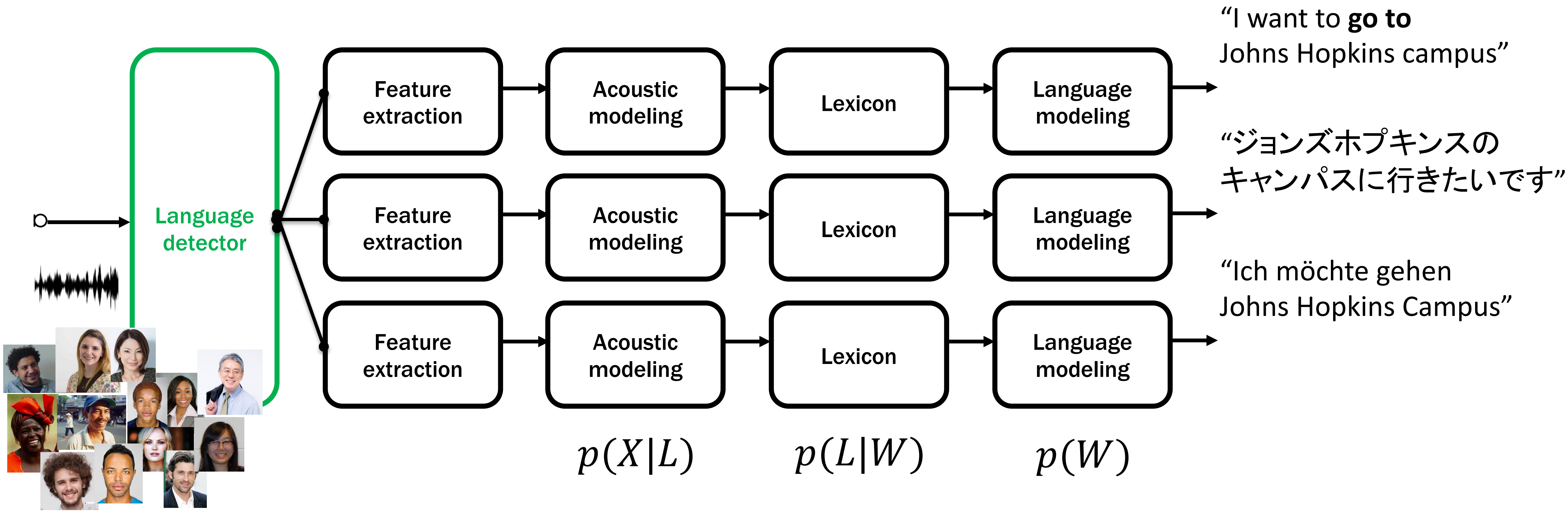
➡ Multilingual e2e ASR (2017)

➡ Multilingual code-switching e2e ASR (2018)

# Speech recognition pipeline



# Multilingual speech recognition pipeline



# Multilingual speech recognition pipeline

**End-to-End Neural Network**

"I want to **go to**
Johns Hopkins campus"

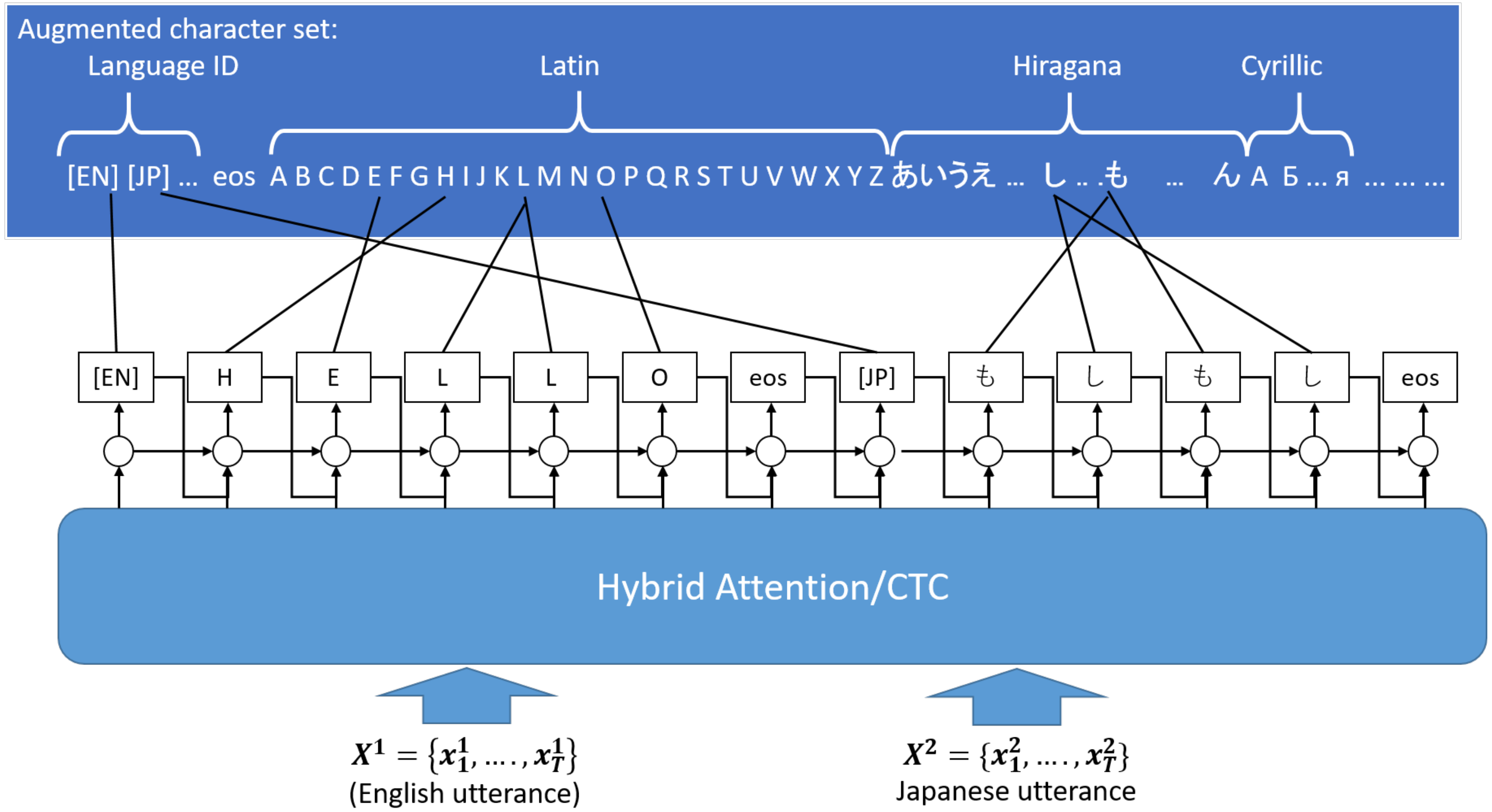
"ジョンズホプキンスの
キャンパスに行きたいです"

"Ich möchte gehen
Johns Hopkins Campus"

# 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**



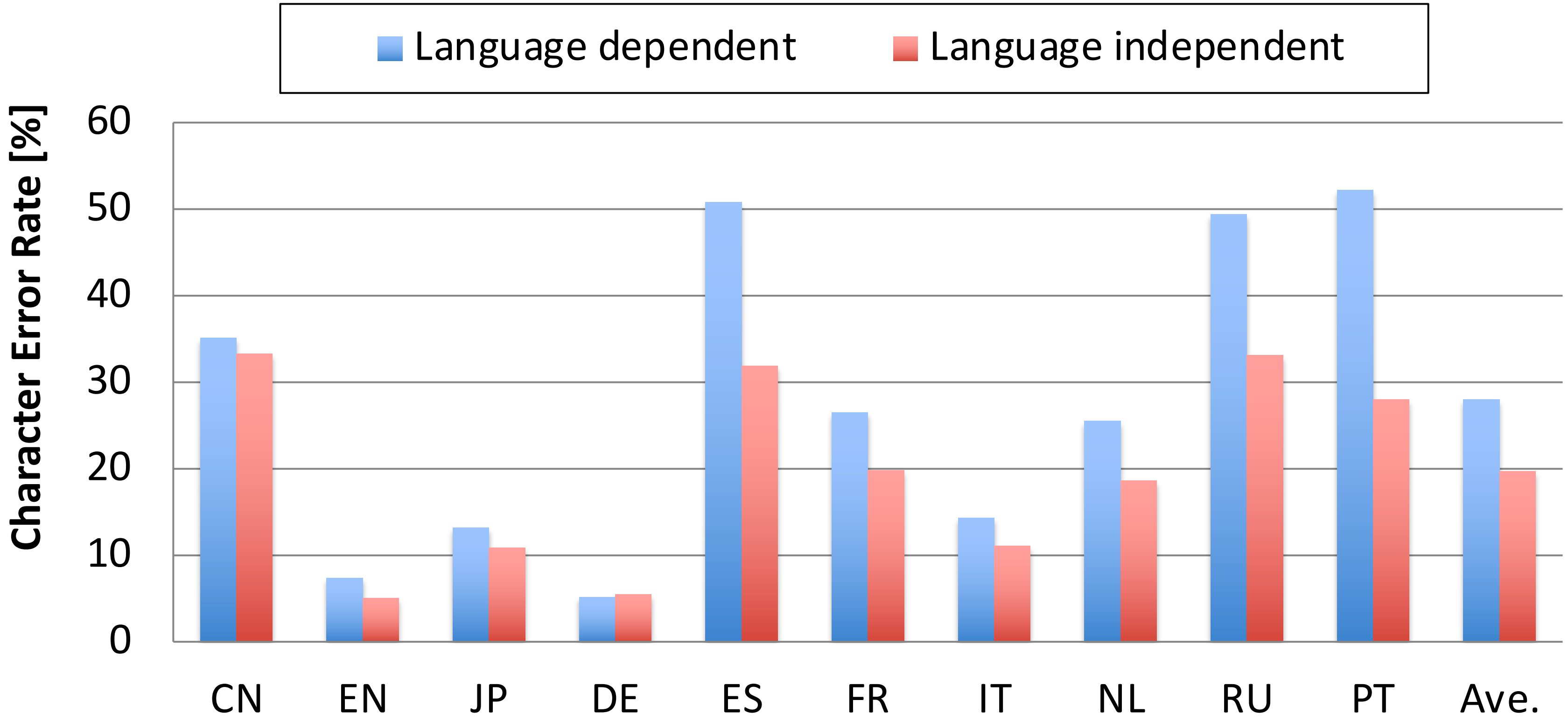

Include all language characters and language ID for final softmax to accept all target languages


DICTIONARY
漢字


DICTIO
NARY

# ASR performance for 10 languages

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



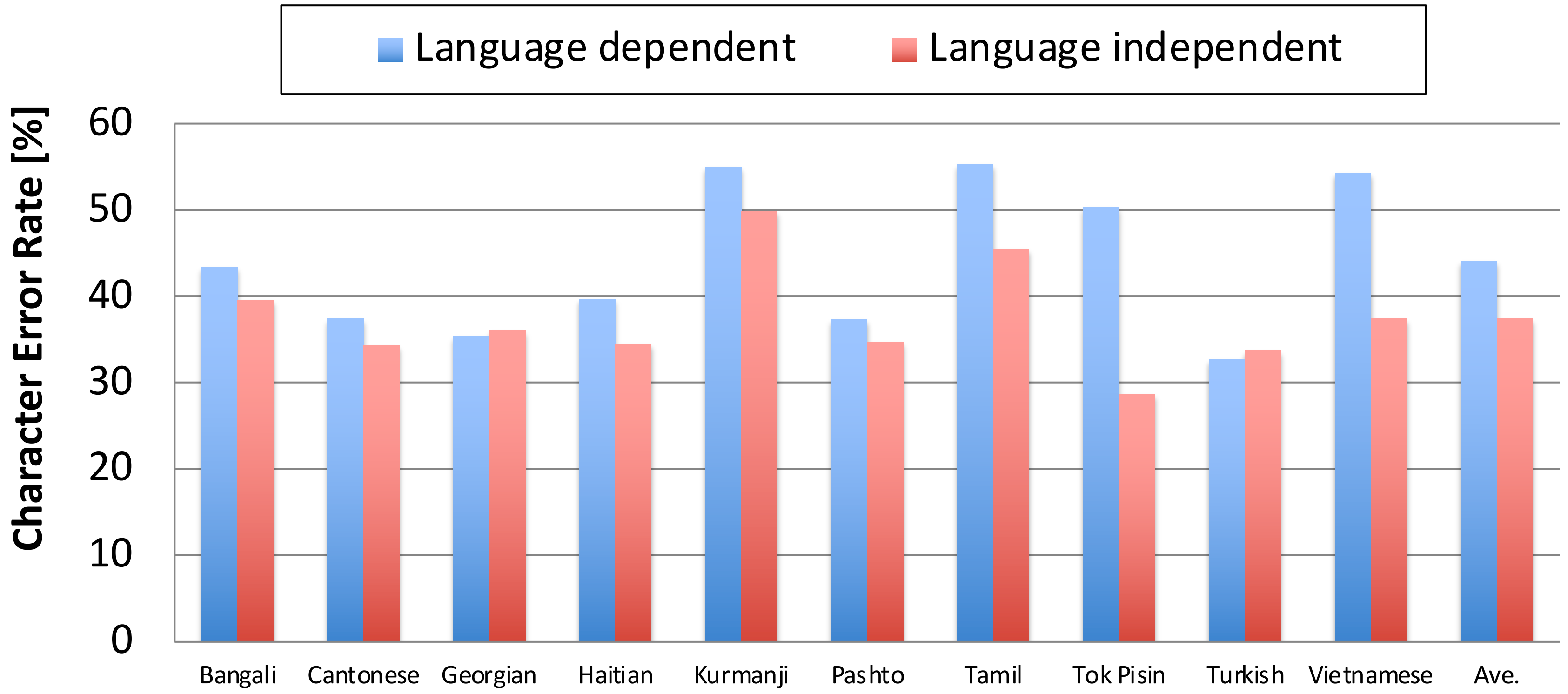

你好
Hello
こんにちは
Hallo
Hola
Bonjour
Ciao
Hallo
Привет
Olá

# Language recognition performance

| | | 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 |

# ASR performance for **low-resource** 10 languages

- Comparison with language dependent systems

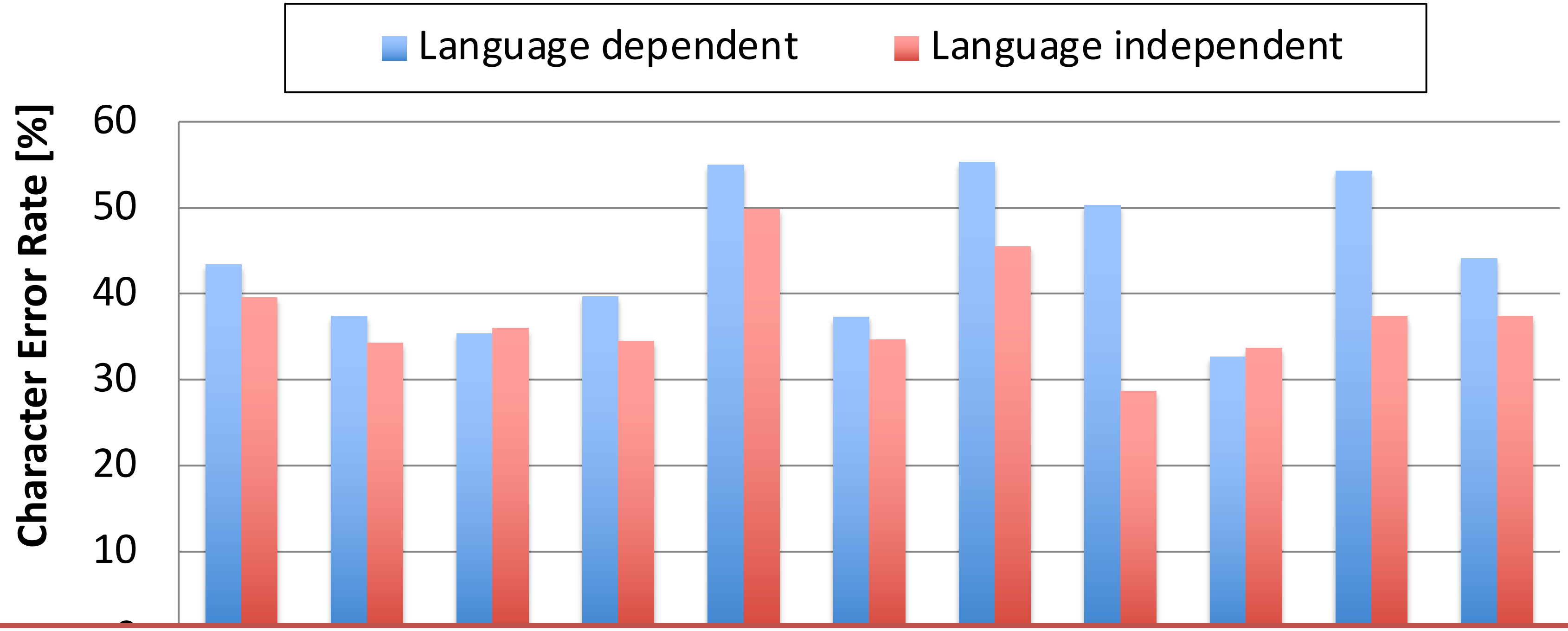



হ্যালো
你好
გამარჯობა
hello
???
سلام
வணக்கம்
???
Merhaba
xin chào

Some MUCS languages (e.g., Tamil) is included in this work

# ASR performance for **low-resource** 10 languages

- Comparison with language dependent systems




হ্যালো
你好
გამარჯობა
hello
???
سلام
வணக்கம்
???
Merhaba
xin chào

~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)

| ID | a04m0051_0.352274410405 |
| --- | --- |
| | REF: [DE] bisher sind diese personen rundherum versorgt worden [EN] u. s. exports rose in the month but not nearly as much as imports<br>ASR: [DE] bisher sind diese personen rundherum versorgt worden [EN] u. s. exports rose in the month but not nearly as much as imports |

| ID | csj-eval:s00m0070-0242356-0244956:voxforge-et-fr:mirage59-20120206-njp-fr-sb-570 |
| --- | --- |
| | REF: [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<br>ASR: [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 |

| ID | voxforge-et-pt:insinfo-20120622-orb-209:voxforge-et-de:guenter-20140127-usn-de5-069:csj-eval:a01m0110-0243648-0247512 |
| --- | --- |
| | REF: [PT] segunda feira [DE] das gilt natürlich auch für bestehende verträge [JP] えー同一人物による異なるメッセージを示しております<br>ASR: [PT] segunda feira [DE] das gilt natürlich auch für bestehende verträge [JP] えー同一人物による異なるメッセージを示しております |

# Timeline

## Shinji's personal experience for end-to-end speech processing

**2017**

**Open source**

- share the knowhow
- Kaldi-style
- Jelinek workshop

# ESPnet

## 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**

I personally didn't like pre-training fine-tuning strategies (but I changed my mind)

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

# 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 **(Part II by Pengcheng and Sathvik covers more examples)**
- 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 transducder
  - Warptransducer, beam search (label-synchronous) decoding
- Use of language models
  - Combination of RNNLM/n-gram trained with external text data (shallow fusion)
- **Part II (by Pengcheng) covers more concrete descriptions about the recipe and new functions**

# 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?

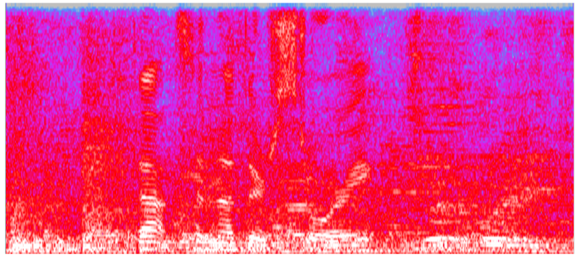
# High-level benefit of e2e neural network

- **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, opennmtpy, lingvo, speechbraing, etc. etc., in an unified manner

# Automatic speech recognition (ASR)

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


$$X = (x(l) \in \mathbb{Z} | l = 1, \dots, L)$$
$$L = 43263$$



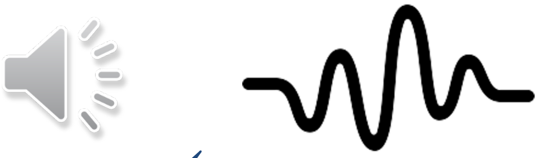
$$X = (\mathbf{x}_t \in \mathbb{R}^D | t = 1, \dots, T)$$
$$T = 268$$

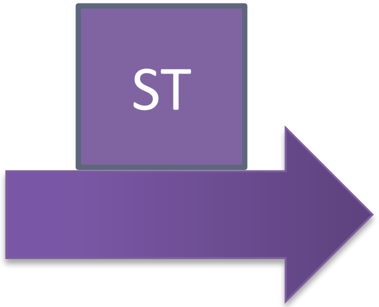
"That's another story"

$$W = (\mathbf{w}_n \in \mathcal{V} | n = 1, \dots, N)$$
$$N = 18$$

# Speech to text translation (ST)

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

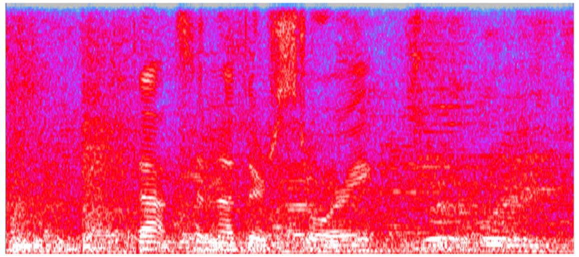





$$X = (x(l) \in \mathbb{Z} | l = 1, \ldots, L)$$

$$L = 43263$$

That's another story



$$X = (\mathbf{x}_t \in \mathbb{R}^D | t = 1, \ldots, T)$$

$$T = 268$$

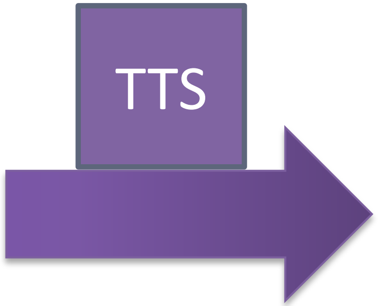
"Das ist eine andere Geschichte"

$$W = (\mathbf{w}_n \in \mathcal{V} | n = 1, \ldots, N)$$

$N$=31

# Text to speech (TTS)

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



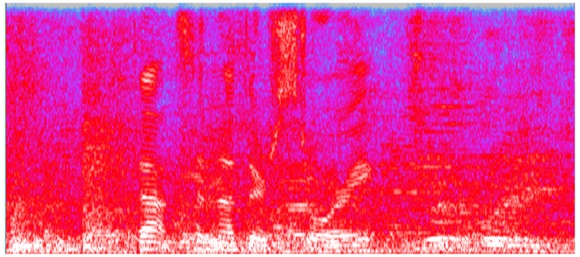

$$X = (x(l) \in \mathbb{Z} | l = 1, \ldots, L)$$

$$L = 43263$$

"That's another story"

$$W = (\mathbf{w}_n \in \mathcal{V} | n = 1, \ldots, N)$$

$$N = 18$$

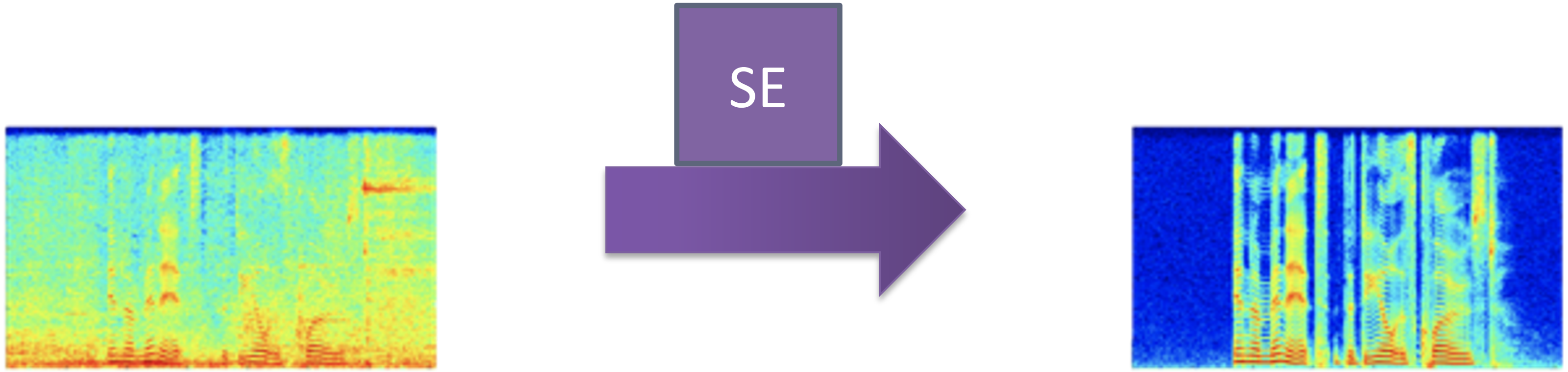


$$X = (\mathbf{x}_t \in \mathbb{R}^D | t = 1, \ldots, T)$$

$$T = 268$$

# Speech enhancement (SE)

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




$$X = (\mathbf{x}_t \in \mathbb{R}^D | t = 1, \ldots, T)$$
$$T = 268$$

$$X' = (\mathbf{x}'_t \in \mathbb{R}^D | t = 1, \ldots, T)$$
$$T = 268$$

# All of the problems

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

# Unified view with sequence to sequence

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

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

  - ASR: *X* = Speech, *Y* = Text
  - TTS: *X* = Text, *Y* = Speech
  - ST: *X* = Speech (EN), *Y* = Text (JP)
  - Speech Enhancement: *X* = Noisy speech, *Y* = Clean speech
- Mapping function $f(\cdot)$
  - Sequence to sequence (seq2seq) function
  - ASR as an example

# Seq2seq end-to-end ASR

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

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

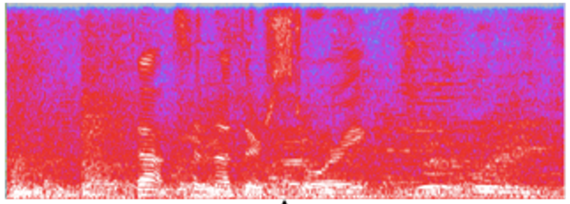
- Target speech processing problems: find a mapping function from *sequence* to *sequence* (**unification**)

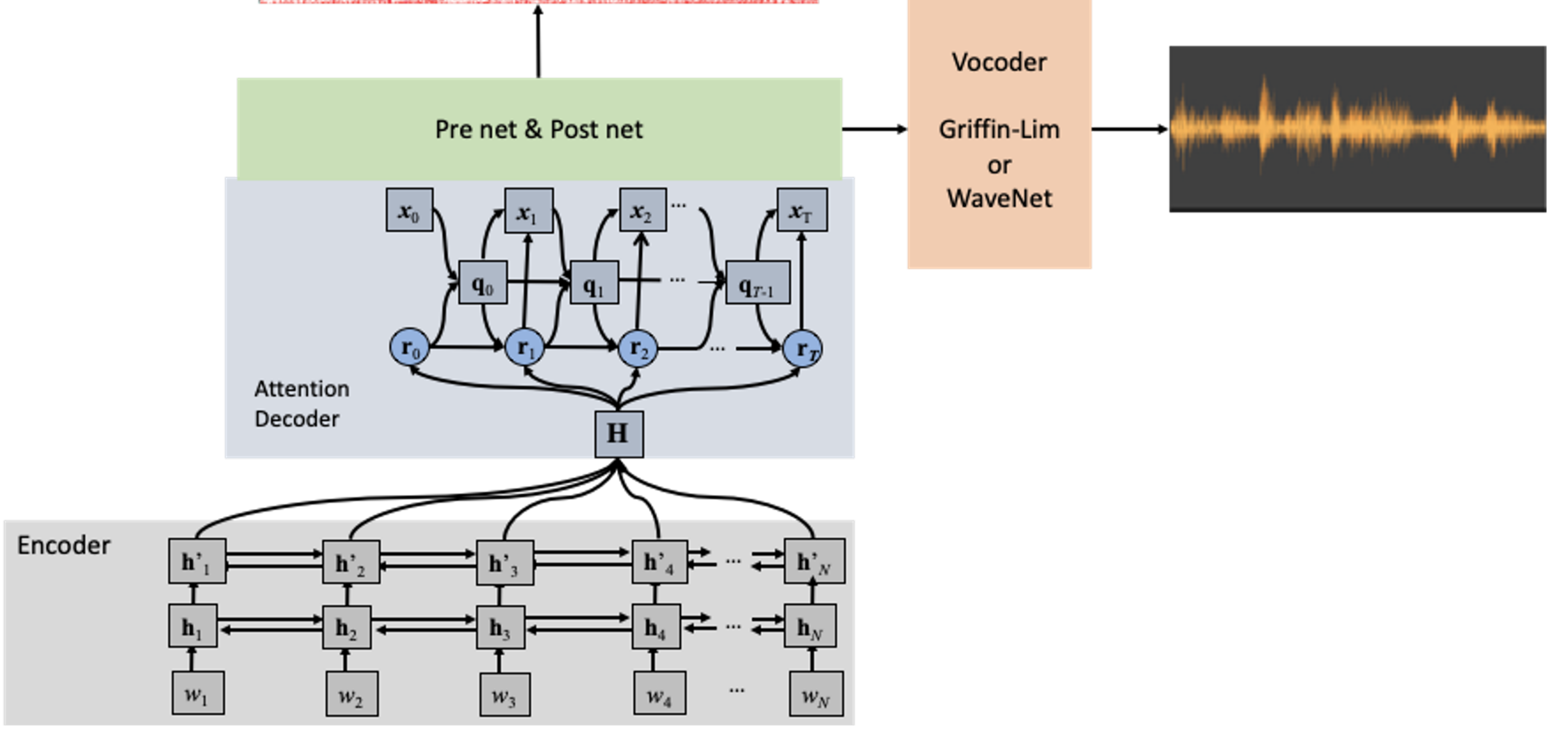
$$\boxed{X = (x_1, x_2, \cdots, x_T) \xrightarrow{f} Y = (y_1, y_2, \cdots, y_N)}$$

  - 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 → Unified software design

We design a new speech processing toolkit based on

$$X = (x_1, x_2, \cdots, x_T) \qquad Y = (y_1, y_2, \cdots, y_N)$$

**ESPnet**

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

$$f(\cdot)$$

# Unified view → Unified software design

We design a new speech processing toolkit based on

$X = (x_1, x_2, \cdots, x_T)$

Speech
Text
English Speech
Noisy Speech

**ESPnet**

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

$f(\cdot)$

$Y = (y_1, y_2, \cdots, y_N)$

Text
Speech
German Text
Clean Speech

# Unified view → 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

ESPnet: End-to-end speech processing toolkit

CTC
Attention
Joint C/A
RNN-T
Transformer

$f(\cdot)$

# Unified view → 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

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

- Many speech processing applications can be **unified** based on seq2seq
- Again, **Espresso, Nemo, Fairseq, Lingvo, SpeechBrain** 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

https://github.com/nttcslab-sp/dnn_wpe, [Subramanian'19]

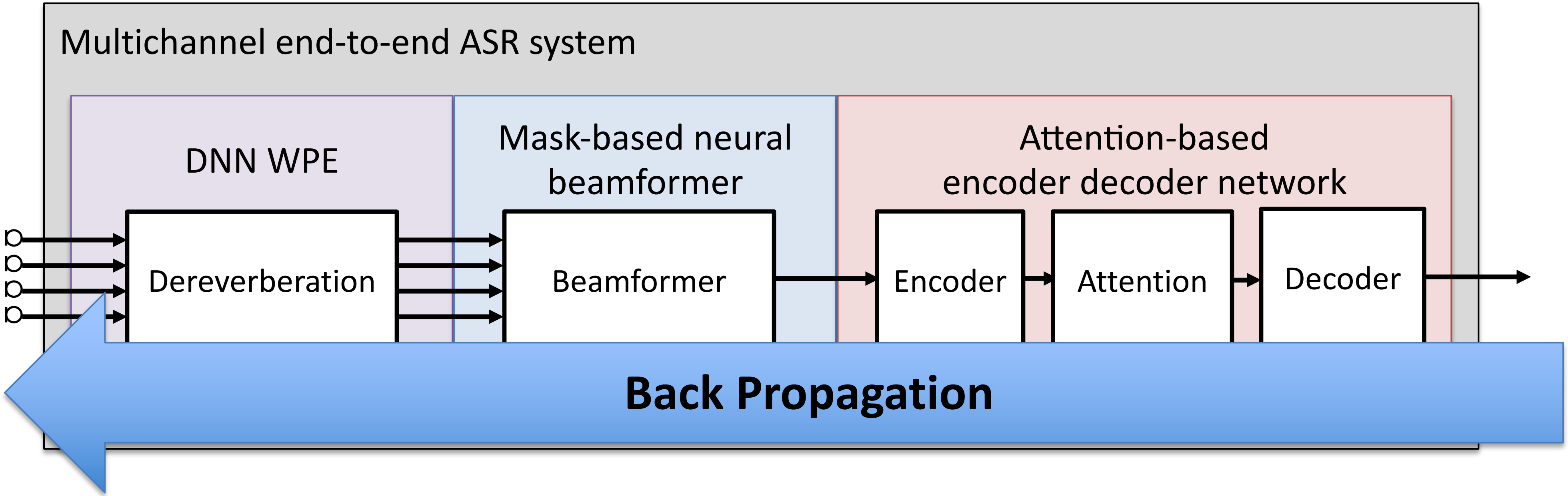
- Multichannel end-to-end ASR framework
  - integrates entire process of **speech dereverberation (SD), beamforming (SB)**and **speech recognition (SR)**, by single neural-network-based architecture

↓

**SD : DNN-based weighted prediction error (DNN-WPE) [Kinoshita et al., 2016]**

**SB : Mask-based neural beamformer [Erdogan et al., 2016]**

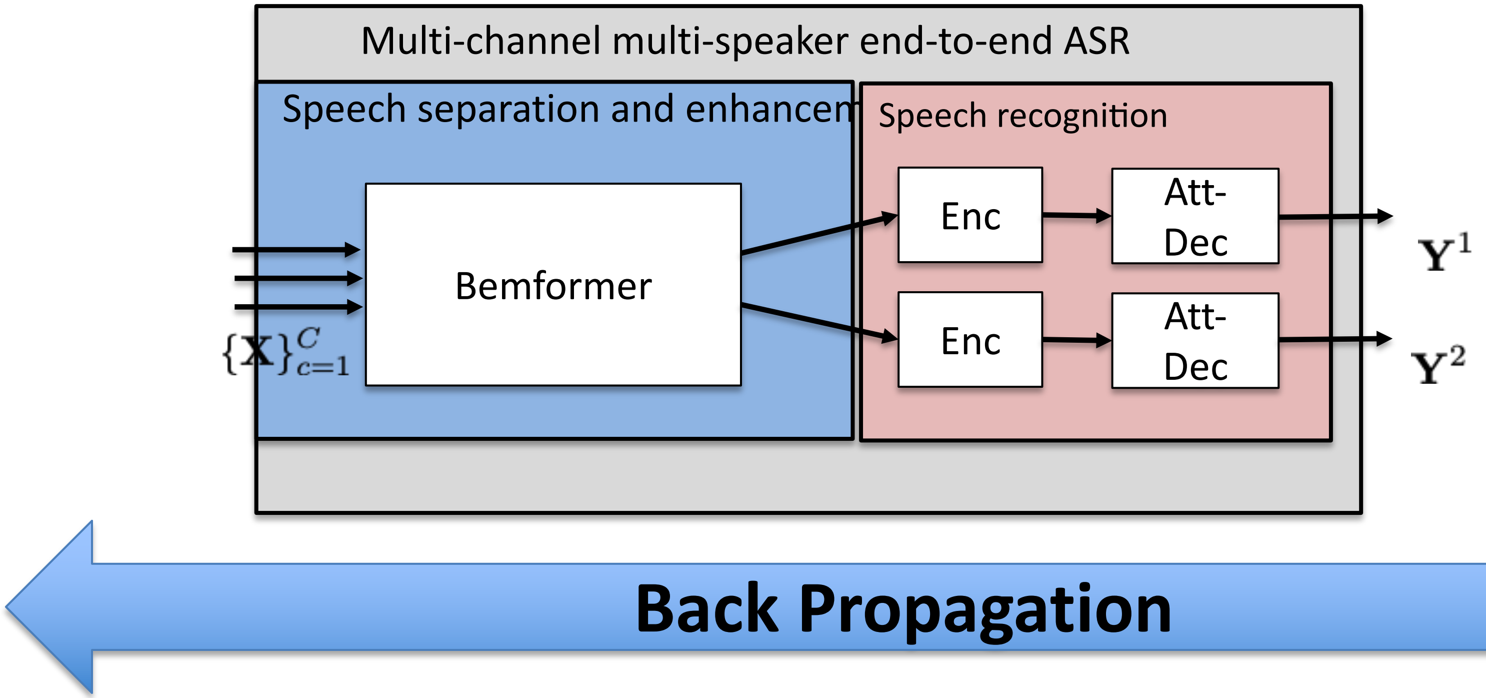
**SR : Attention-based encoder-decoder network [Chorowski et al., 2014]**



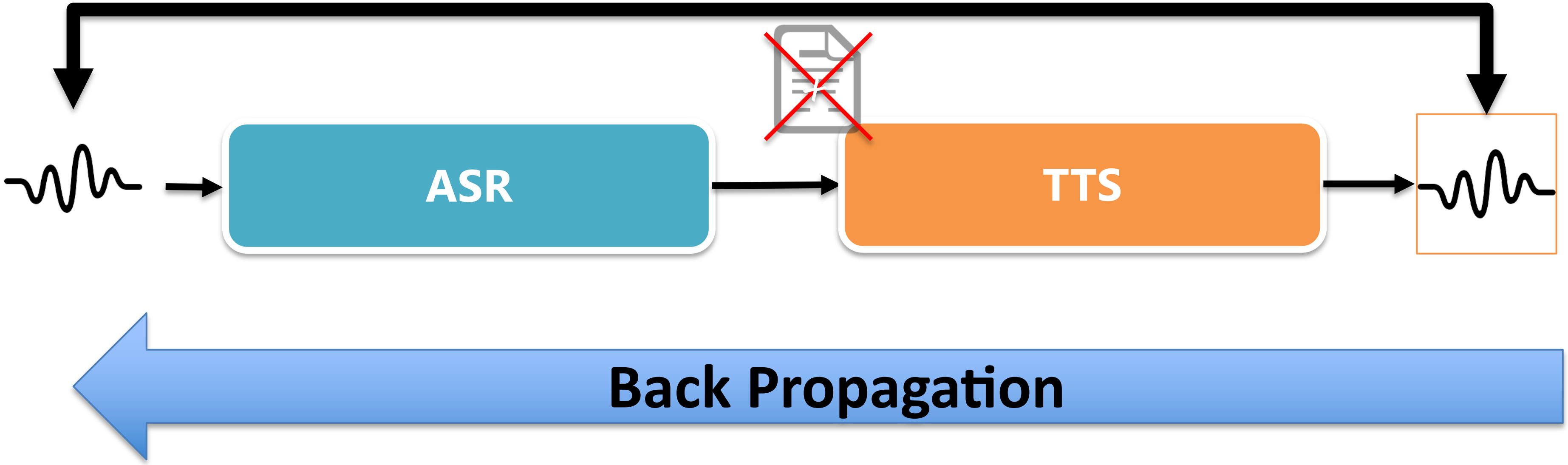
# Beamforming + separation + ASR
## [Xuankai Chang., 2019, ASRU]

- 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**



# 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)

| Toolkit | dev_clean | dev_other | test_clean | test_other |
|---|---|---|---|---|
| Facebook wav2letter++ | 3.1 | 10.1 | 3.4 | 11.2 |
| RWTH RASR | 2.9 | 8.8 | 3.1 | 9.8 |
| Nvidia Jasper | 2.6 | 7.6 | 2.8 | 7.8 |
| Google SpecAug. | N/A | N/A | **2.5** | 5.8 |
| | | | | |

- Very impressive results by Google

# Experiments (~ **1000** hours) Librispeech

| Toolkit | dev_clean | dev_other | test_clean | test_other |
|---|---|---|---|---|
| Facebook wav2letter++ | 3.1 | 10.1 | 3.4 | 11.2 |
| RWTH RASR | 2.9 | 8.8 | 3.1 | 9.8 |
| Nvidia Jasper | 2.6 | 7.6 | 2.8 | 7.8 |
| Google SpecAug. | N/A | N/A | **2.5** | 5.8 |
| **ESPnet** | **2.2** | **5.6** | 2.6 | **5.7** |

- **Reached Google's best performance by community-driven efforts (on September 2019)**

GAFAM


GAFAM
ESPnet

Good example of “Collapetition”

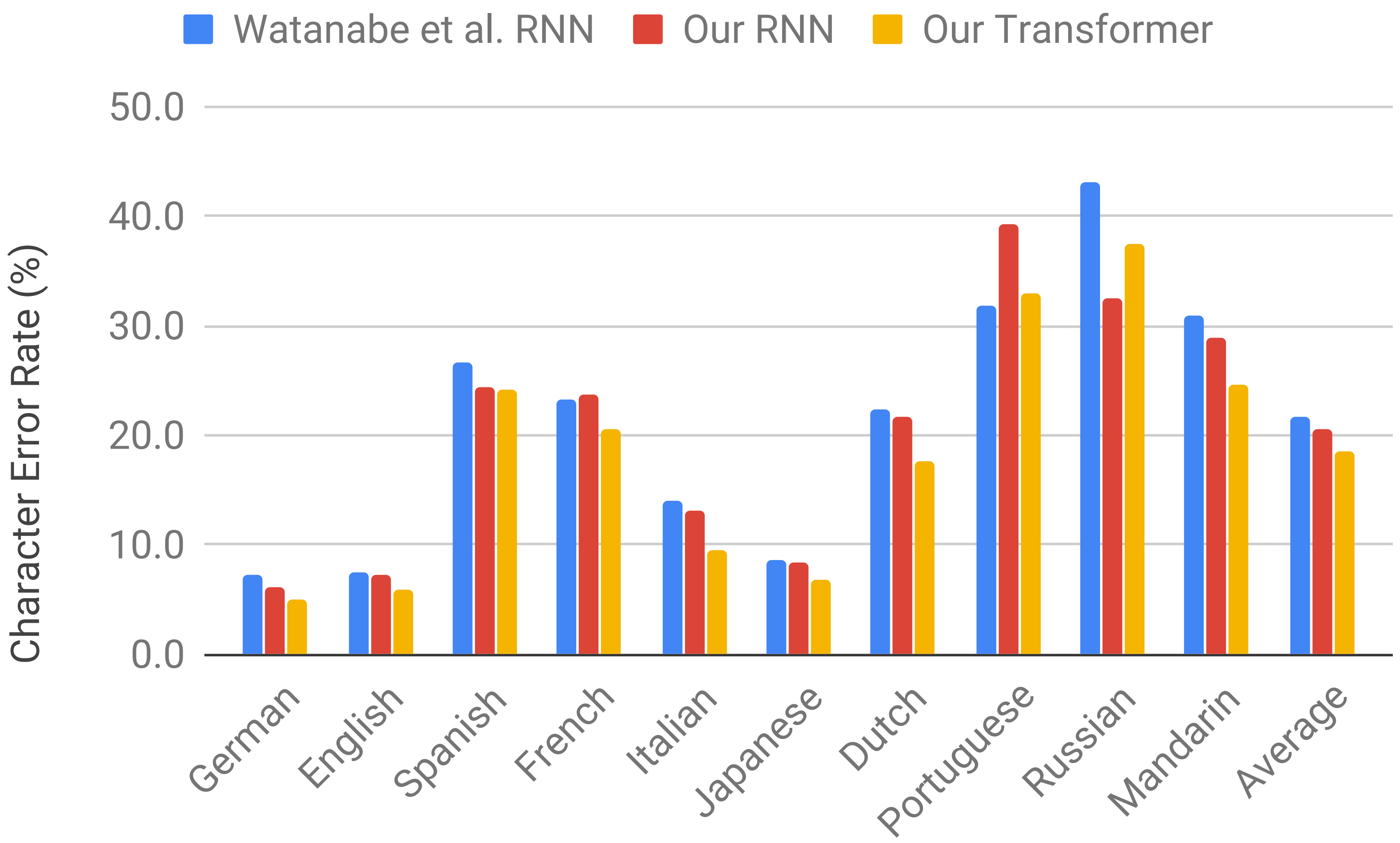
= Collaboration + Competition

# Experiments (~ **1000** hours) Librispeech

| Toolkit | dev_clean | dev_other | test_clean | test_other |
|---|---|---|---|---|
| Facebook wav2letter++ | 3.1 | 10.1 | 3.4 | 11.2 |
| RWTH RASR | 2.9 | 8.8 | 3.1 | 9.8 |
| Nvidia Jasper | 2.6 | 7.6 | 2.8 | 7.8 |
| Google SpecAug. | N/A | N/A | 2.5 | 5.8 |
| **ESPnet** | 2.2 | 5.6 | 2.6 | 5.7 |
| MS Semantic Mask (ESPnet) | **2.1** | **5.3** | 2.4 | **5.4** |
| Facebook wav2letter Transformer | **2.1** | **5.3** | **2.3** | 5.6 |

- Just after a few months… **And more results in Part II by Pengcheng**

# Transformer is powerful for multilingual ASR



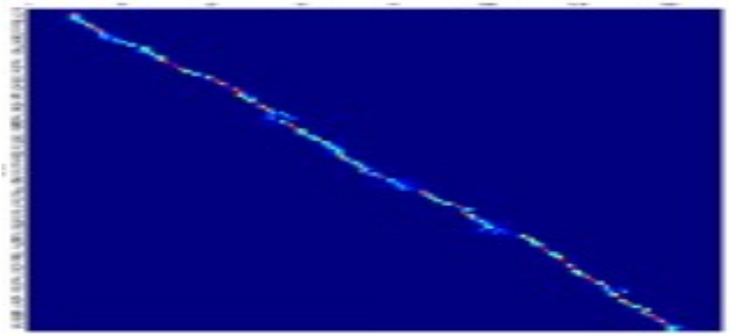

One of the most stable and biggest gains compared with other multilingual ASR techniques

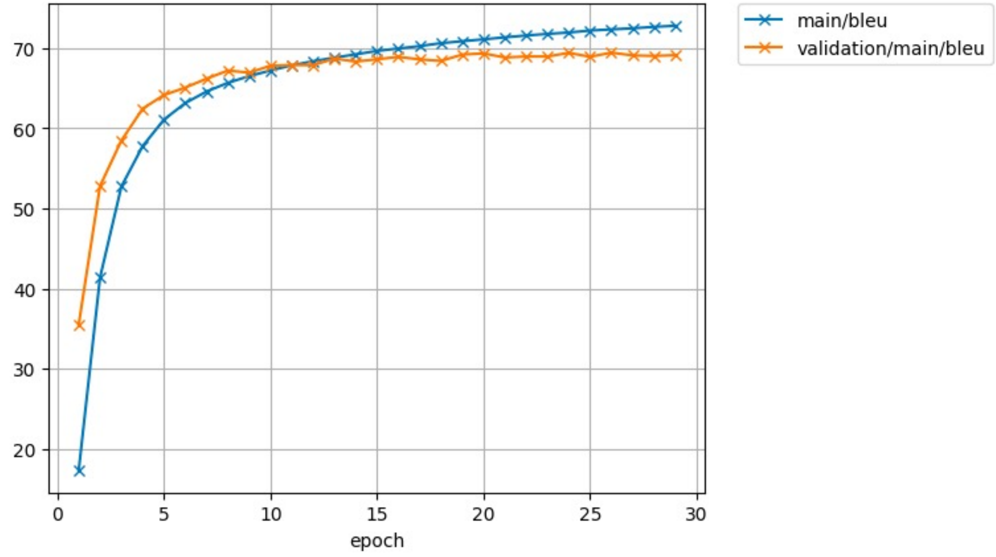
# FAQ (before transformer)

- How to debug attention-based encoder/decoder?
- Please check

  **Attention pattern!**

  **Learning curves!**
- It gives you a lot of intuitive information!





# FAQ (after transformer)

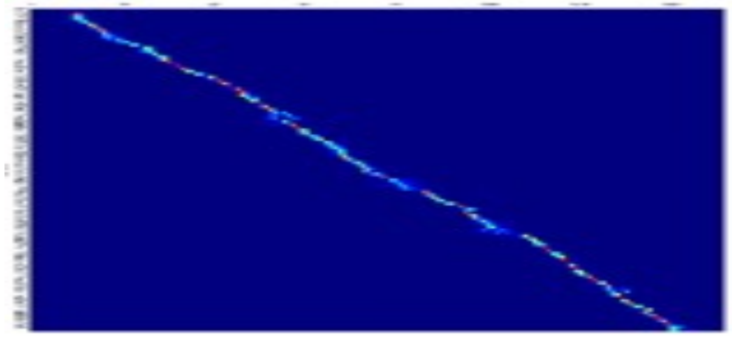
- How to debug attention-based encoder/decoder?

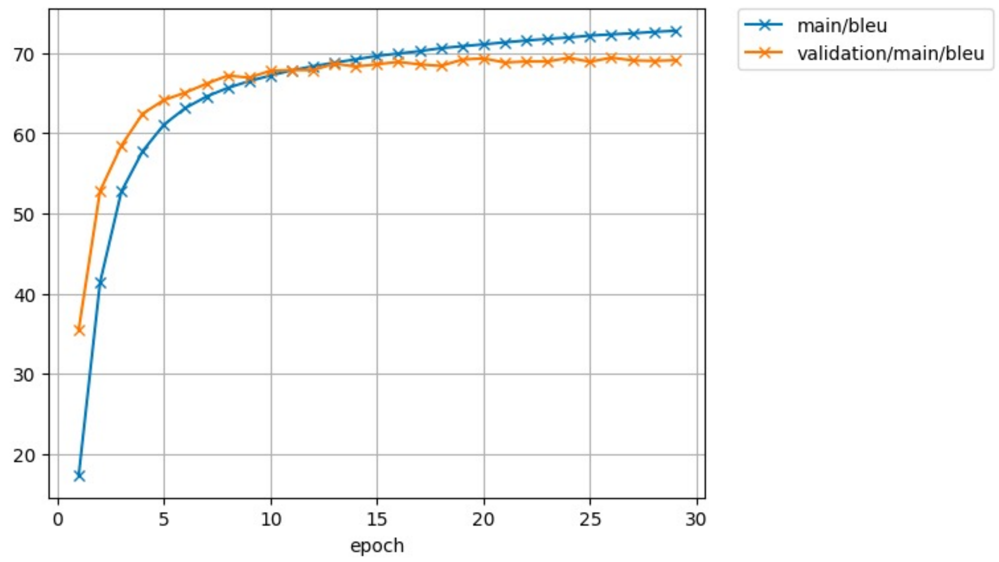
- Please check

  **Attention pattern (including self attention)!**

  **Learning curves!**

- It gives you a lot of intuitive information!
- **Tune optimizers!**





# Timeline

## Shinji's personal experience for end-to-end speech processing

| -2015 | 2016 | 2017 | 2018 | 2019 | 2020 |
|---|---|---|---|---|---|
| **First impression** | **Initial implementation** | **Open source** | **ASR+X** | **Improvement** | |
| - No more conditional independence assumption<br>- DNN tool blossom | - CTC/attention hybrid<br>- Japanese e2e -> multilingual. | - share the knowhow<br>- Kaldi-style<br>- Jelinek workshop | - TTS<br>- Speech translation<br>- Speech enhancement + ASR | - Transformer<br>- Open source acceleration | |

# What's next?

- **Non autoregressive ASR**
- **Time-domain processing** (real end-to-end including feature extraction and speech enhancement)
- **Differentiable WFST**

- **New architecture**
  - **Conformer**
- **Self-supervised training**
  - **Wav2vec2, HuBert**

**By Pengcheng in Part II**

# Overview of today's tutorial

- **5pm to 6pm:** part I presentation by Shinji
  - Introduction of end-to-end ASR and ESPnet
- **6pm to 6:30 pm:** Q&A for part I and break
- **6:30pm to 7pm:** part II presentation by Pengcheng
  - Advanced techniques in ESPnet
- **7pm to 7:15 pm:** part II espnet mucs recipe by Sathvik
  - espnet mucs recipe, and demo
- **7:15pm to 7:30pm:** summary and Q&A by Shinji



Carnegie Mellon University
Language Technologies Institute

# Introduction of ESPnet, End-to-End Speech Processing Toolkit

**Shinji Watanabe**
Carnegie Mellon University
**Pengcheng Guo**
Northwestern Polytechnical University
**Sathvik Udupa**
Indian Institute of Science

MUCS 2021: MUltilingual and Code-Switching ASR Challenges for Low Resource Indian Languages
**12-13 August 2021**

# Timeline of ESPnet

## -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-

# Timeline of ESPnet

## -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-

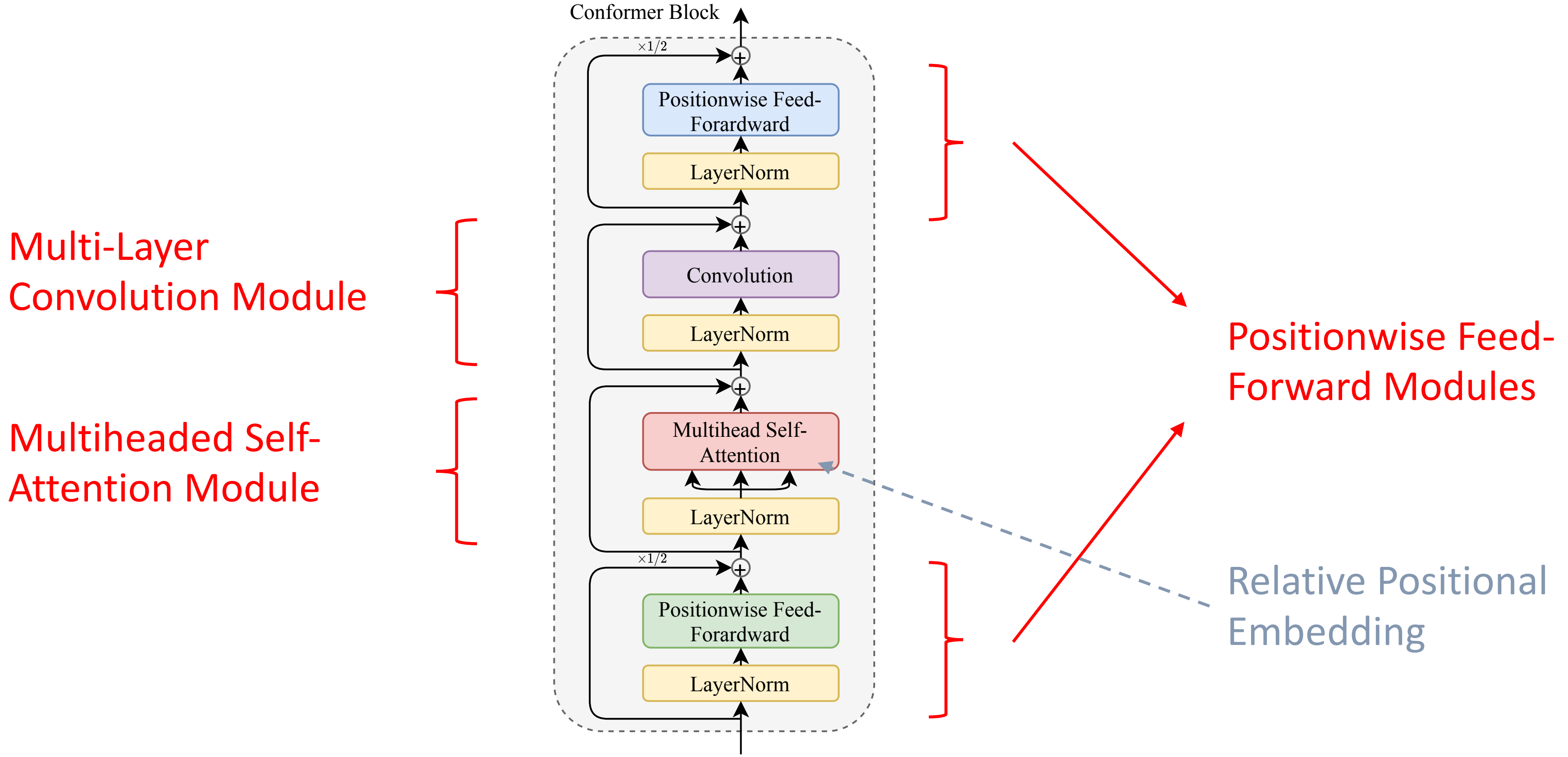
**More intriguing Ideas**

- Conformer
- Self-supervised pretrained models

# Conformer: Covolution-augmented Transformer

[Gulati+ 2020]

- Combine the multi-headed self-attention layer with the convolutional layer in the encoder



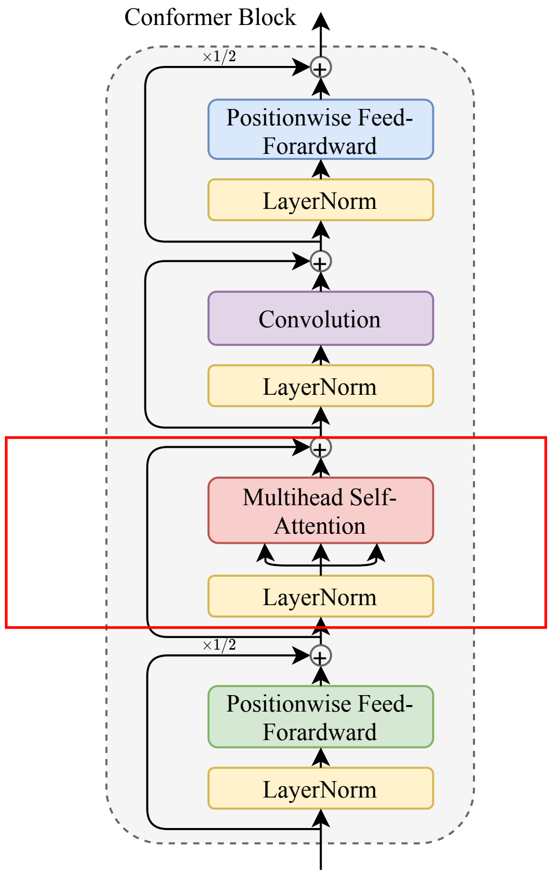
# Conformer: Covolution-augmented Transformer

[Gulati+ 2020]

- Multiheaded self-attention module
  - Aim to learn the global context

$$\text{MHSA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)\mathbf{W}^{\mathbf{O}}$$

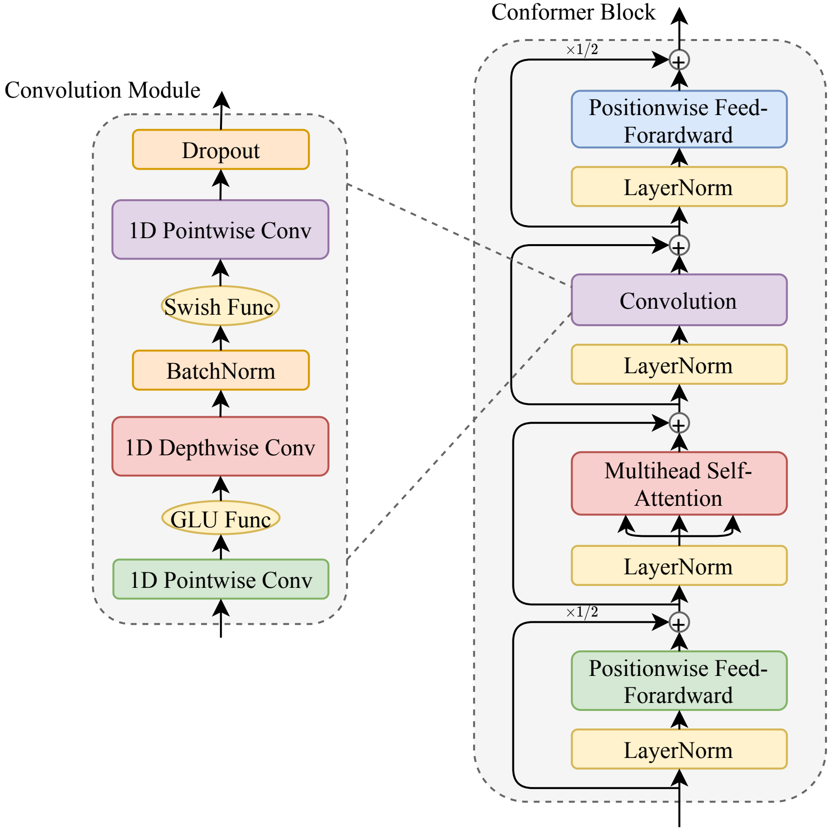
$$\text{head}_{\boldsymbol{i}} = \text{Attention}(\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h)$$



# Conformer: Covolution-augmented Transformer

[Gulati+ 2020]

- Multi-layer convolution module
  - Efficiently capture the local correlations



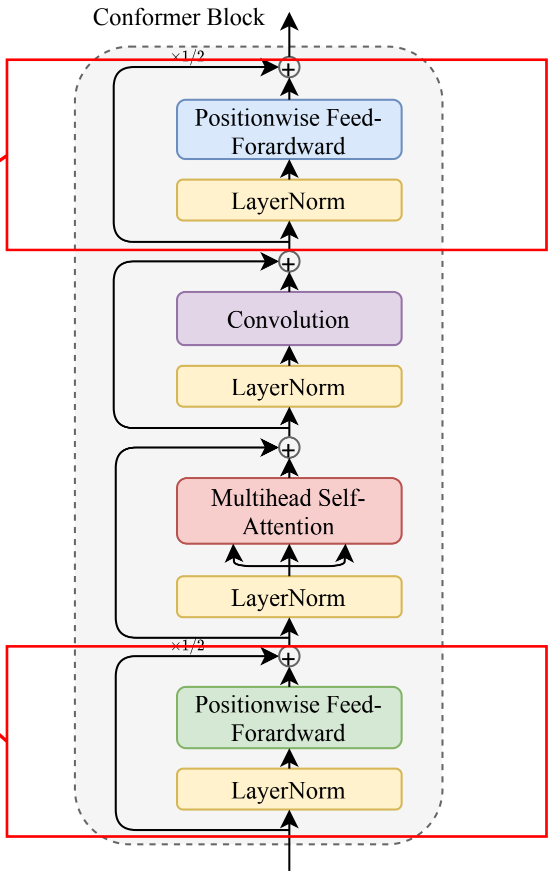
# Conformer: Covolution-augmented Transformer

[Gulati+ 2020]

- Pointwise feed-forward module
  – Consists of two linear transformations with a ReLU activation in the between.

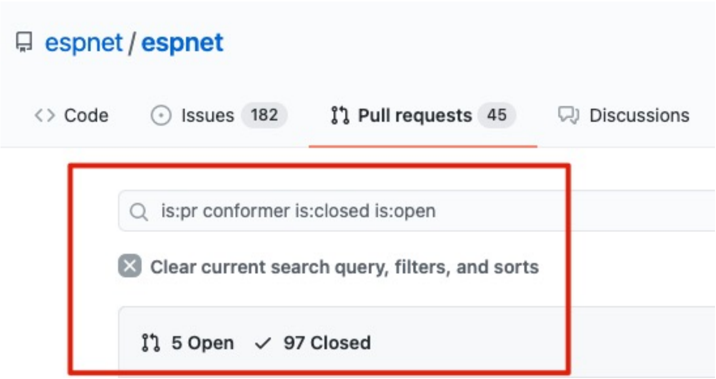
Macaron-Net Style

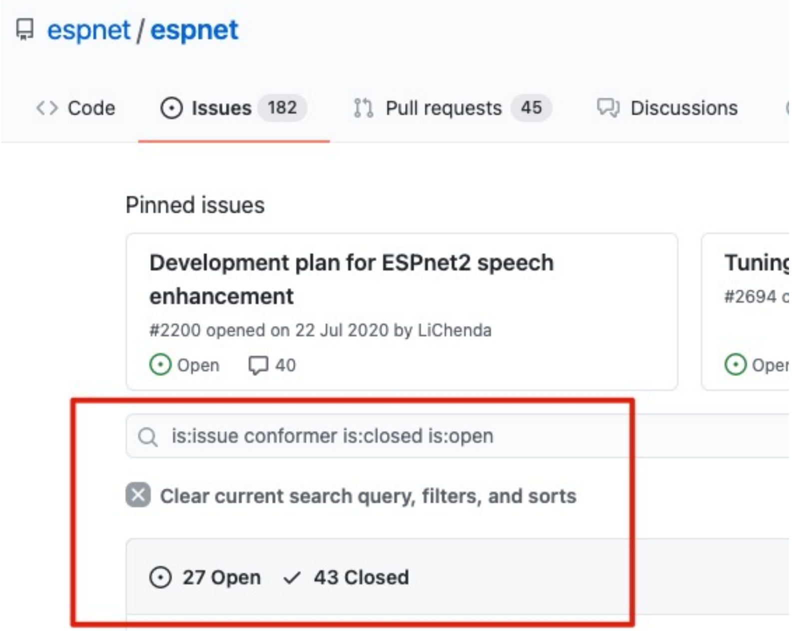
$$\text{FFN}(\mathbf{X}) = \mathbf{W}_2\text{ReLU}(\mathbf{W}_1\mathbf{X} + b_1) + b_2$$



# How to Implement the Conformer in ESPnet

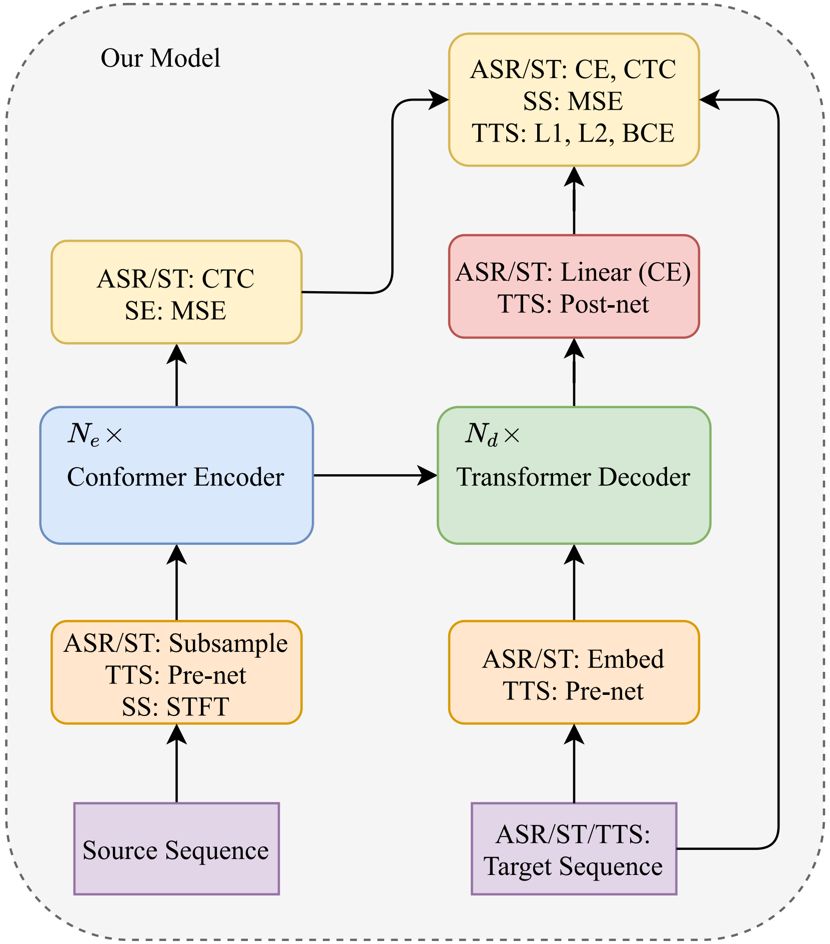
- Initial implementation (Jun 2020)
  - GLU activation takes 2 tensors for the element-wise product
    - Increase the channel dimension?
    - Use 2 different 1D Pointwise Conv layers?
  - The usage of relative positional embeddings
    - Share the hyper-parameters or not?
  - Can't reproduce Google's results, etc.
- First Pull Request (Jul 2020)



# Conformer Model in ESPnet [Watanabe+ 2018, Guo+ 2020]

- Conformer Encoder + Transformer Decoder
  - 😄 Efficiently capture both global and local context in the encoder
  - 😄 Very good performance on various speech processing tasks (ASR, ST, TTS, etc.)
  - 😓 Off-line, slow inference



# ASR Experiments (178 hours Mandarin task)

- Character Error Rate (%) on AISHELL-1 corpus

| Models | dev | test |
|---|---|---|
| Kaldi Chain Model | N/A | 7.4 |
| Tsinghua CTC-CAT | N/A | 6.3 |
| Mobvoi U2 | N/A | 4.7 |
| ESPnet Transformer | 6.0 | 6.7 |
| **ESPnet Conformer** | **4.4** | **4.7** |

**Achieve the state-of-art results (on October 2020)**

# ASR Experiments (960 hours English task)

- Word Error Rate (%) on Librispeech corpus

| Toolkit | dev_clean | dev_other | test_clean | test_other |
|---|---|---|---|---|
| Kaldi Chain Model | 3.9 | 10.4 | 4.3 | 10.8 |
| Facebook wav2letter++ | 3.1 | 10.1 | 3.4 | 11.2 |
| RWTH RASR | 2.9 | 8.8 | 3.1 | 9.8 |
| Nvidia Jasper | 2.6 | 7.6 | 2.8 | 7.8 |
| Google SpecAug. | N/A | N/A | 2.5 | 5.8 |
| Google Conformer | 2.1 | **4.3** | **1.9** | **3.9** |
| **ESPnet Conformer** | **1.9** | 4.9 | 2.1 | 4.9 |

**Reached Google's best performance.**

# ASR Performance boosted by Conformer [Guo+ 2020]

- ASR performance was improved on 14/17 corpora
- Show better results on the multi-speaker WSJ-2mix task

| Dataset | Vocab | Metric | Evaluation Sets | Transformer | Conformer |
|---|---|---|---|---|---|
| AIDATATANG | Char | CER | dev / test | (†) 5.9 / 6.7 | **4.3 / 5.0** |
| AISHELL-1 | Char | CER | dev / test | (†) 6.0 / 6.7 | (*) **4.4 / 4.7** |
| AISHELL-2 | Char | CER | android / ios / mic | (†) 8.9 / 7.5 / 8.6 | **7.6 / 6.8 / 7.4** |
| AURORA4 | Char | WER | dev_0330 (A / B / C / D) | **3.3 / 6.0 / 4.5** / 10.6 | 4.3 / 6.0 / 5.4 / **9.3** |
| CSJ | Char | CER | eval{1, 2, 3} | (∗) 4.7 / 3.7 / 3.9 | (∗) **4.5 / 3.3 / 3.6** |
| CHiME4 | Char | WER | {dt05, et05}_{simu, real} | (†) 9.6 / 8.2 / 15.7 / 14.5 | **9.1 / 7.9 / 14.2 / 13.4** |
| Fisher-CallHome | BPE | WER | dev / dev2 / test / devtest / evltest | 22.1 / 21.5 / 19.9 / 38.1 / 38.2 | **21.5 / 21.1 / 19.4 / 37.4 / 37.5** |
| HKUST | Char | CER | dev | (†) 23.5 | (†) **22.2** |
| JSUT | Char | CER | our split | (†) 18.7 | **14.5** |
| LibriSpeech | BPE | WER | {dev, test}_{clean, other} | 2.1 / 5.3 / 2.5 / 5.5 | **1.9 / 4.9 / 2.1 / 4.9** |
| REVERB | Char | WER | et_{near, far} | (†) 13.1 / 15.4 | (†) **10.5 / 13.9** |
| Switchboard | BPE | WER | eval2000 (callhm / swbd) | 17.2 / 8.2 | **14.0 / 6.8** |
| TEDLIUM2 | BPE | WER | dev / test | 9.3 / 8.1 | **8.6 / 7.2** |
| TEDLIUM3 | BPE | WER | dev / test | 10.8 / 8.4 | **9.6 / 7.6** |
| VoxForge | Char | CER | our split | (§) 9.4 / 9.1 | (§) **8.7 / 8.2** |
| WSJ | BPE | WER | dev93/ eval92 | (‡) **7.4 / 4.9** | (‡) 7.7 / 5.3 |
| WSJ-2mix | Char | WER | tt | (§) 12.6 | (§) **11.7** |

# ASR Performance boosted by Conformer [Guo+ 2020]

- Achieve more than 15% rel. improvement on low-resource language corpora

| Dataset | Transformer | Conformer + Data Augmentation |
|---|---|---|
| Yoloxóchitl-Mixtec | 23.0 / 23.2 | **16.0 / 16.1** |
| Puebla-Nahuat | 27.9 / 26.0 | **23.5 / 21.7** |
| Commonvoice-Czech | 38.2 / 44.3 | **15.3 / 20.6** |
| Commonvoice-Welsh | 32.0 / 21.8 | **20.0 / 14.2** |
| Commonvoice-Russian | 22.0 / 27.3 | **6.9 / 8.5** |
| Commonvoice-Italian | 31.8 / 33.7 | **15.6 / 17.0** |
| Commonvoice-Persian | 8.5 / 10.2 | **1.4 / 2.1** |
| Commonvoice-Polish | 24.1 / 15.1 | **8.8 / 2.6** |

# ASR Performance boosted by Conformer [Guo+ 2020]

- Both Conformer-CTC and Conformer-Transducer show consistent improvement
- Conformer-CTC model even achieves competitive results over Transformer model w/ decoder
- More results: Boyer et al. "A Study of Transducer based End-to-End ASR with ESPnet: Architecture, Auxiliary Loss, and Decoding Strategies"

CER/WER results of pure CTC models.

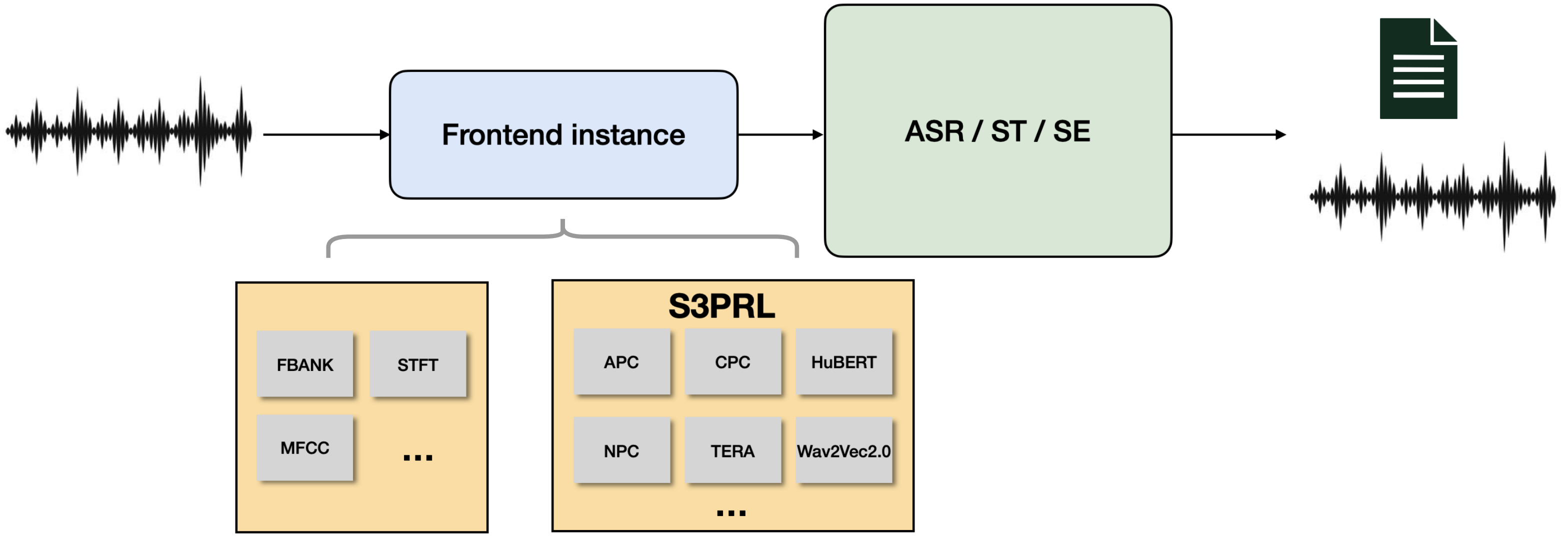
| Dataset | Transformer-CTC | Conformer-CTC |
|---|---|---|
| CSJ | 6.0 / 4.2 / 4.8 | **4.8 / 3.7 / 3.8** |
| TEDLIUM2 | 16.7 / 16.6 | **9.3 / 8.7** |
| VoxForge | 14.0 / 14.1 | **9.2 / 8.4** |
| WSJ | 19.4 / 15.5 | **12.9 / 10.9** |

CER results of different Transducer models on the VIVOS corpus.

| Model | dev | test |
|---|---|---|
| Transformer-Transducer | 17.2 | 17.1 |
| Conformer-Transducer | 13.7 | 14.0 |
| TDNN-Conformer-Transducer | **11.6** | **13.1** |

# Combine ESPnet with S3PRL [Yang+ 2021, Chang+ 2021]

- Self-supervised pretraining on speech data have achieved a lot of progress, like wav2vec2.0 [Baevski+ 2020], Hubert [Hsu+ 2021], etc.
- S3PRL* toolkit provides an integration of pretrained speech representation models and speech tasks, e.g., wav2vec2.0 + LSTM acoustic model for ASR.
- Support combine the pretrained models with advanced end-to-end speech processing models in a simple way.




*: https://github.com/s3prl/s3prl

# ASR Experiments (80 hours English task)

- Word Error Rate (%) on WSJ corpus

| Models | dev93 | dev92 |
|---|---|---|
| Kaldi Chain Model | 4.3 | 2.3 |
| ESPnet Conformer | 6.6 | 4.4 |
| **ESPnet Conformer + wav2vec2.0** | **2.8** | **1.8** |
| **ESPnet Conformer + Hubert** | 3.1 | **1.8** |

**Reach the state-of-the-art results.**

# ASR Experiments (960 hours English task)

- Word Error Rate (%) on Librispeech corpus

| Toolkit | dev_clean | dev_other | test_clean | test_other |
|---|---|---|---|---|
| Google Conformer | 2.1 | 4.3 | 1.9 | 3.9 |
| ESPnet Conformer | 1.9 | 4.6 | 2.1 | 4.7 |
| Facebook wav2vec2.0 (60k LibriVox) | **1.6** | **3.0** | **1.8** | **3.3** |
| Facebook Hubert | 1.7 | 3.0 | 1.9 | 3.5 |
| **ESPnet Conformer + wav2vec2.0** | 1.9 | 5.4 | 2.2 | 5.2 |
| **ESPnet Conformer + Hubert** | **1.7** | **3.4** | **1.8** | **3.6** |

**Obtain further improvements with the help of self-supervised pretrained models.**

# Future Work on Self-Supervised Pretrained Models

- Conduct comprehensive experiments on more corpora and more tasks
- Investigate the efficiency of self-supervised pretrained models on multi-lingual datasets
- Explore different scenarios, like domain mismatch, low-resource, etc.

# How to build an ASR system with ESPnet

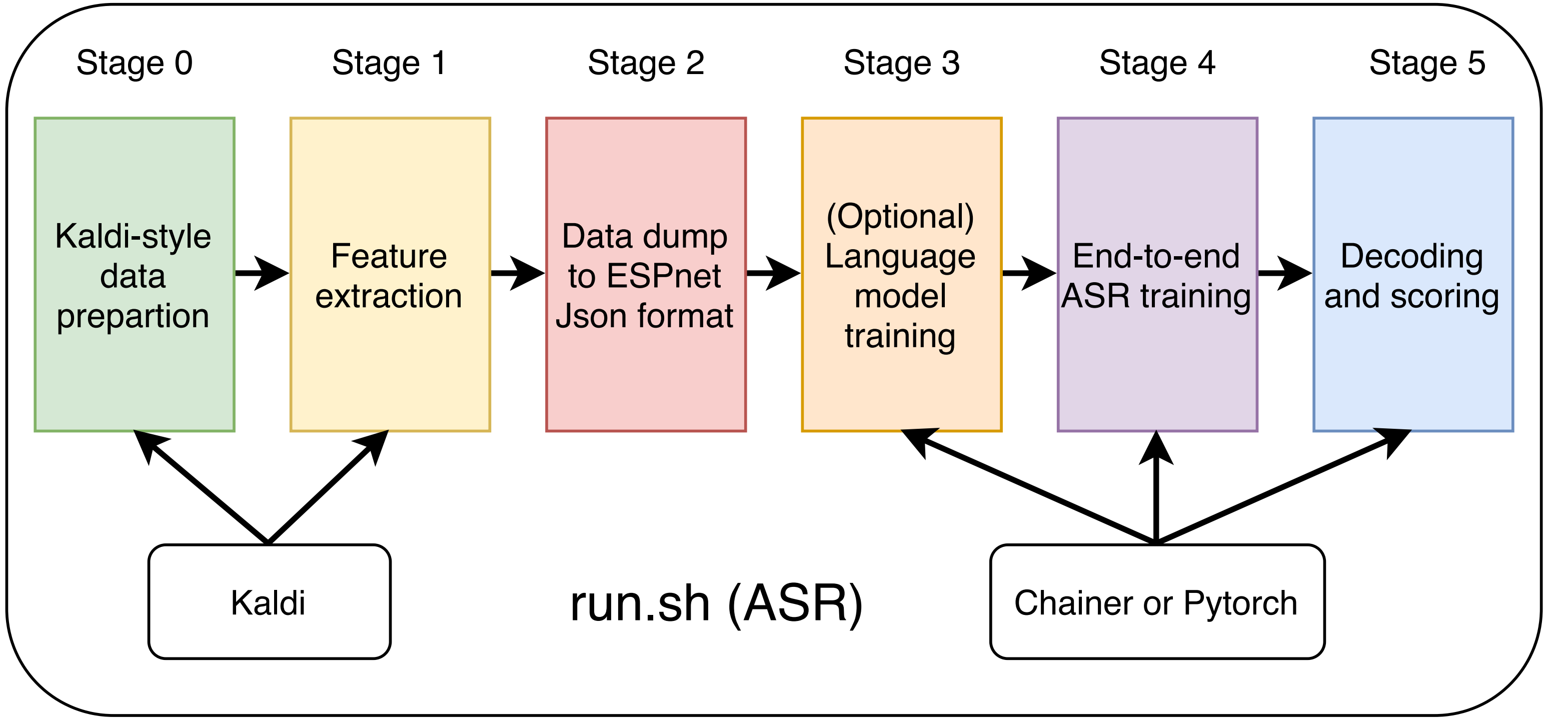
- Each recipe is organized as "egs/***/asr1/run.sh"
- The most import directories:
  - "conf/": configurations for stages and computation clusters
  - "data/": raw data prepared by Kaldi, e.g., wav.scp, text, utt2spk, etc.
  - "dump/": dumped json format data for ESPnet
  - "exp/": saved model parameters and log files

```
!tree -L 1 espnet/egs/librispeech/asr1
```

```
espnet/egs/librispeech/asr1
├── cmd.sh
├── conf
├── local
├── path.sh
├── RESULTS.md
├── run.sh
├── steps -> ../../../tools/kaldi/egs/wsj/s5/steps
└── utils -> ../../../tools/kaldi/egs/wsj/s5/utils
```

# How to build an ASR system with ESPnet

- Basic flow of recipes




https://github.com/espnet/espnet/blob/master/egs/librispeech/asr1/run.sh

- Simple Flow
  - No GMM
  - No FST
  - No alignment
  - No lattice output
- Easy to expand
  - Various frameworks
- All-in-one recipe
  - Data download
  - Data preparation
  - Training & inference
  - Reproducible results
  - Pretrained models

# A more flexible structure: ESPnet2

- Main differences between ESPnet1 and ESPnet2

| | ESPnet1 | ESPnet2 |
|---|---|---|
| Data prepartion | Rely on Kaldi, before training | Kaldi-free, on-the-fly processing |
| Feature extraction | | |
| Data dump | Memory & time cost | |
| LM training | Chainer-based trainer DataParallel Training | Chainer-free DistributedDataParallel Training |
| ASR training | | |

# How to combine self-supervised pretrained models

- ESPnet2 has already supported loading the self-supervised pretrained models as the ASR frontends
- All we need to do is change the configuration file

Freeze the params. of upstream model

```
freeze_param: [
"frontend.upstream"
]
```

Choose an upstream model

```
frontend: s3prl
frontend_conf:
    frontend_conf:
        upstream: hubert_large_ll60k  # Note: If the upstream is changed, please change the input_size in the preencoder.
    download_dir: ./hub
    multilayer_feature: true
```

Add a feature transform layer

```
preencoder: linear
preencoder_conf:
    input_size: 1024  # Note: If the upstream is changed, please change this value accordingly.
    output_size: 80
```

# Next Section by Sathvik

How to build a multilingual and code-switching ASR system for the low resource India languages?

# Overview of today's tutorial

- **5pm to 6pm:** part I presentation by Shinji
  - Introduction of end-to-end ASR and ESPnet
- **6pm to 6:30 pm:** Q&A for part I and break
- **6:30pm to 7pm:** part II presentation by Pengcheng
  - Advanced techniques in ESPnet
- **7pm to 7:15 pm:** part II espnet mucs recipe by Sathvik
  - espnet mucs recipe, and demo
- **7:15pm to 7:30pm:** summary and Q&A by Shinji

# espnet mucs recipe, and demo

- Materials
  - https://github.com/bloodraven66/writeup/blob/main/TUTORIAL.MD
- ASR demo (we'll update MUCS models soon)
  - https://colab.research.google.com/github/espnet/notebook/blob/master/espnet2_asr_realtime_demo.ipynb

# Overview of today's tutorial

- **5pm to 6pm:** part I presentation by Shinji
  - Introduction of end-to-end ASR and ESPnet
- **6pm to 6:30 pm:** Q&A for part I and break

- **6:30pm to 7pm:** part II presentation by Pengcheng
  - Advanced techniques in ESPnet
- **7pm to 7:15 pm:** part II espnet mucs recipe by Sathvik
  - espnet mucs recipe, and demo

- **7:15pm to 7:30pm:** summary and Q&A by Shinji

# Summary

- End-to-End speech processing has a lot of potentials especially for the multilingual setup
  - But it always has pros and cons
- ESPnet now reaches SOTA again
  - Conformer/self-supervised training
- We can easily build an ESPnet recipe for a new language

- Why I like end-to-end?
  - It becomes very simple

# Summary

- End-to-End speech processing has a lot of potentials especially for the multilingual setup
  - But it always has pros and cons
- ESPnet now reaches SOTA again
  - Conformer/self-supervised training
- We can easily build an ESPnet recipe for a new language

- Why I like end-to-end?
  - It becomes very simple

# One of my research goals

- Speech recognition as a **simple machine learning** problem
  - Like MNIST, speech recognition would be a tutorial of machine learning toolkit soon with more simplifications
  - Everyone (even high school student) can build an ASR system

# One of my research goals

- Speech recognition as a **simple machine learning** problem
  - Like MNIST, speech recognition would be a tutorial of machine learning toolkit soon with more simplifications
  - Everyone (even high school student) can build an ASR system

- Do we lose a job?
  - Don’t worry. We still have tons of more challenging speech recognition problems

noisy speech, multispeaker, understanding, dialogue systems

# One of my research goals

- Speech recognition as a **simple machine learning** problem
  - Like MNIST, speech recognition would be a tutorial of machine learning toolkit soon with more simplifications
  - Everyone (even high school student) can build an ASR system

- Do we lose a job?
  - Don’t worry. We still have tons of more challenging speech recognition problems

noisy speech, multispeaker, understanding, dialogue systems

Let’s work together to make ASR more simple/easier problems (by techniques or open source) and focus on other challenging issues!

# One of my research goals

- Speech recognition as a **simple machine learning** problem
  - Like MNIST, speech recognition would be a tutorial of machine learning toolkit soon with more simplifications
  - Everyone (even high school student) can build an ASR system

- Do we lose a job?
  - Don’t worry. We still have tons of more challenging speech recognition problems

noisy speech, multispeaker, understanding, dialogue systems

Let’s work together to make ASR more simple/easier problems
(by techniques or open source)
and focus on other challenging issues!

# Thanks!

Special thanks to Prasanta Kumar Ghosh, Anuj Diwan, Sanket Shah, Shreya Khare, Preethi Jyothi for their great help on this tutorial