Speech Recognition

음성인식

경북대학교 인공지능학과 김준우 박사과정
kaen2891@gmail.com

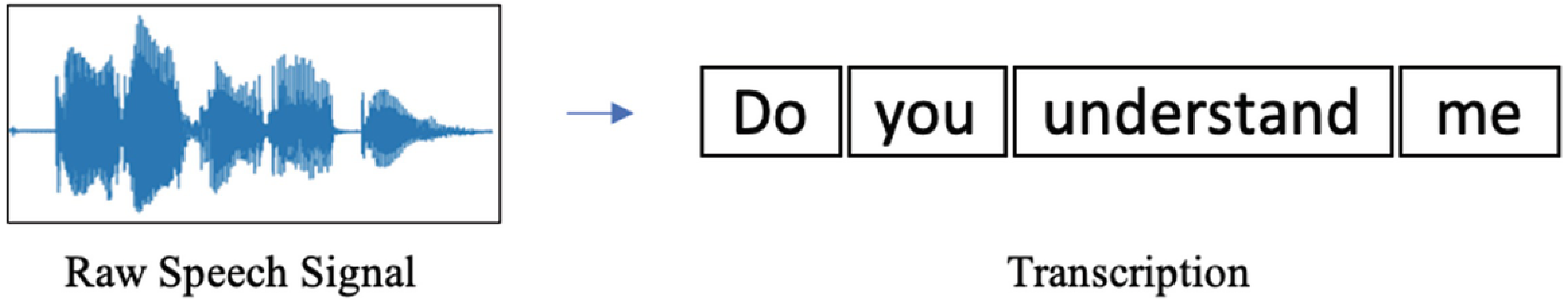
14, Feb. 2022.

Introduction of ASR

Speech Recognition

- Speech Recognition (SR)
  - Automatic Speech Recognition (ASR)
  - STT (Speech-to-Text)

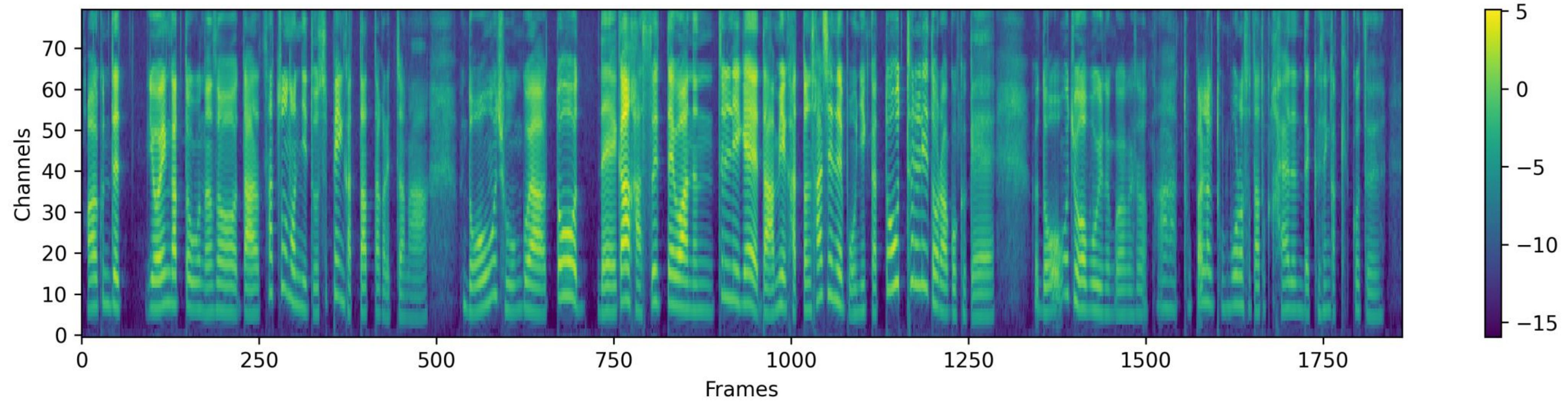
- 주어진 입력 파형을 완벽한 문자로 변환



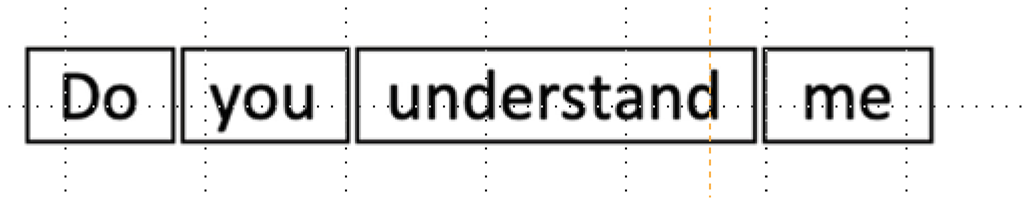
Speech Recognition

- ~ 2015
  - HMM (Hidden Markov Model) & GMM (Gaussian Mixture Model)-based machine learning approach
- 2016 ~
  - Deep Learning-based End-to-End ASR

- Model Input
  - Mel-spectrogram

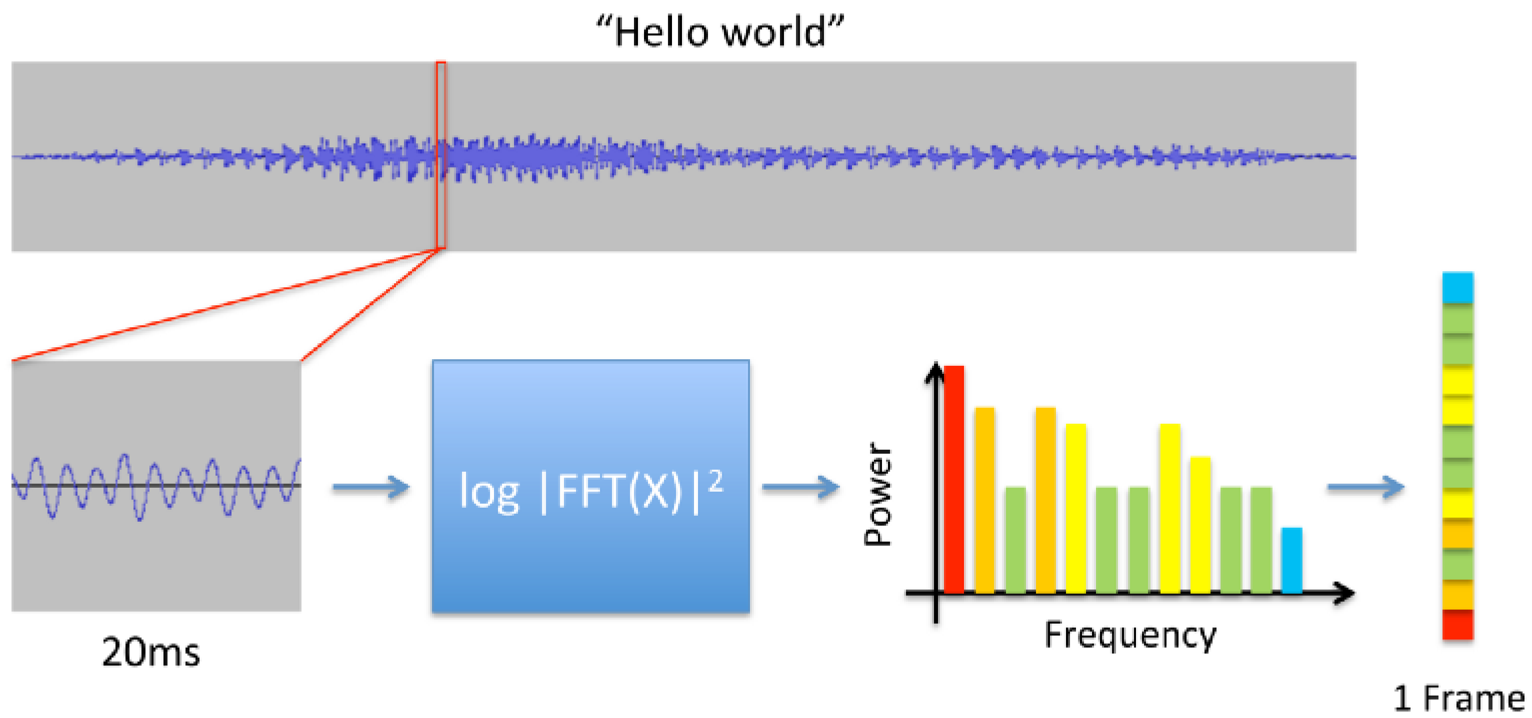


- Output
  - Text



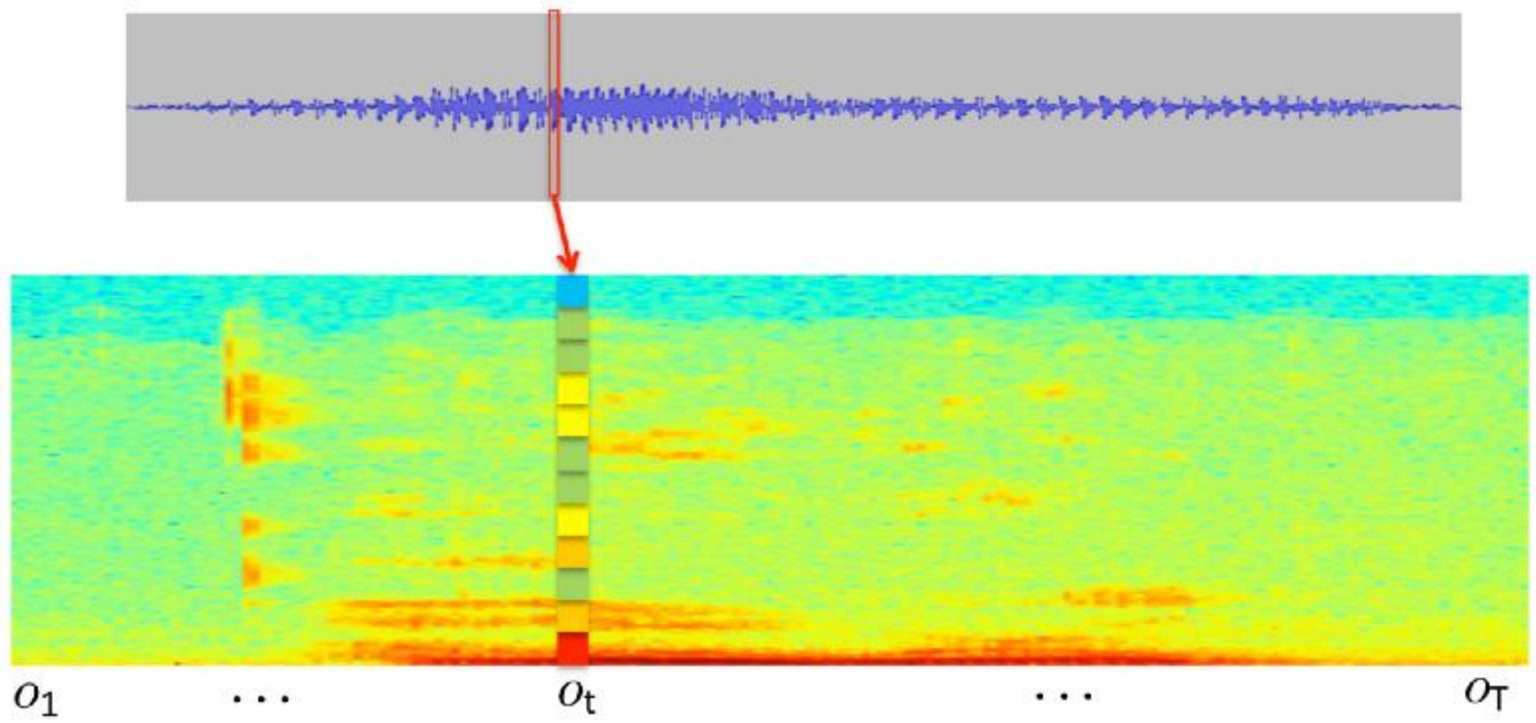
Speech Recognition

- Dataset
  - Short-time Fourier Transform (STFT)



Speech Recognition

- Dataset
  - Spectrogram

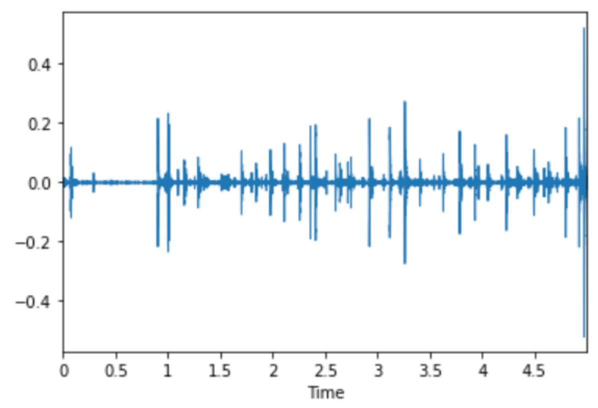


- Dataset
  - Waveform, Spectrogram, Mel-spectrogram

```
[45] display(Audio(audio_file, autoplay=True))
     librosa.display.waveplot(wav, sr)
```

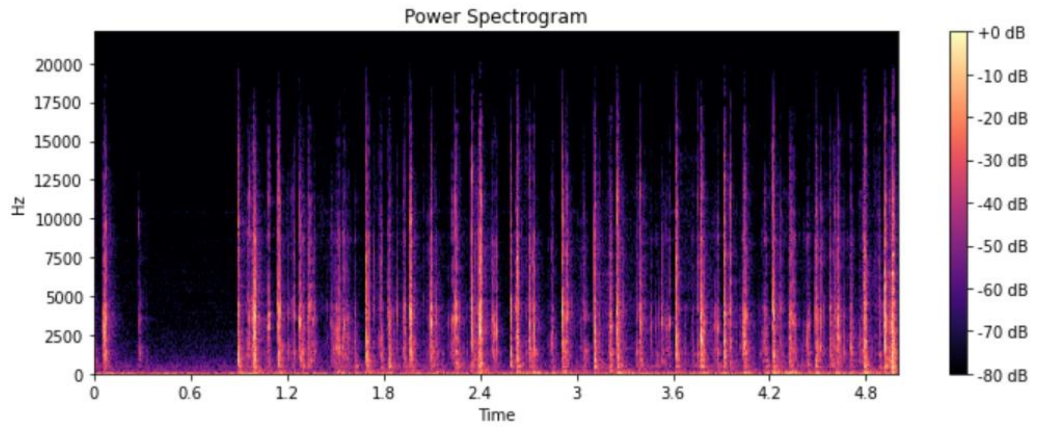
0:05 / 0:05

<matplotlib.collections.PolyCollection at 0x7fb127154c50>

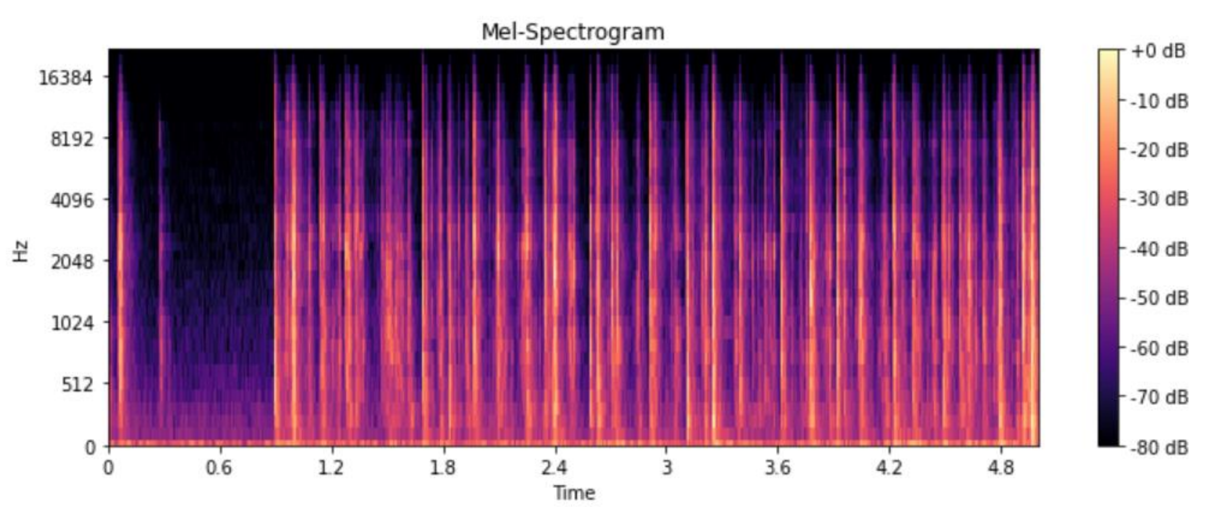



```
[47] plt.figure(figsize=(10, 4))
     librosa.display.specshow(librosa.amplitude_to_db(audio_stft, ref=np.max), y_axis='hz', x_axis='time', sr=sr, hop_length=256)
     plt.title("Power Spectrogram")
     plt.colorbar(format='%+2.0f dB')
     plt.tight_layout()
```

```
/usr/local/lib/python3.7/dist-packages/librosa/core/spectrum.py:1642: UserWarning: amplitude_to_db was called on complex inpu
  "amplitude_to_db was called on complex input so phase "
```

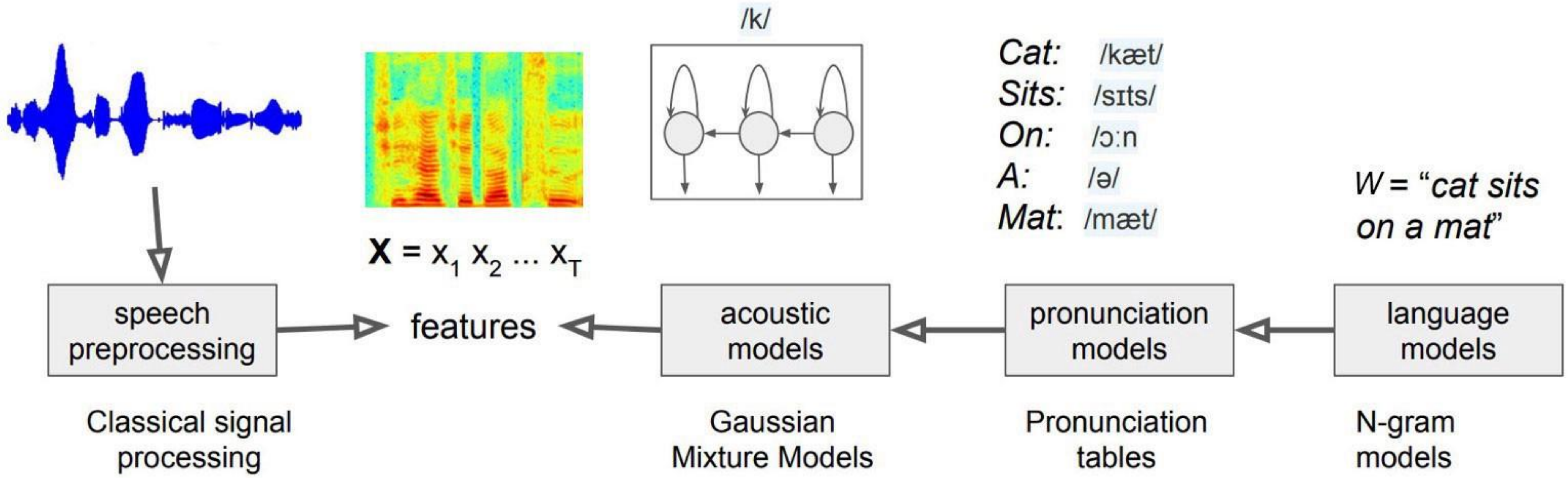



```
[49] plt.figure(figsize=(10, 4))
     librosa.display.specshow(librosa.power_to_db(audio_mel, ref=np.max), y_axis='mel', x_axis='time', sr=sr, hop_length=256)
     plt.title("Mel-Spectrogram")
     plt.colorbar(format='%+2.0f dB')
     plt.tight_layout()
```



Cascade ASR

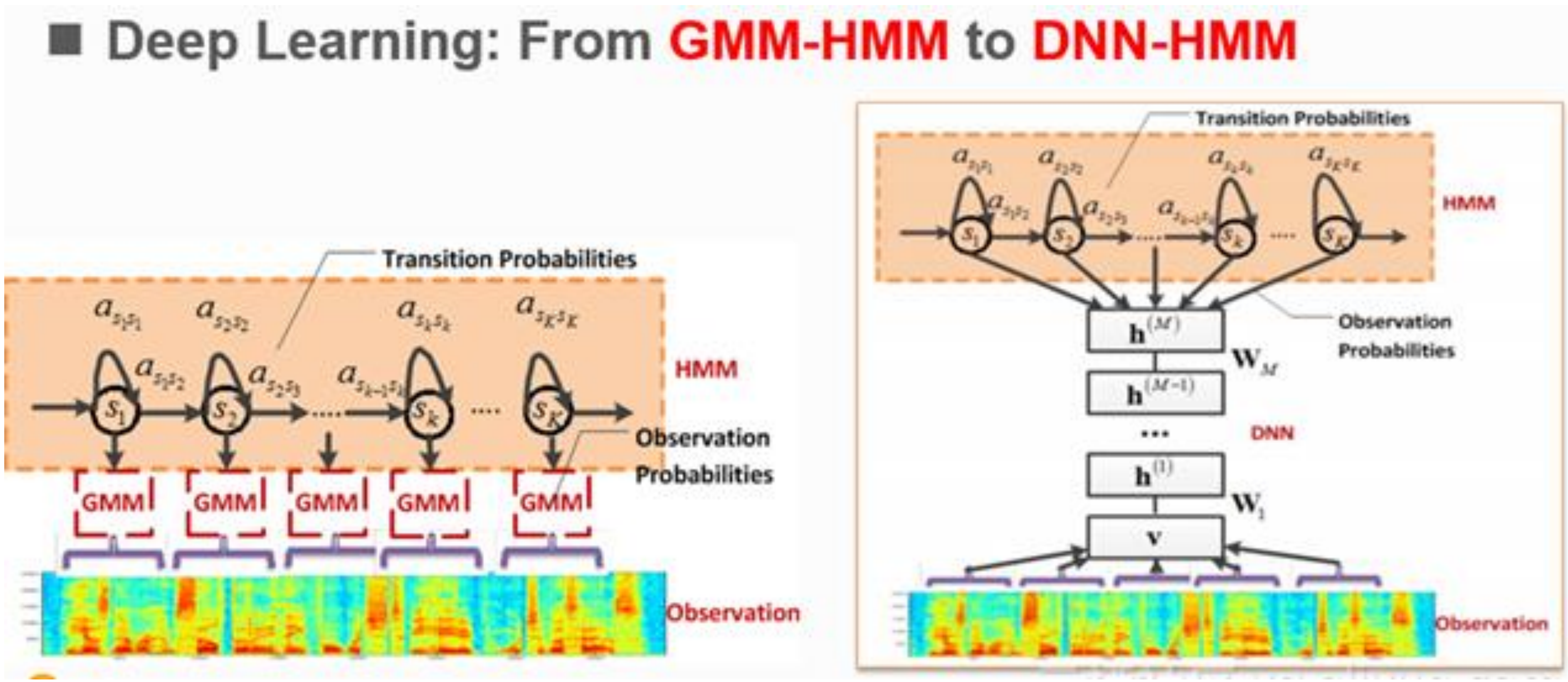
- LM (Language Model)
- AM (Acoustic Model)



$$W^* = \arg\max_{W} \underbrace{p(\mathbf{X}|\mathbf{W})}_{\text{acoustic model}} \underbrace{p(\mathbf{W})}_{\text{language model}}$$

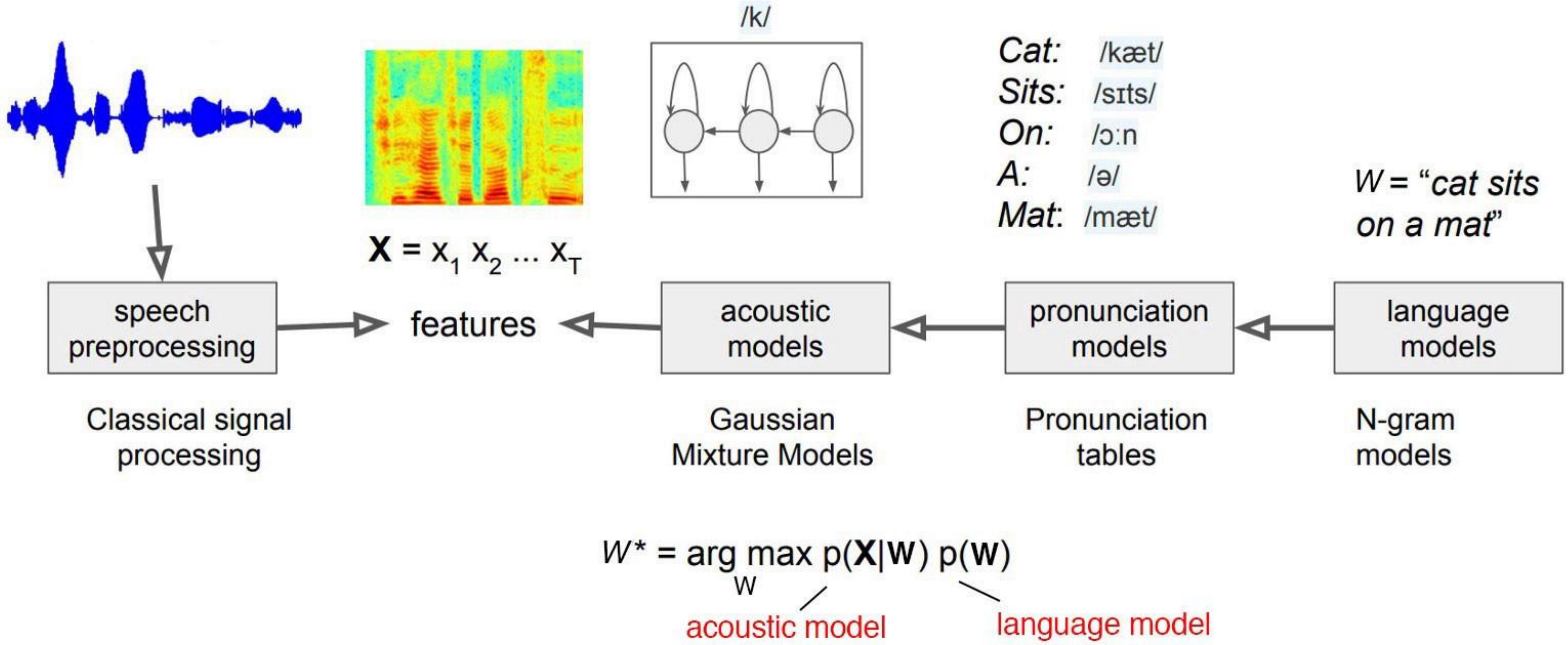
Previous ASR

- GMM-HMM



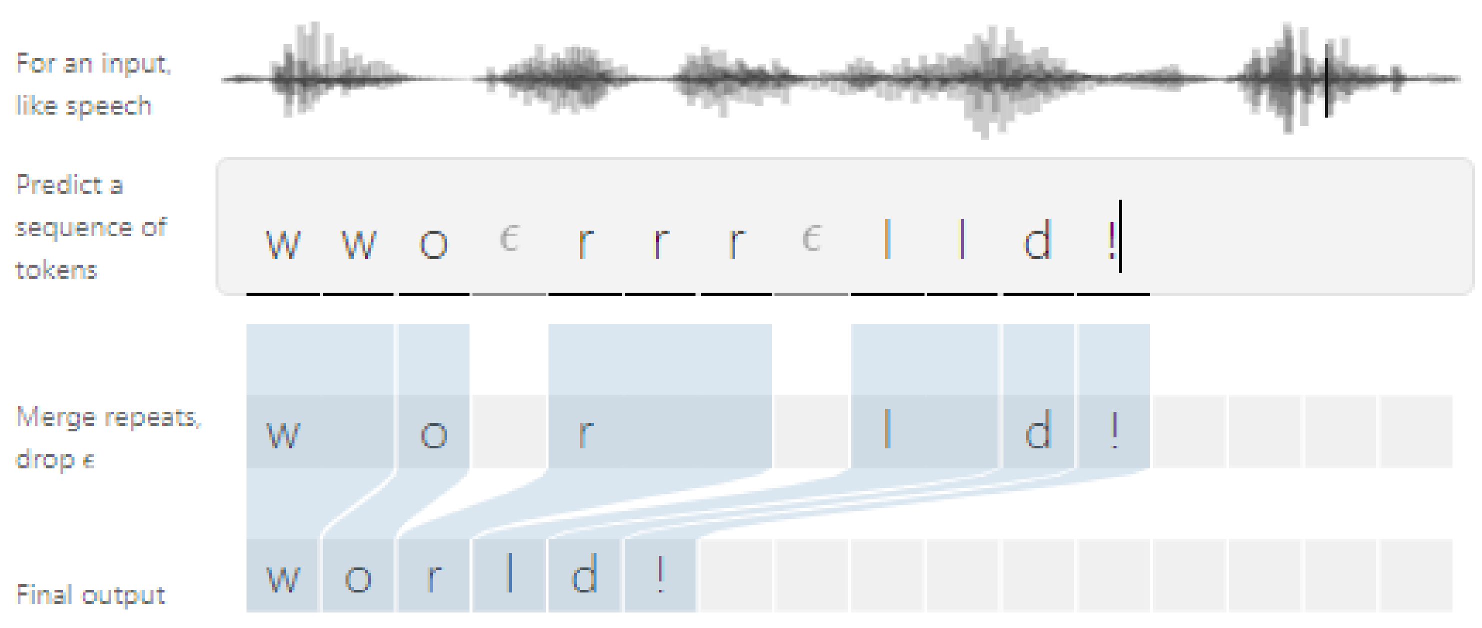
Previous ASR

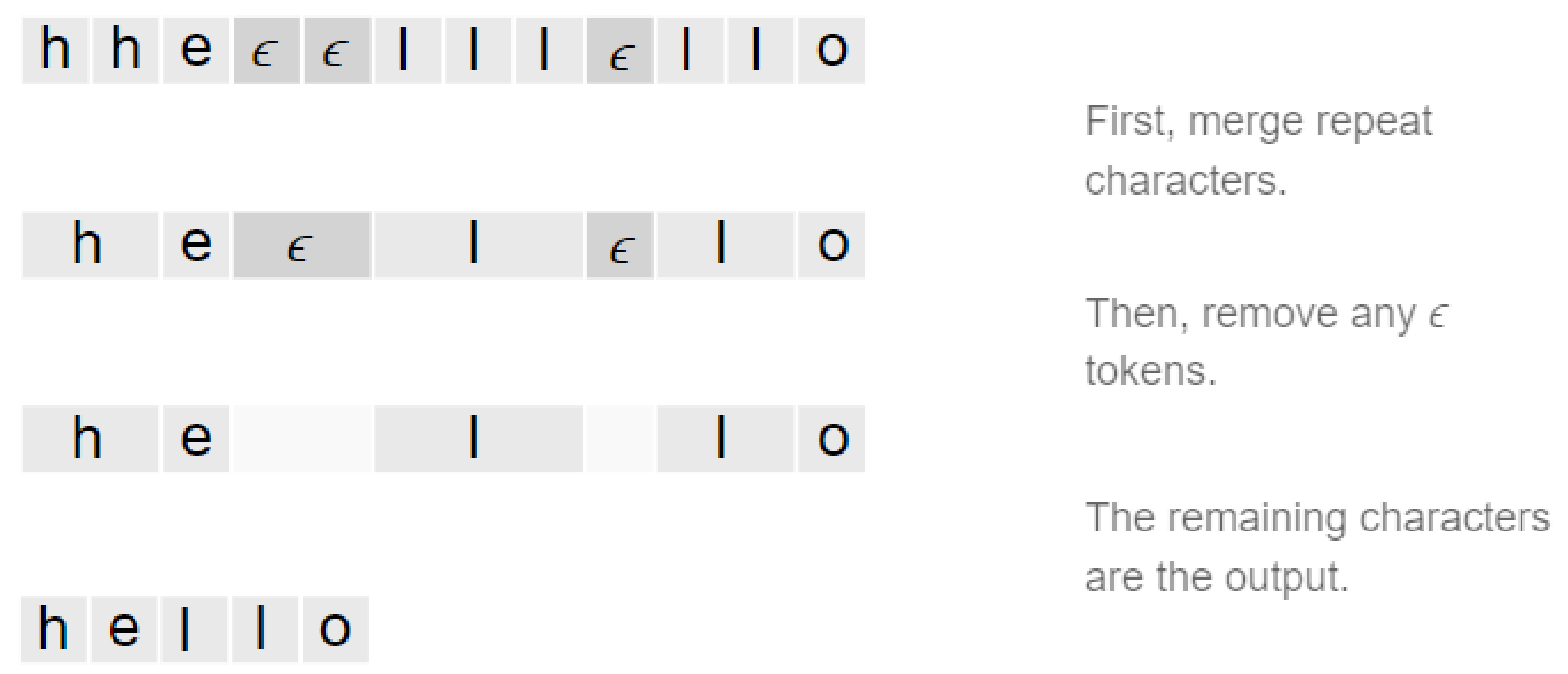
- GMM-HMM



CTC (2006)

- Connectionist Temporal Classification (CTC)

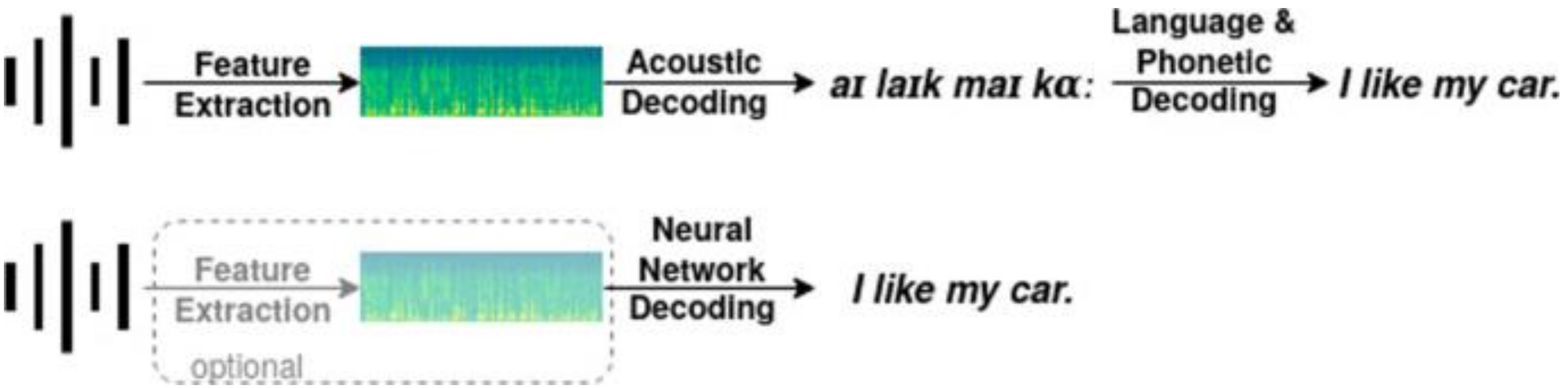




Graves, Alex, et al. "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks." *Proceedings of the 23rd international conference on Machine learning*. 2006.

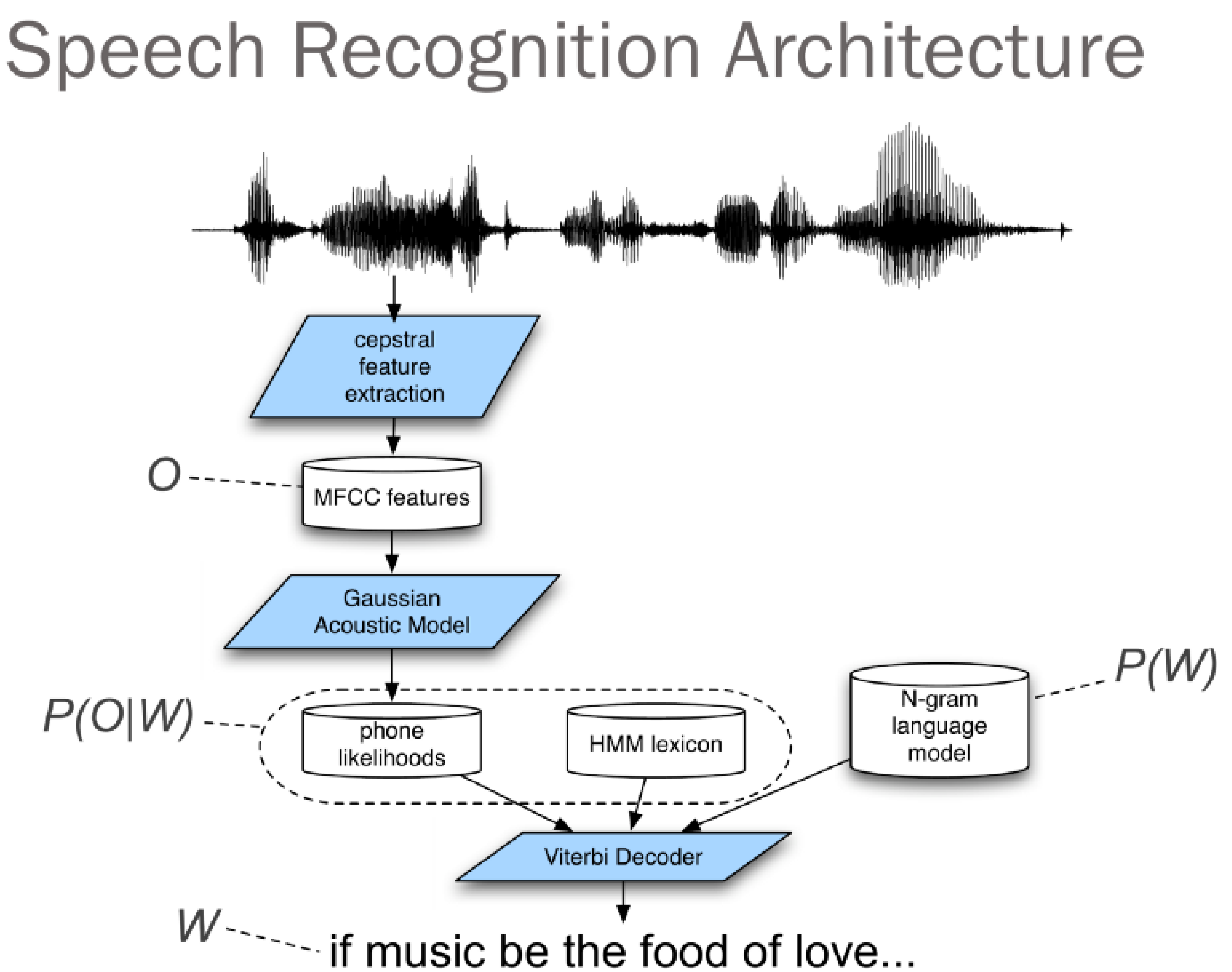
ASR

- End-to-End vs Cascade
  - Pipeline (top) vs end-to-end (bottom) ASR



Georgescu, Alexandru-Lucian, et al. "Performance vs. hardware requirements in state-of-the-art automatic speech recognition." *EURASIP Journal on Audio, Speech, and Music Processing* 2021.1 (2021): 1-30.

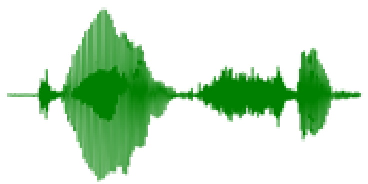
- Cascade



Speech Recognition Architecture
cepstral feature extraction
O
MFCC features
Gaussian Acoustic Model
P(O|W)
phone likelihoods
HMM lexicon
N-gram language model
P(W)
Viterbi Decoder
W
if music be the food of love...

- End-to-End ASR

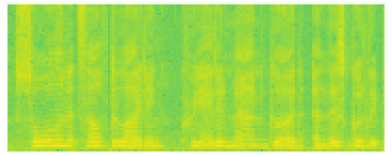
audio



Preprocessing

- Data augmentation
    - additive Gaussian noise
    - time stretch (resampling)
- Windowing 20-25 ms, stride 10 ms
- FFT, log
- Normalization

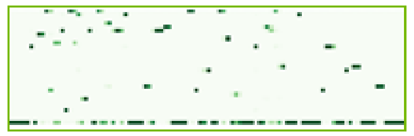
spectrogram



DNN

- 2-3 convolutional layers
    - ch=32, ks=[11, 41], s=[2, 2]
    - ch=32, ks=[11, 21], s=[1, 2]
    - ch=64, ks=[11, 21], s=[1, 2]
- 3-7 bi-/uni- directional GRUs/LSTMs
- 1 row conv layer (for unidirectional)
- 1-2 fully connected layers
- BN/dropout for regularization

$P_t(c)$



training

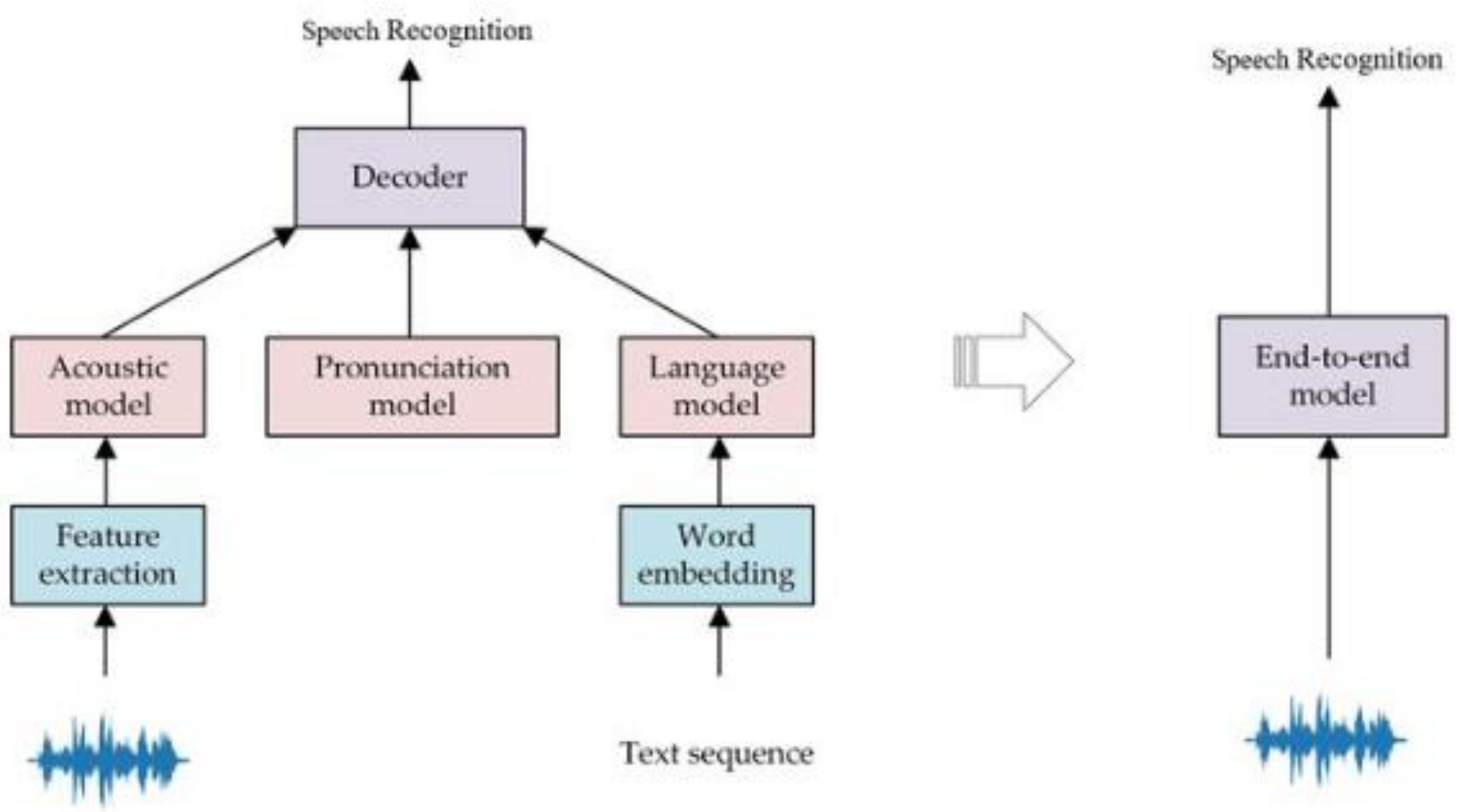
CTC Loss

inference

Decoder

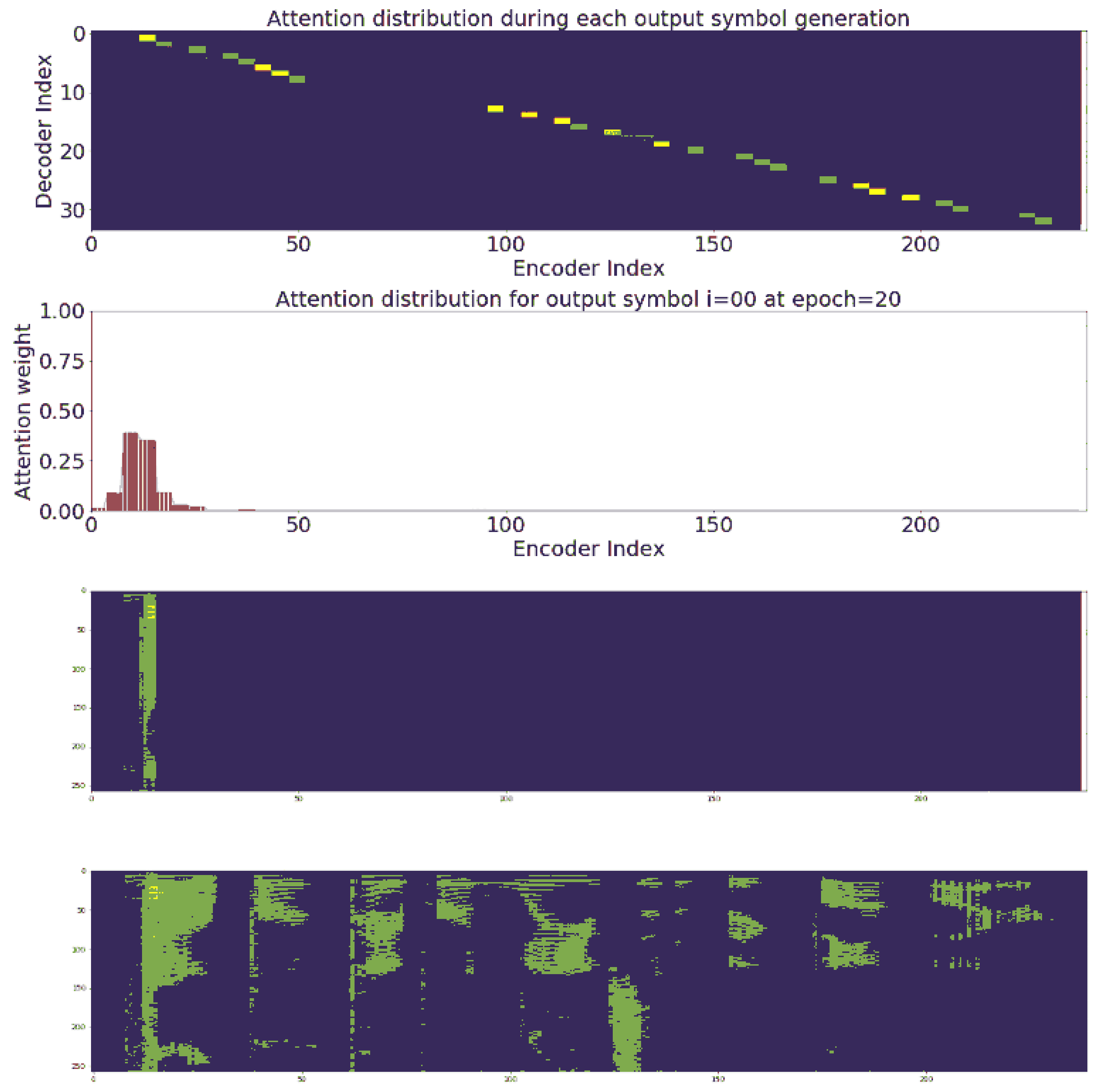
- greedy (argmax)
- beam search
- beam search with language model (width=2000-8000)

- End-to-End vs Cascade

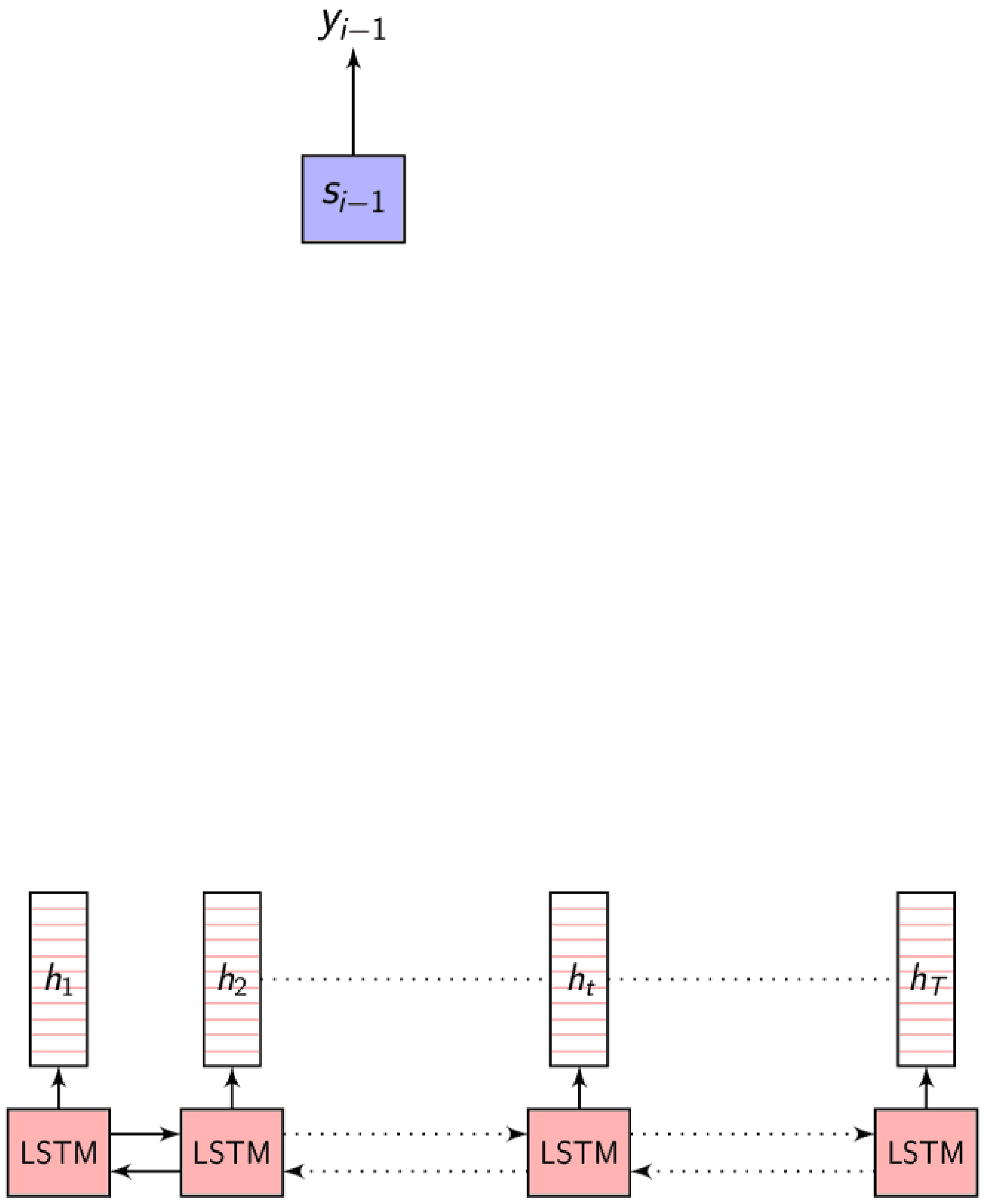


ASR

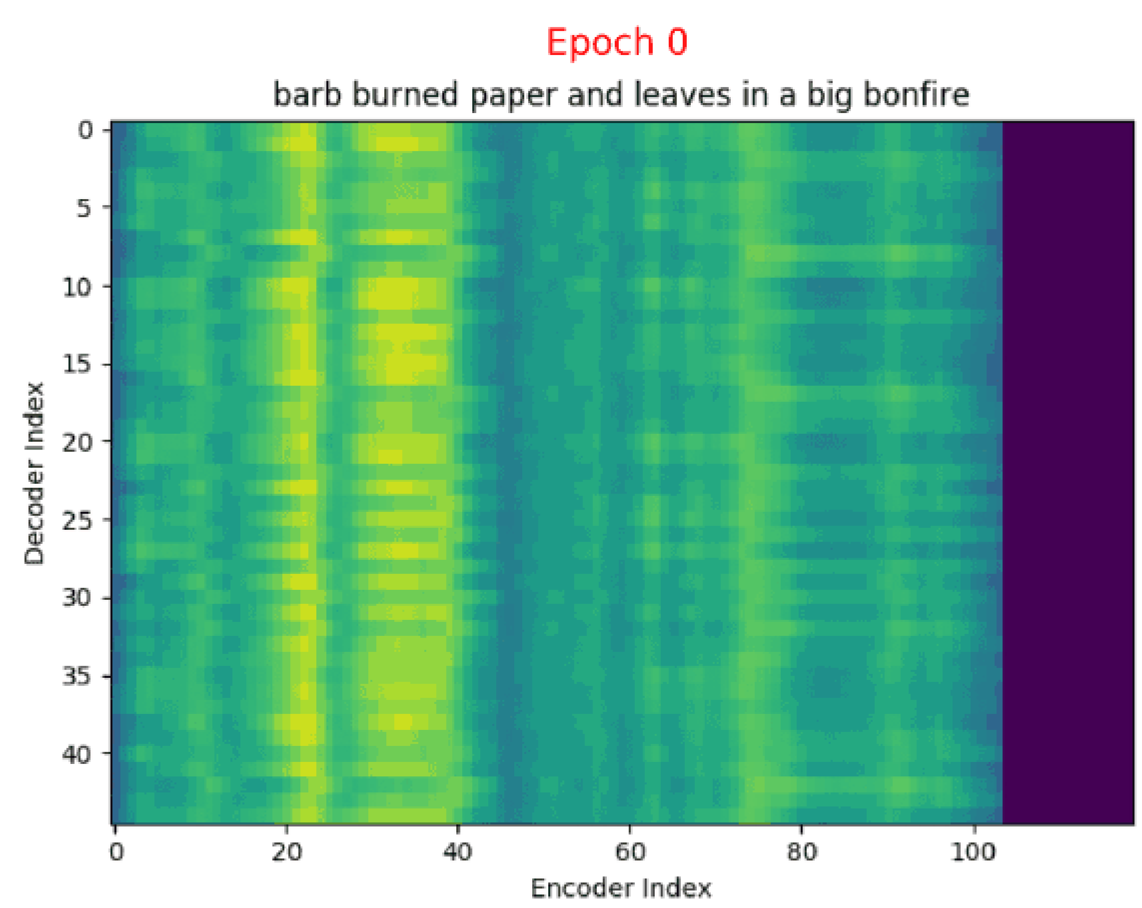
- End-to-End vs Cascade



- LSTM-based ASR



- Seq2seq with Attention



LAS (2015)

- Listen, attend and spell
- First end-to-end ASR model
  - Hierarchical encoder reduces time resolution (pyramidal BiLSTM)

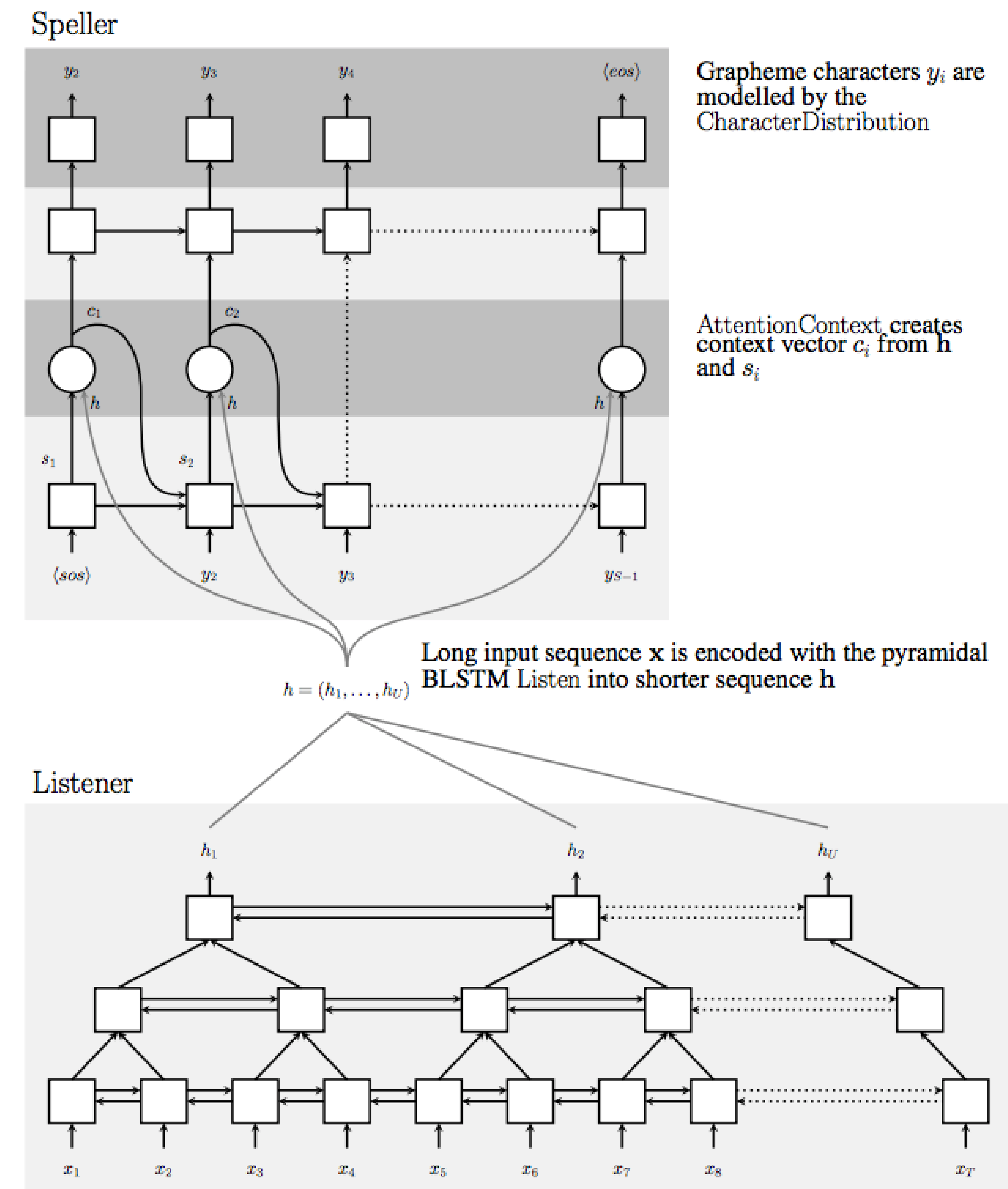


Figure 1: Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence **x** into high level features **h**, the speller is an attention-based decoder generating the **y** characters from **h**.

Chan, William, et al. "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition." *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016.

LAS (2015)

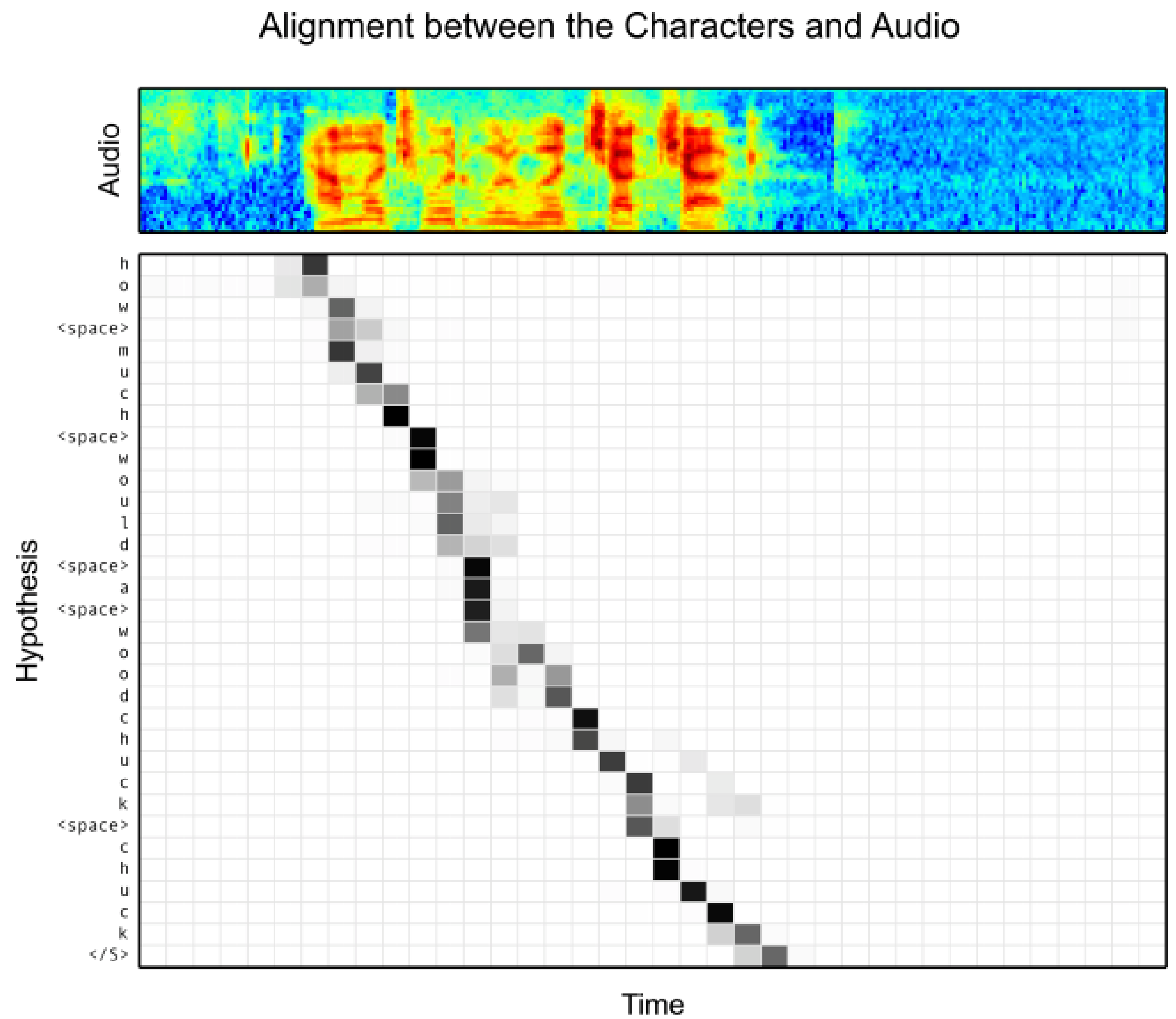


Figure 2: Alignments between character outputs and audio signal produced by the Listen, Attend and Spell (LAS) model for the utterance "how much would a woodchuck chuck". The content based attention mechanism was able to identify the start position in the audio sequence for the first character correctly. The alignment produced is generally monotonic without a need for any location based priors.

Chan, William, et al. "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition." *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016.

- CTC + Attention

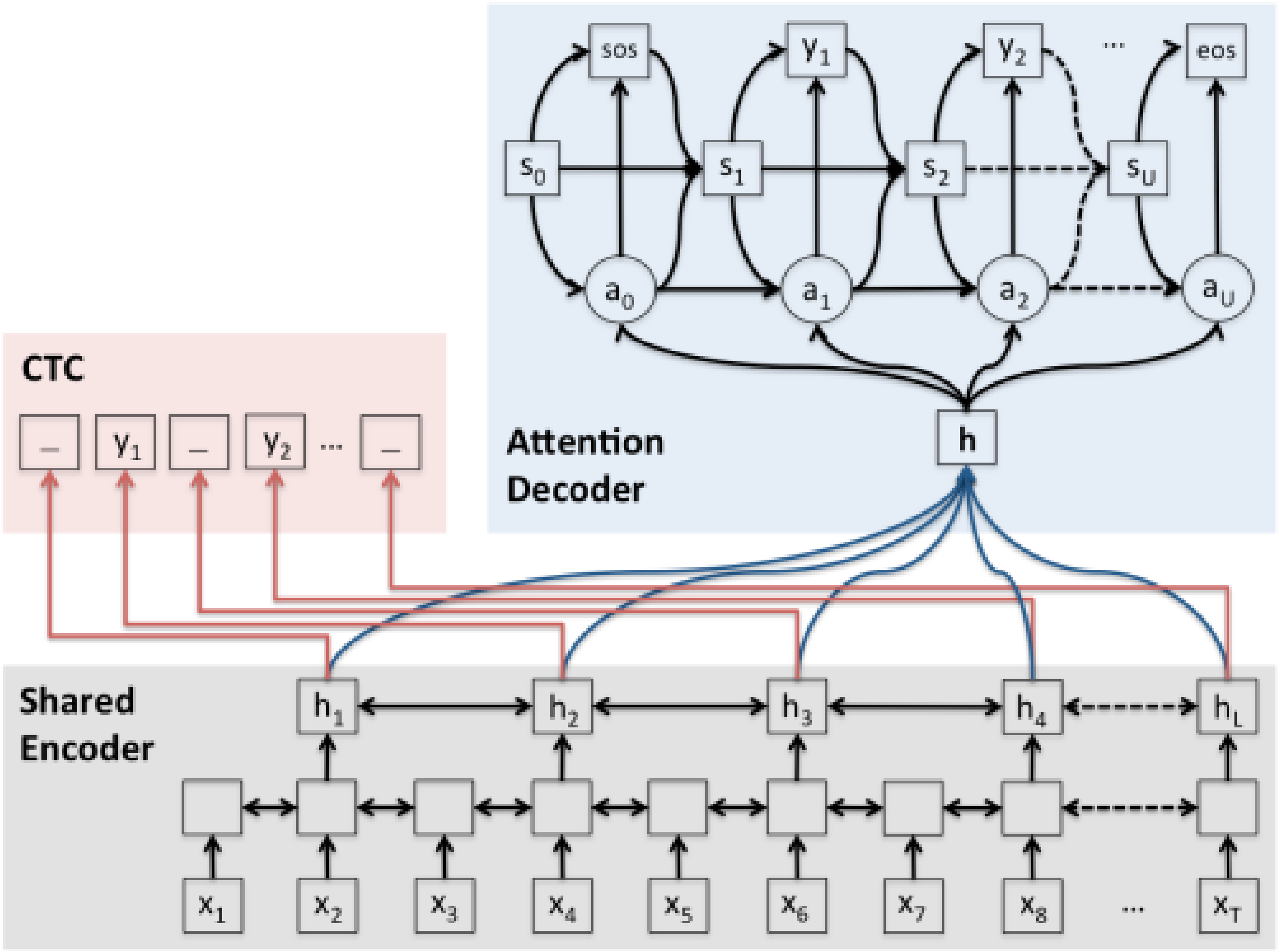


Fig. 1: Our proposed Joint CTC-attention based end-to-end framework: the shared encoder is trained by both CTC and attention model objectives simultaneously. The shared encoder transforms our input sequence $\boldsymbol{x}$ into high level features $\boldsymbol{h}$, the location-based attention decoder generates the character sequence $\boldsymbol{y}$.

Kim, Suyoun, Takaaki Hori, and Shinji Watanabe. "Joint CTC-attention based end-to-end speech recognition using multi-task learning." *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017.

Hybrid CTC-Attention (2017)

- CTC + Attention

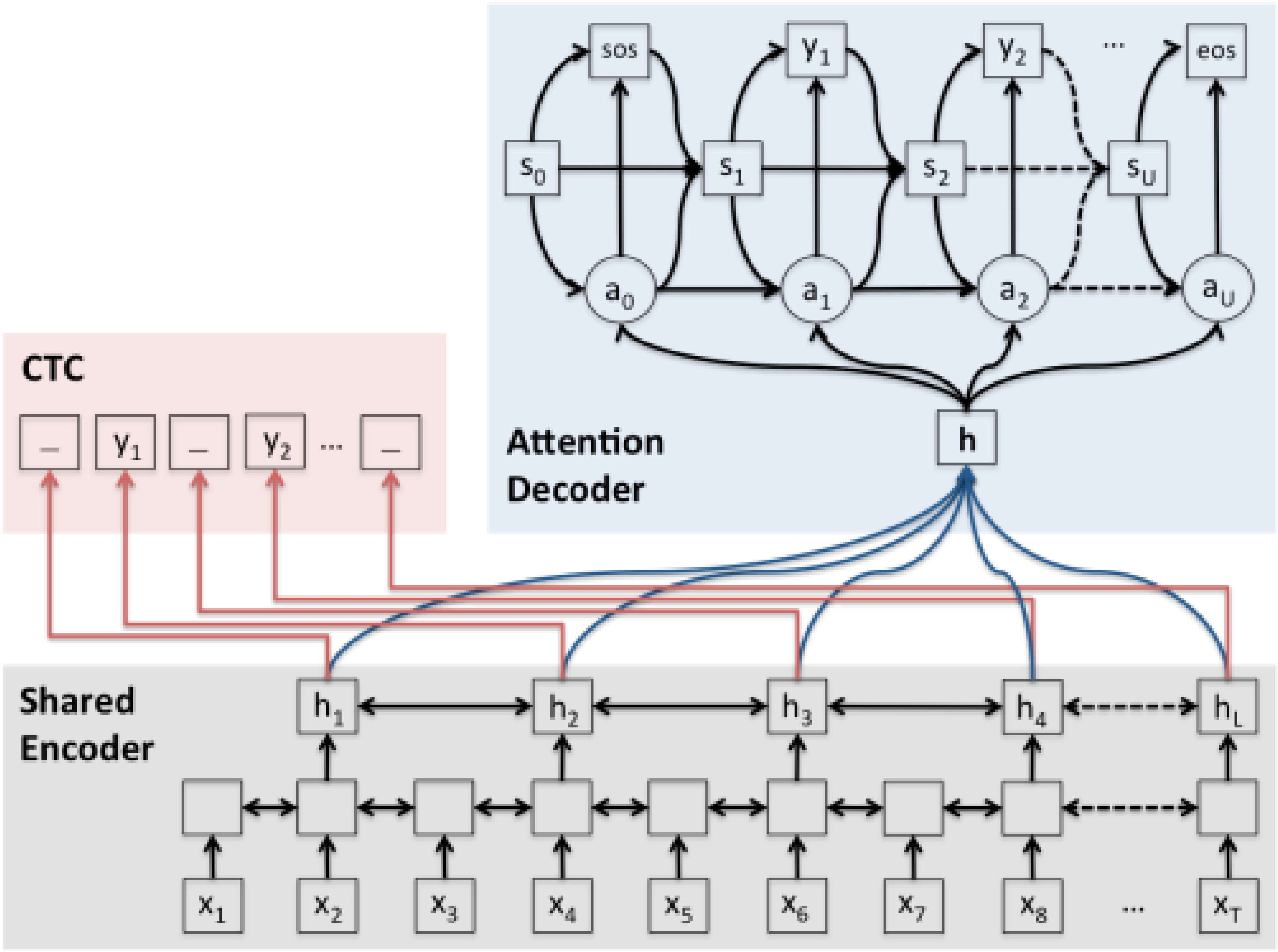
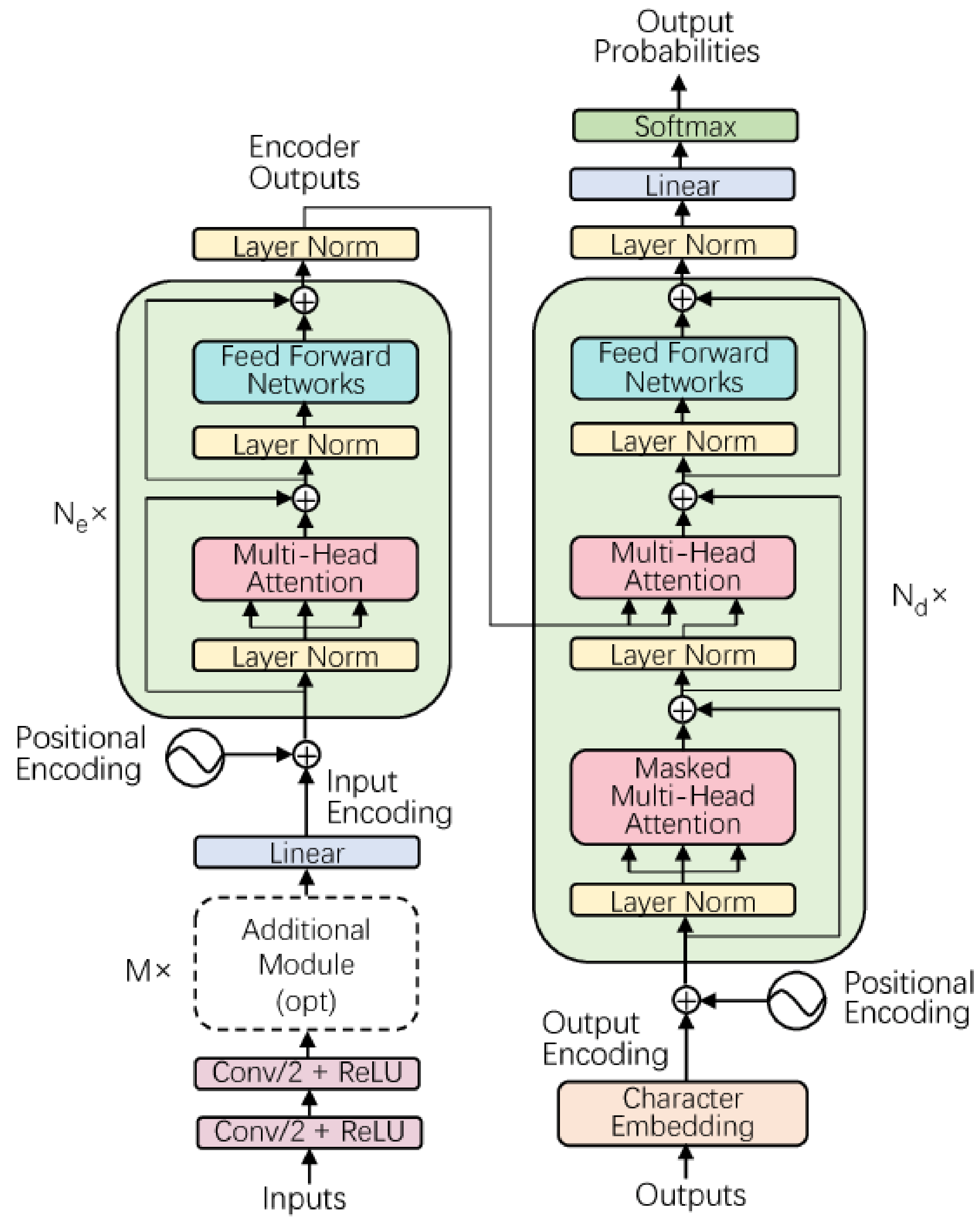


Fig. 1: Our proposed Joint CTC-attention based end-to-end framework: the shared encoder is trained by both CTC and attention model objectives simultaneously. The shared encoder transforms our input sequence $\boldsymbol{x}$ into high level features $\boldsymbol{h}$, the location-based attention decoder generates the character sequence $\boldsymbol{y}$.

Kim, Suyoun, Takaaki Hori, and Shinji Watanabe. "Joint CTC-attention based end-to-end speech recognition using multi-task learning." *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017.

ASR with Transformer (2018)

- ASR with Transformer
  - Adopt Transformer-based End-to-End ASR model



Dong, Linhao, Shuang Xu, and Bo Xu. "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition." *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018.

Conformer (2020)

- Covolutional Transformer
  - Adopt the advantages of CNN and Transformer

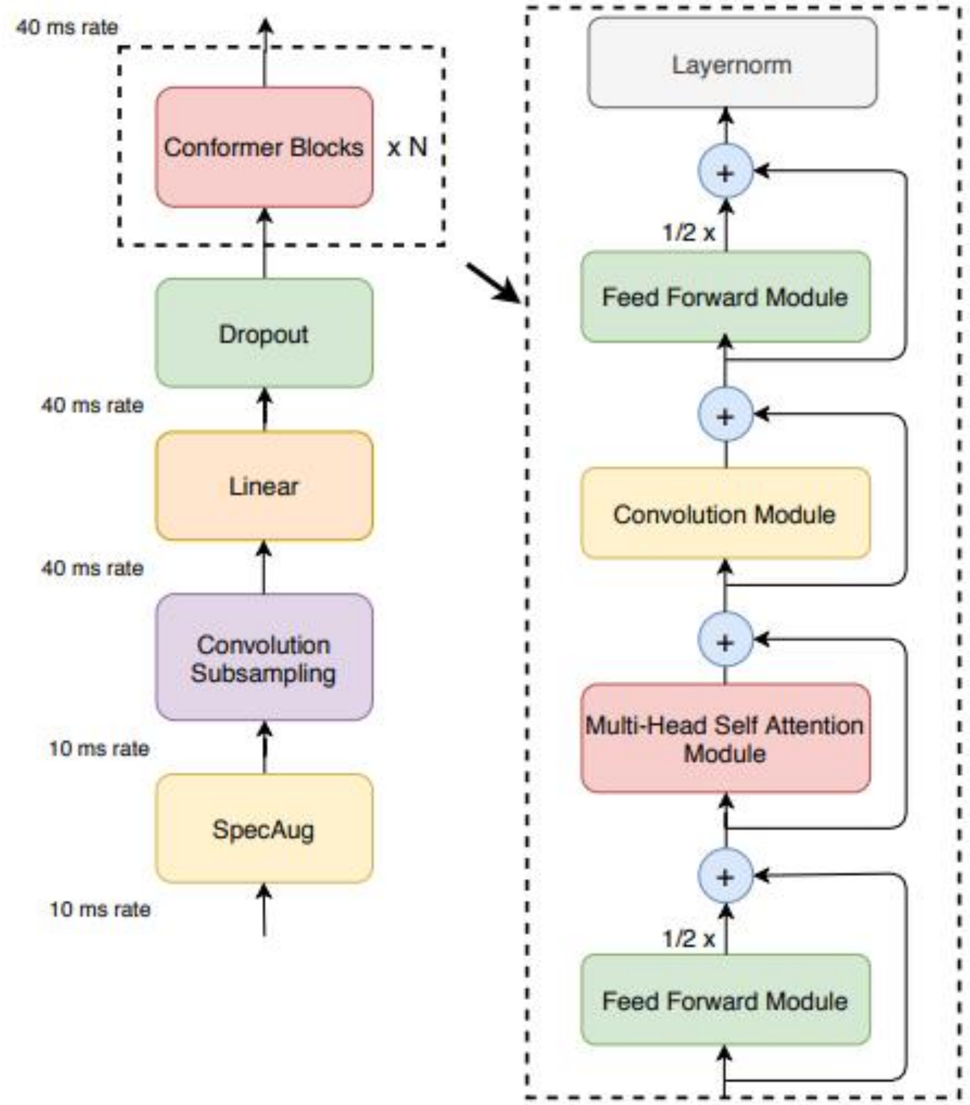


Figure 1: ***Conformer encoder model architecture.*** *Conformer comprises of two macaron-like feed-forward layers with half-step residual connections sandwiching the multi-headed self-attention and convolution modules. This is followed by a post layernorm.*

Gulati, Anmol, et al. "Conformer: Convolution-augmented Transformer for Speech Recognition." *Proc. Interspeech 2020* (2020): 5036-5040.

- Multi-Head Self Attention Module
    - Sinusoidal positional encoding -> relative sinusoidal positional encoding in [Ref]

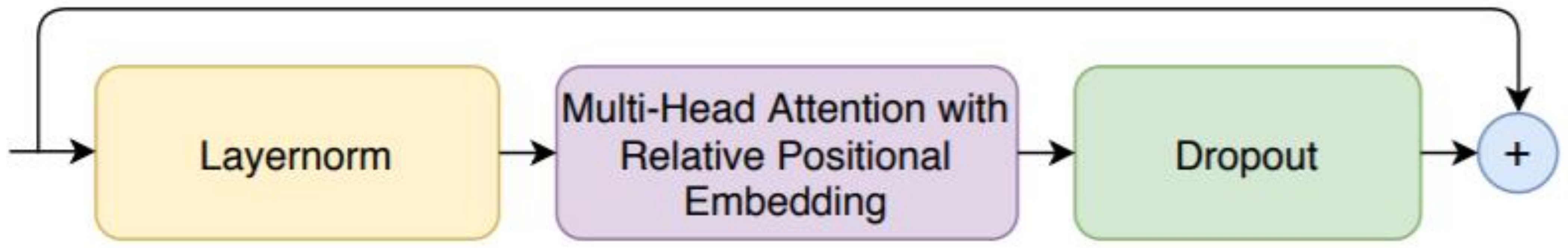


Figure 3: ***Multi-Headed self-attention module.*** *We use multi-headed self-attention with relative positional embedding in a pre-norm residual unit.*

[Ref] Dai, Zihang, et al. "Transformer-XL: Attentive Language Models beyond a Fixed-Length Context." *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019.

Conformer (2020)

- Conformer block
  - It contains two Feed Forward modules sandwiching [9] the MHSA module and the Conv module
  - For input x_i to a Conformer block i, the output y_i of the block is
  - $\bar{x}_i = x_i + \frac{1}{2}\mathrm{FFN}(x_i)$
  - $x_i' = \bar{x}_i + \mathrm{MHSA}(\bar{x}_i)$
  - $x_i'' = x_i' + \mathrm{Conv}(x_i')$
  - $y_i = \mathrm{Layernorm}(x_i'' + \frac{1}{2}\mathrm{FFN}(x_i''))$

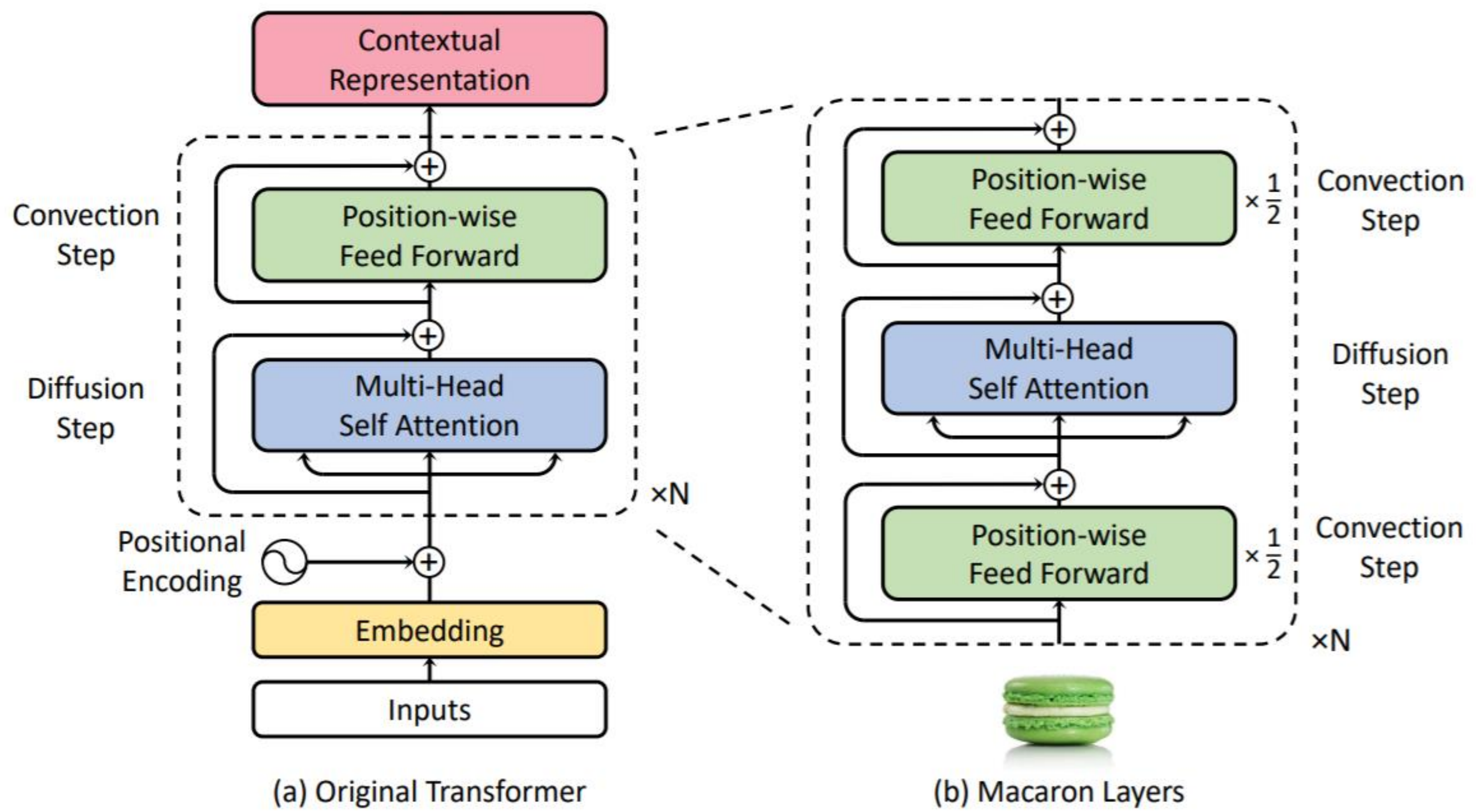


Figure 2: The Transformer and our Macaron architectures.

Conformer (2020)

- Conv Module

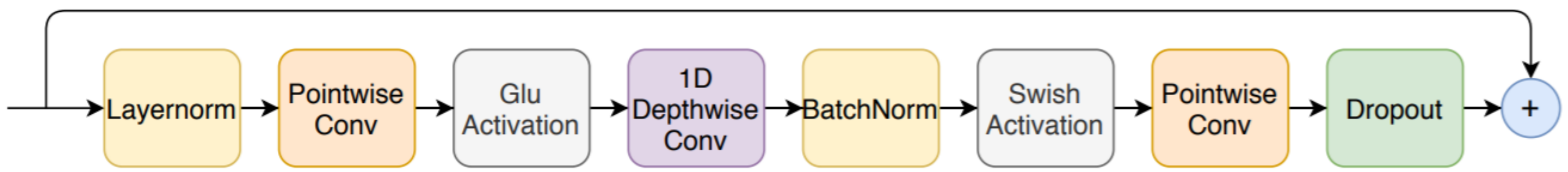


Figure 2: ***Convolution module.*** *The convolution module contains a pointwise convolution with an expansion factor of 2 projecting the number of channels with a GLU activation layer, followed by a 1-D Depthwise convolution. The 1-D depthwise conv is followed by a Batchnorm and then a swish activation layer.*

- FFN Module

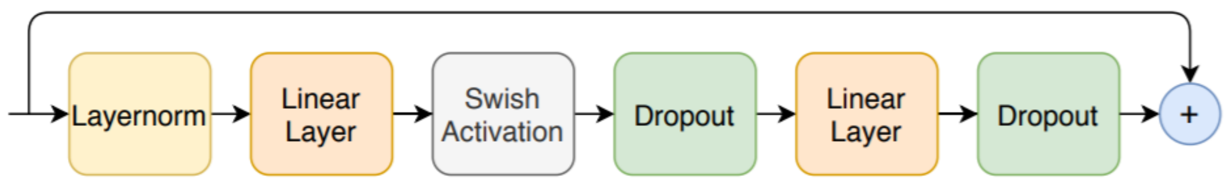
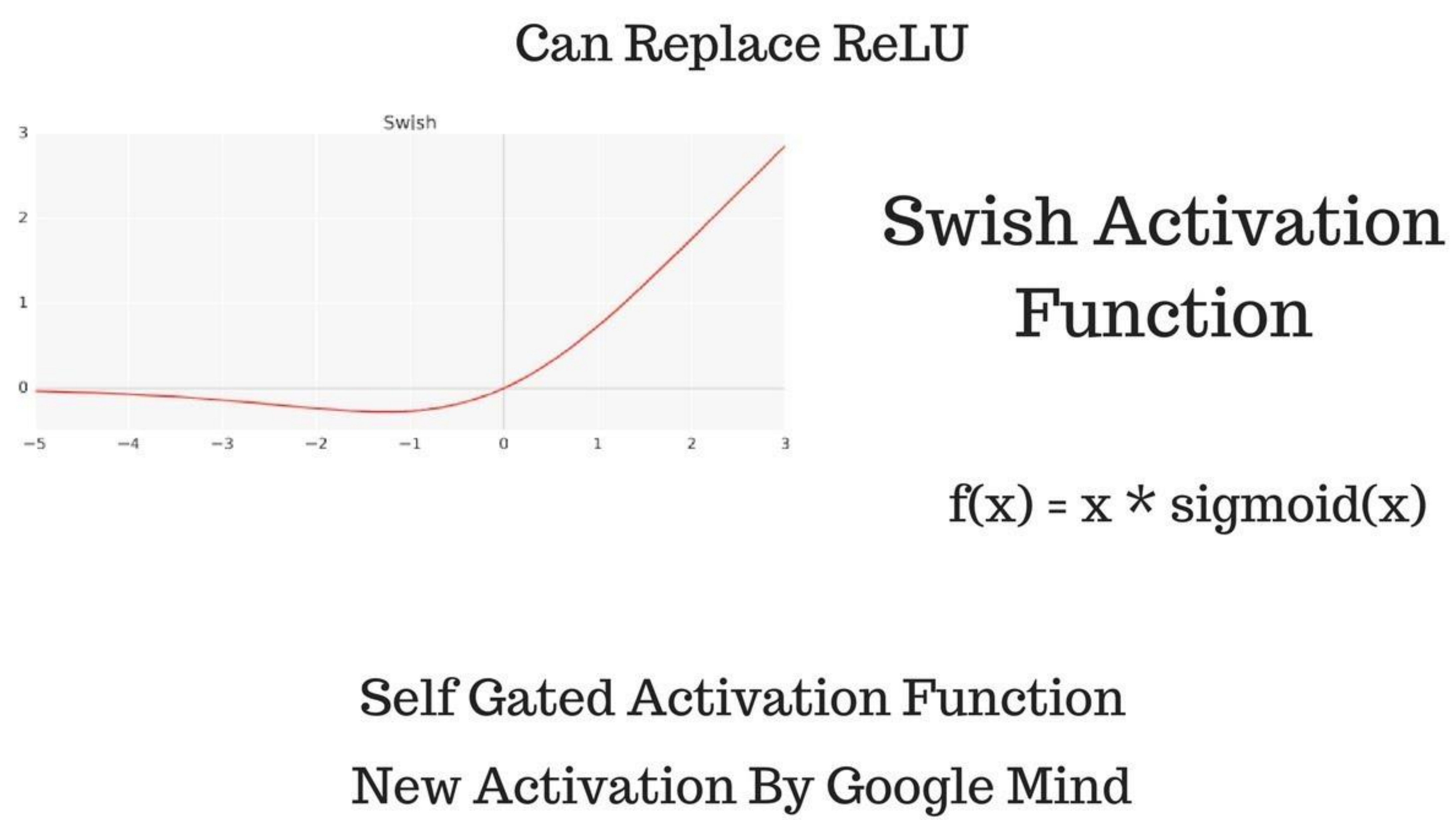


Figure 4: ***Feed forward module.*** *The first linear layer uses an expansion factor of 4 and the second linear layer projects it back to the model dimension. We use swish activation and a pre-norm residual units in feed forward module.*

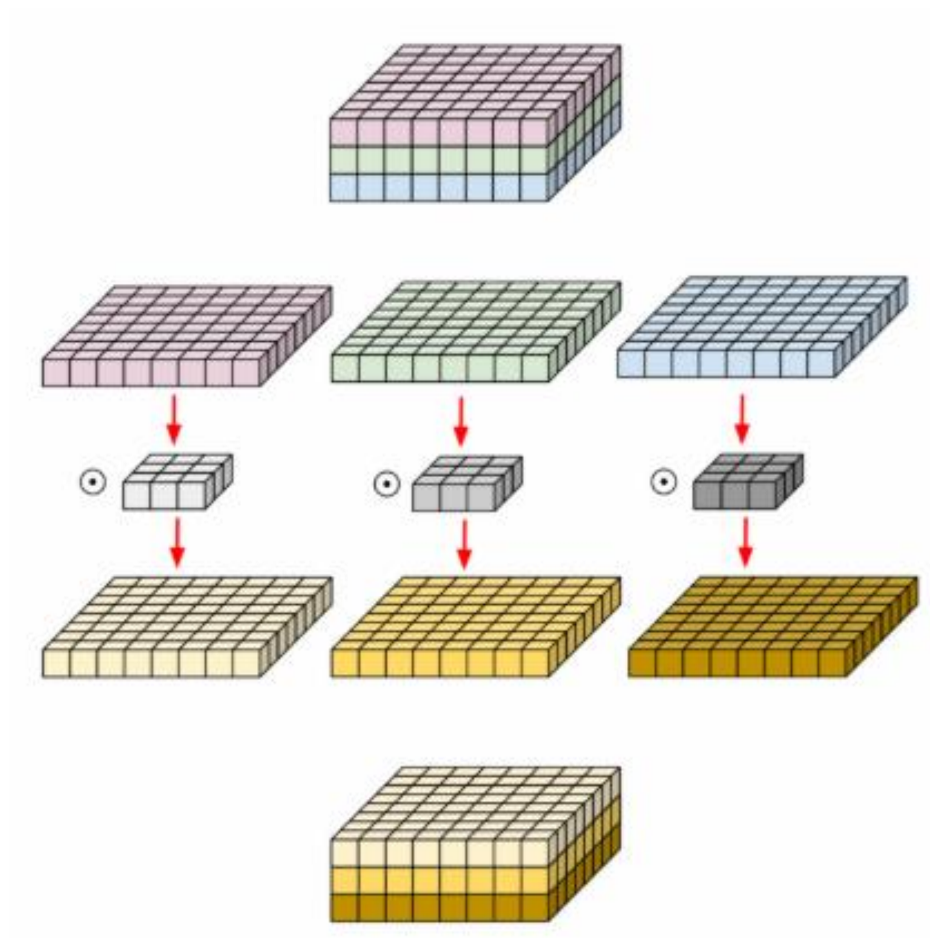
- Swish activation
  - - 로 경계가 있고 + 로는 제한이 없음
  - Relu와 달리 smooth
  - 비 단조함수로써 - 값이 0이 되지 않음

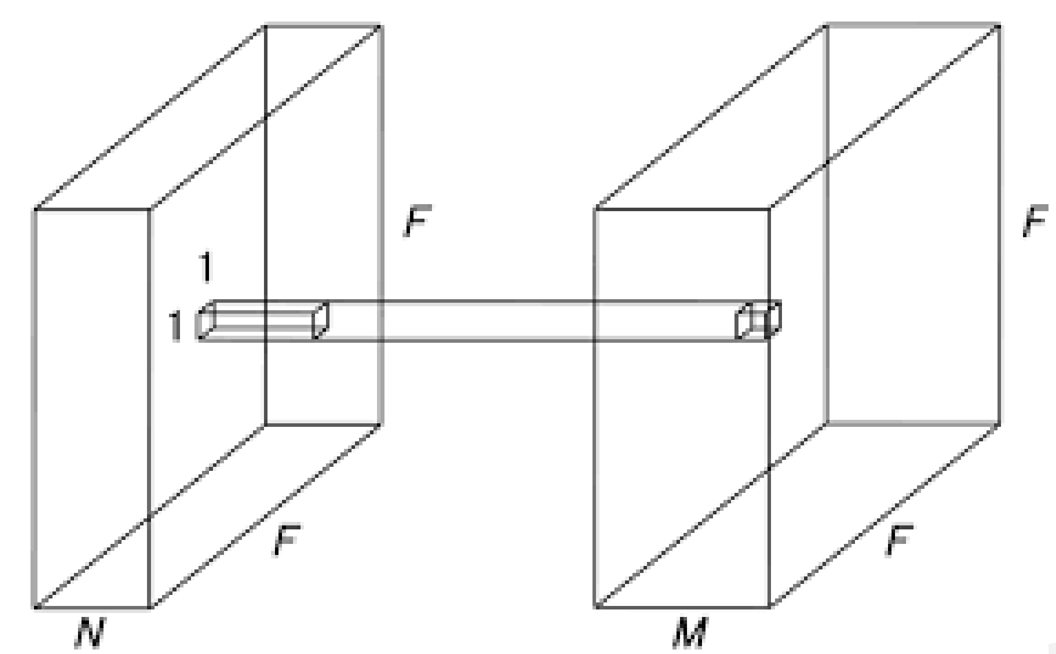


Conformer (2020)

- Depth-wise Conv
  - $H * \mathrm{W} * \mathrm{C}$의 conv output을 C단위로 분리하여 각각 conv filter 적용 → output → 다시 합침
  - 적은 params로 동일한 크기의 output 출력 가능

- Pointwise Conv
  - 1x1 Conv filter
  - 기존의 matrix의 결과를 shuffle해 뽑아내는 것을 목적으로 함
  - Channel 수를 줄이거나 늘리는 목적으로 사용





Conformer (2020)

- Results on LibriSpeech
  - Achieved WER of 2.1%/4.3% without LM
  - 1.9%/3.9% with an external LM

Method	#Params (M)	WER Without LM		WER With LM	
		testclean	testother	testclean	testother
Hybrid					
Transformer [33]	-	-	-	2.26	4.85
CTC					
QuartzNet [9]	19	3.90	11.28	2.69	7.25
LAS					
Transformer [34]	270	2.89	6.98	2.33	5.17
Transformer [19]	-	2.2	5.6	2.6	5.7
LSTM	360	2.6	6.0	2.2	5.2
Transducer					
Transformer [7]	139	2.4	5.6	2.0	4.6
ContextNet(S) [10]	10.8	2.9	7.0	2.3	5.5
ContextNet(M) [10]	31.4	2.4	5.4	**2.0**	4.5
ContextNet(L) [10]	112.7	**2.1**	4.6	**1.9**	4.1
Conformer (Ours)					
Conformer(S)	10.3	**2.7**	**6.3**	**2.1**	**5.0**
Conformer(M)	30.7	**2.3**	**5.0**	**2.0**	**4.3**
Conformer(L)	118.8	**2.1**	**4.3**	**1.9**	**3.9**

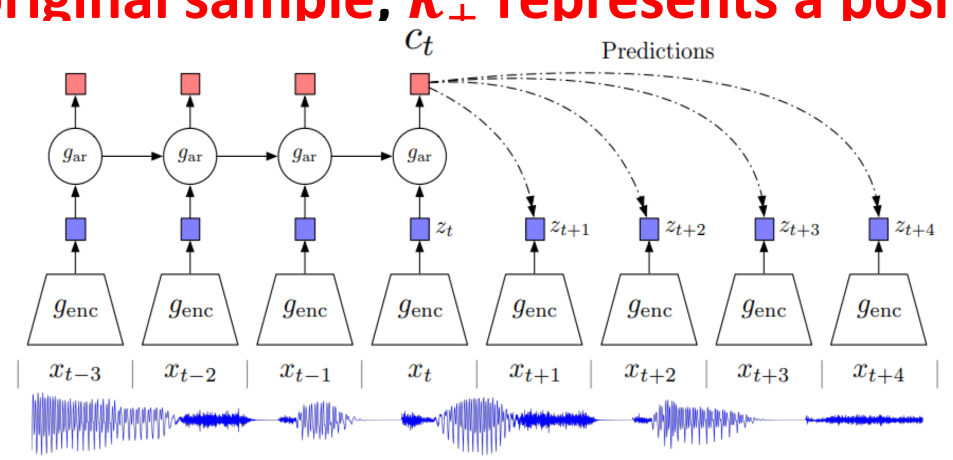
- CPC, wav2vec
  - Given a raw audio signal of length L, CNN-based g_{enc} encodes the audio signals into vector representations
  - An auto-regressive function g_{ar}, such as a RNN, summarizes the past representations and produces context vectors
  - The representations are learned to maximize mutual information between context vectors (c) and future latent representations (z) as follows:

$$\sum_{t,k} I(c_t, z_{t+k}) = \sum_{t,k} \sum_{c_t, z_{t+k}} p(c_t, z_{t+k}|k) \log \frac{p(z_{t+k}|c_t, k)}{p(z_{t+k})}$$

  - CPC objective can be present as follows:

$$L_{NCE} = -\log \frac{\exp\left(\frac{sim(q,k_+)}{\tau}\right)}{\exp\left(\frac{sim(q,k_+)}{\tau}\right) + \exp(\frac{sim(q,k_-)}{\tau})}$$

  - **Where q is the original sample, k_+ represents a positive sample, k_- represents a negative sample, and τ is a hyperparameter**



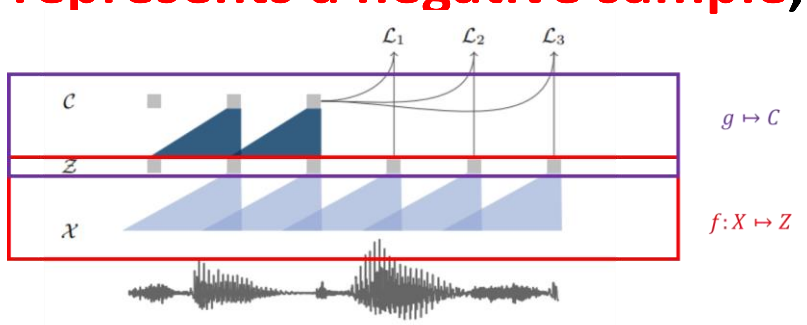
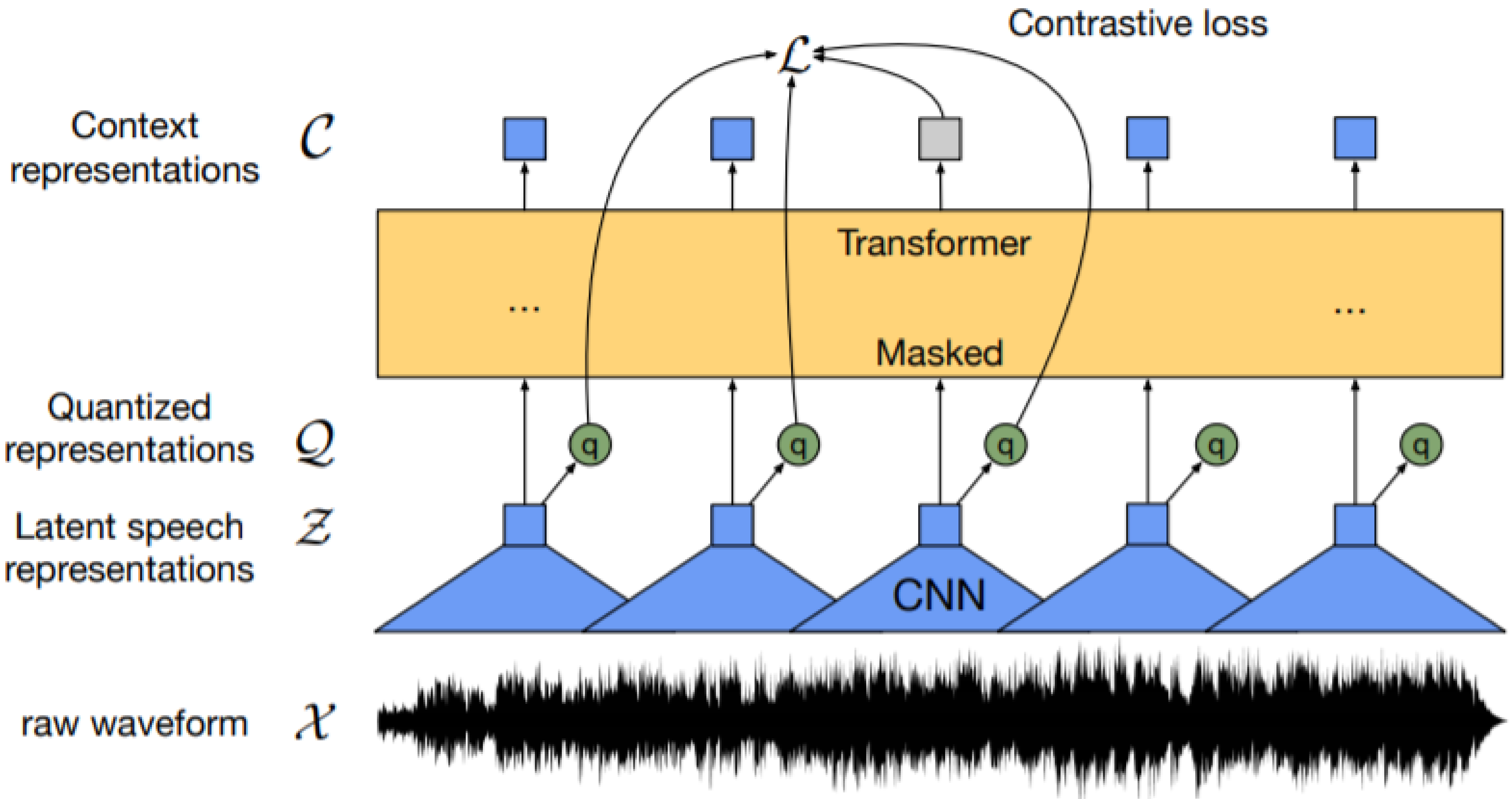


Figure 1: *Illustration of pre-training from audio data $\mathcal{X}$ which is encoded with two convolutional neural networks that are stacked on top of each other. The model is optimized to solve a next time step prediction task.*

Schneider, Steffen, et al. "wav2vec: Unsupervised Pre-Training for Speech Recognition." *Proc. Interspeech 2019* (2019): 3465-3469.

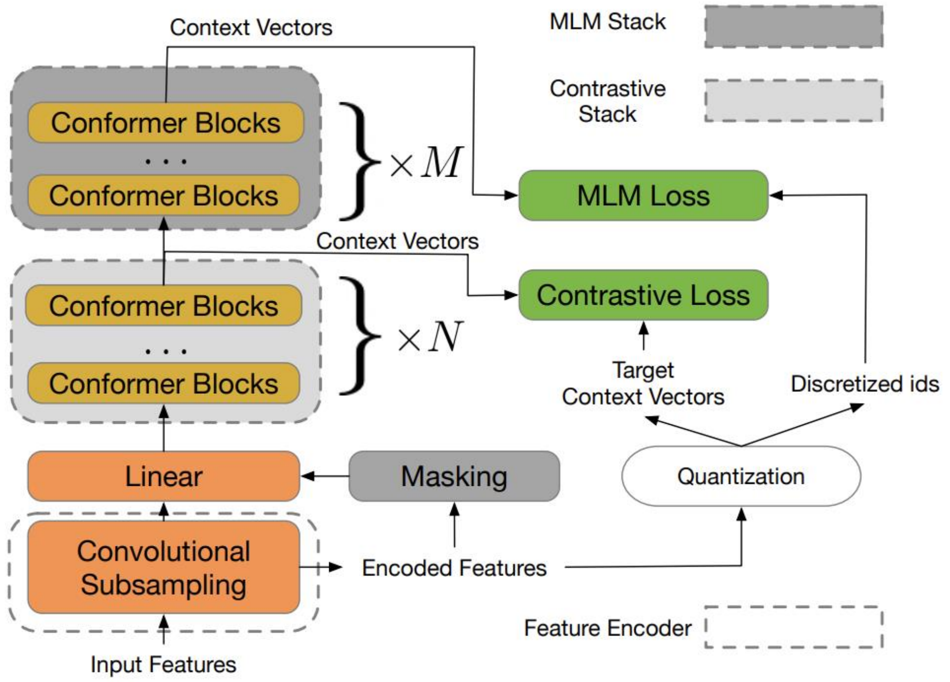
wav2vec 2.0 (2020)

- Using contrastive learning method to get powerful representation for self-supervised learning
  - Adapting the masking method (BERT) to the speech input in the latent space
  - Using CNN + transformer architecture for representation learning
  - Recording LibriSpeech clean 4.8% and noisy 8.2% Word Error Rate (WER) with just 10 minutes of data (40 utterances)



Baevski, Alexei, et al. "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations." *Advances in Neural Information Processing Systems* 33 (2020).

w2v-BERT (2021)



- Feature encoder: two 2D-convolution layers both with stride=(2,2)
  - Given a log-Mel spectrogram as input, resulting in a 4x reduction in the acoustic input's sequence length
- Contrastive module (wav2vec 2.0)
  - Discretizes the feature encoder output into a finite set of representative speech units

$$L_c = L_w + \alpha \cdot L_d$$

- Masked prediction module (BERT)
  - **Extracts high-level contextualized speech representations** using **Context Vectors** produced by the contrastive module
  - Predicts masked position with its corresponding token ID (Cross-entropy loss)

$$L_m = -\sum_{i=1}^{N} (\text{token} - \text{ID})_i \log(\text{Conformer}_{\text{M}}((\text{Context Vectors})_i))$$

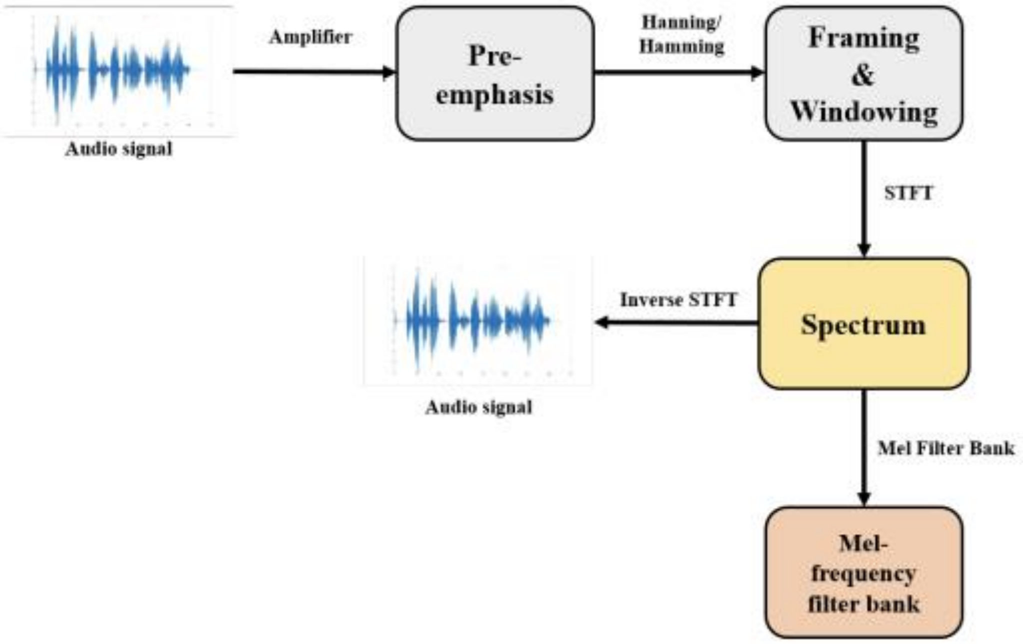
Chung, Yu-An, et al. "W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training." *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2021.

Data preprocessing

Wav and spectrogram

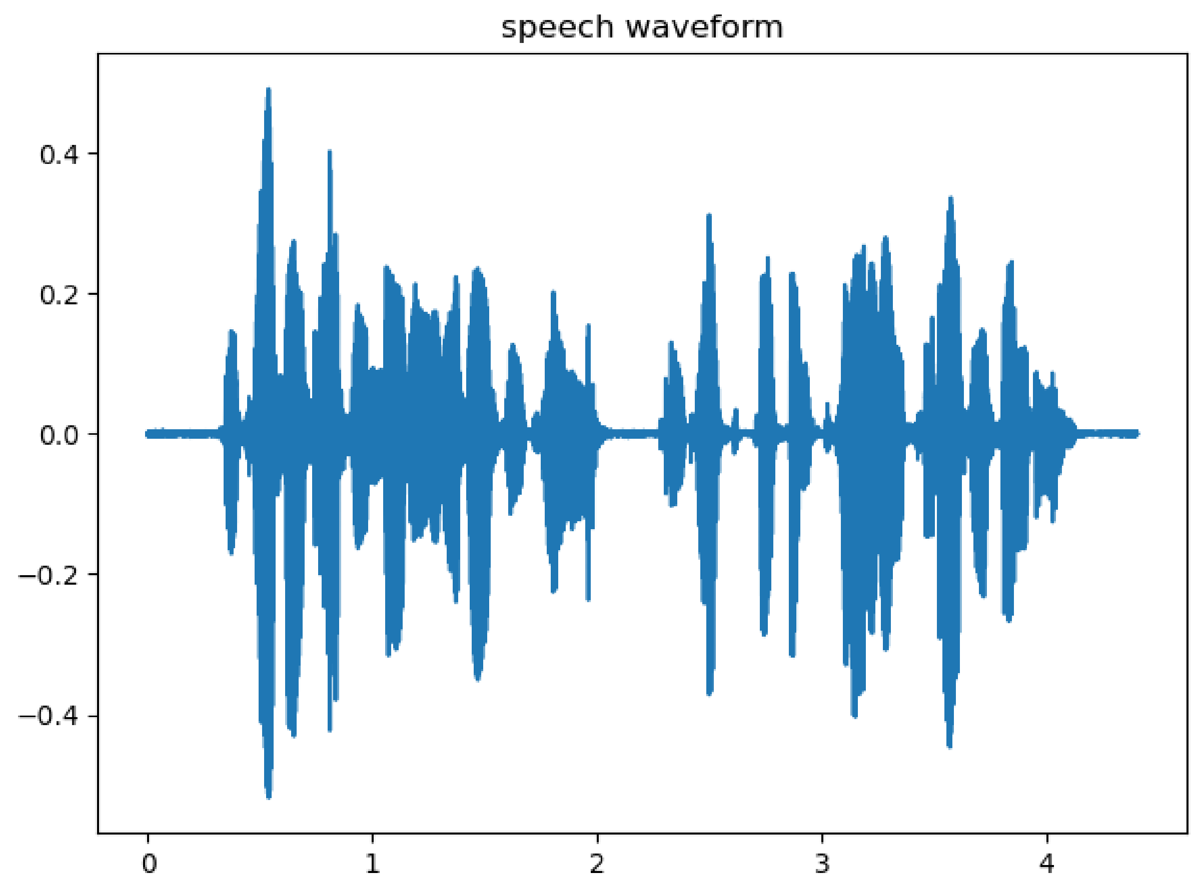
Pre-processing

- Wav – Spectrogram – Mel Fbank



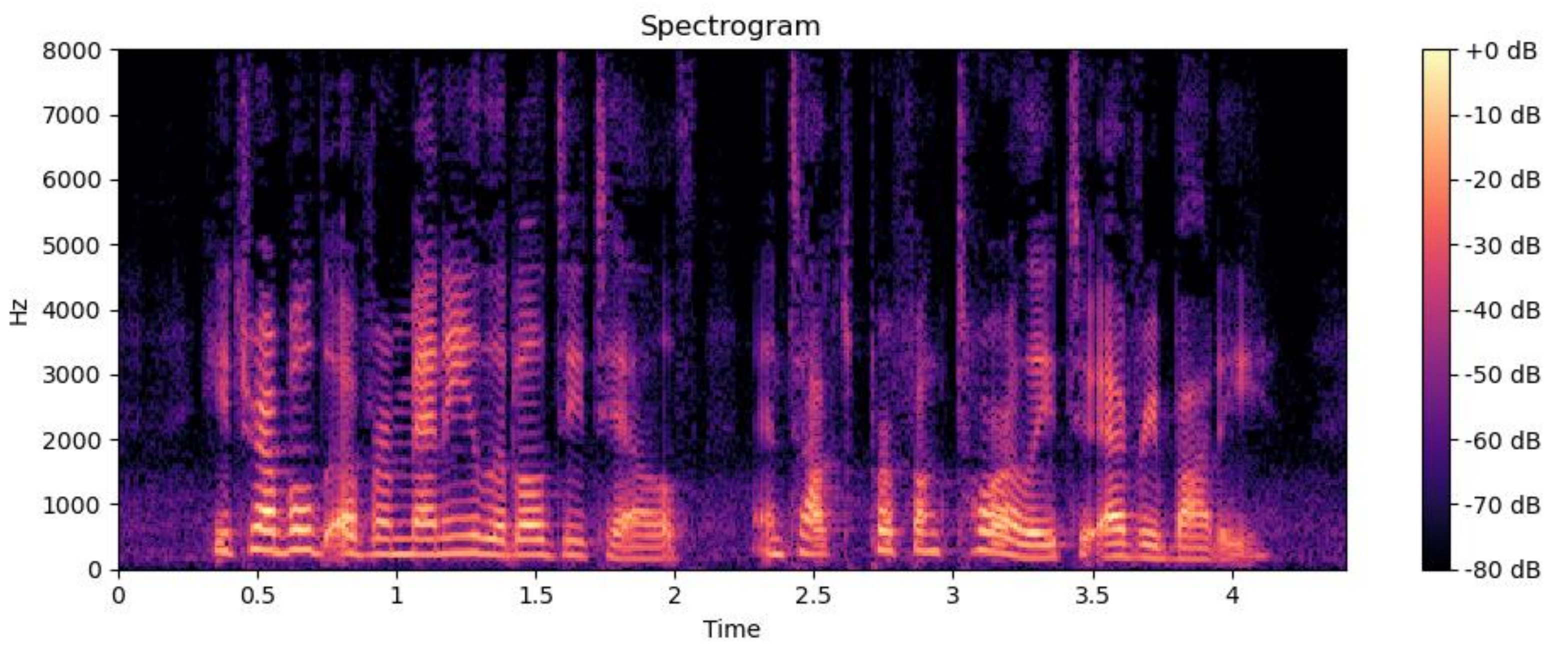
Speech pre-processing

- Read waveform
  - 16000Hz -> 1 seconds (16000)



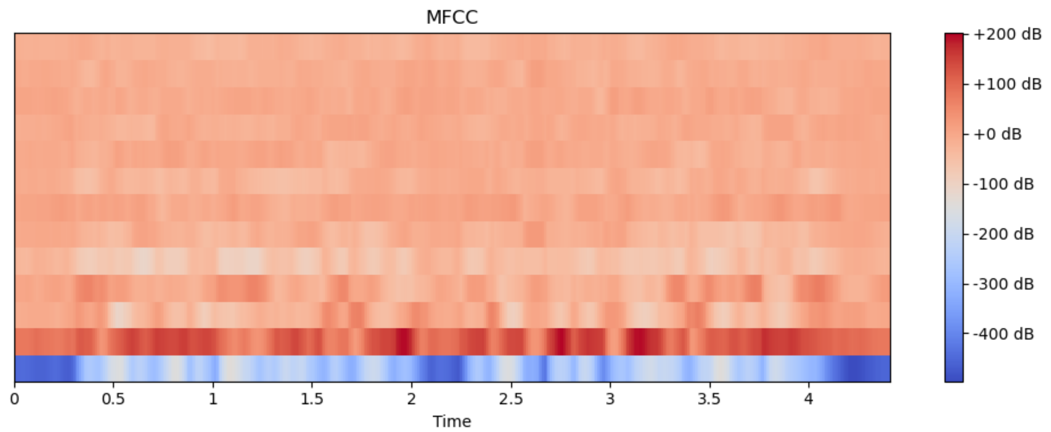
Speech pre-processing

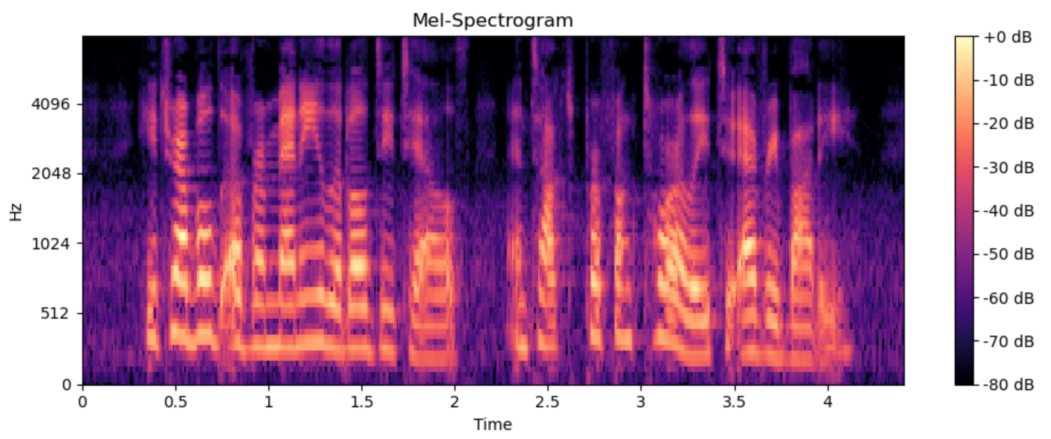
- FFT: Fast Fourier Transform
  - STFT: Short-time Fourier Transform
  - We use librosa, soundfile, torchaudio, etc



Speech pre-processing

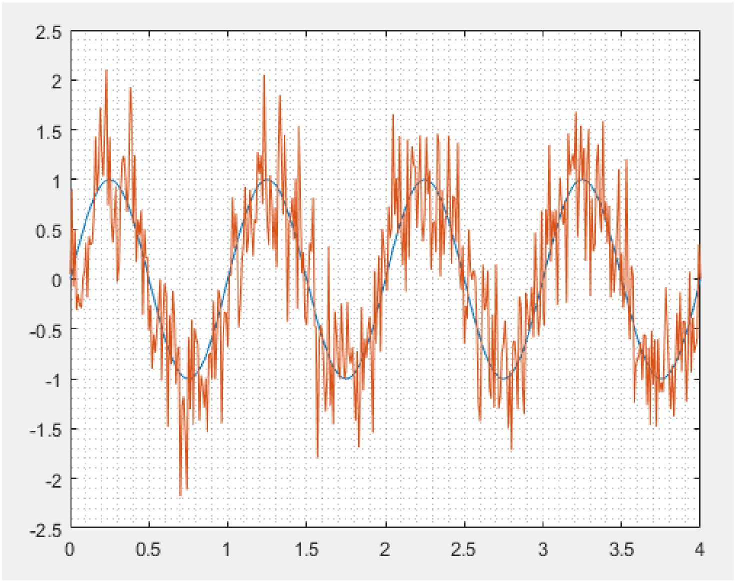
- Mel scale: 사람의 귀를 모방하여 만든 scale
  - MFCC, Mel-spectrogram

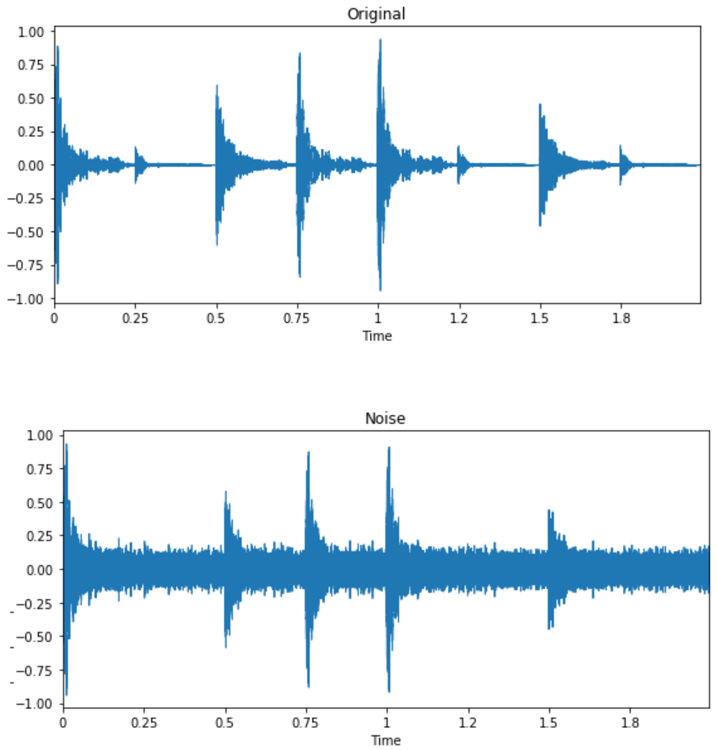




Data augmentation

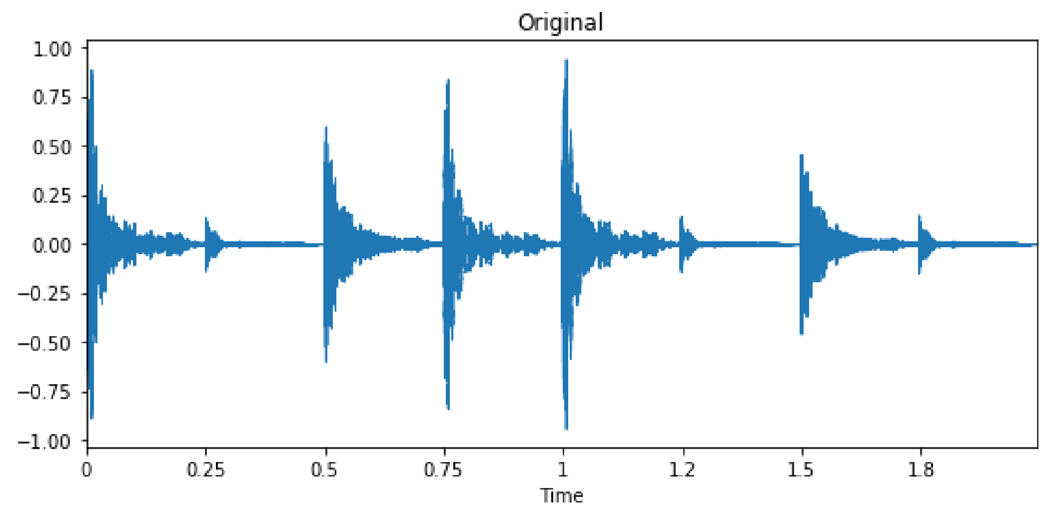
- Augmentation
  - Noise injection
  - 단순하게 numpy library를 사용하여 데이터에 임의의 값 추가

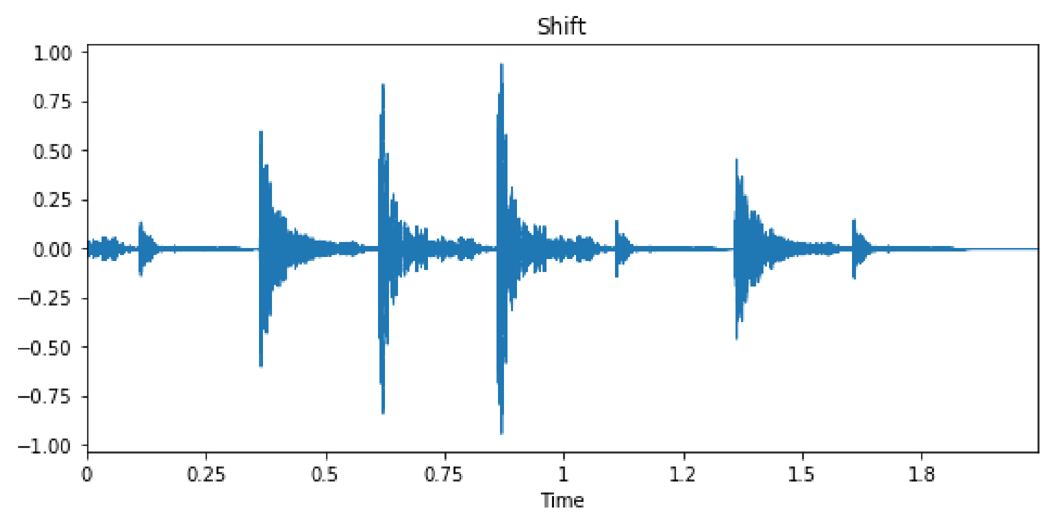




Data augmentation

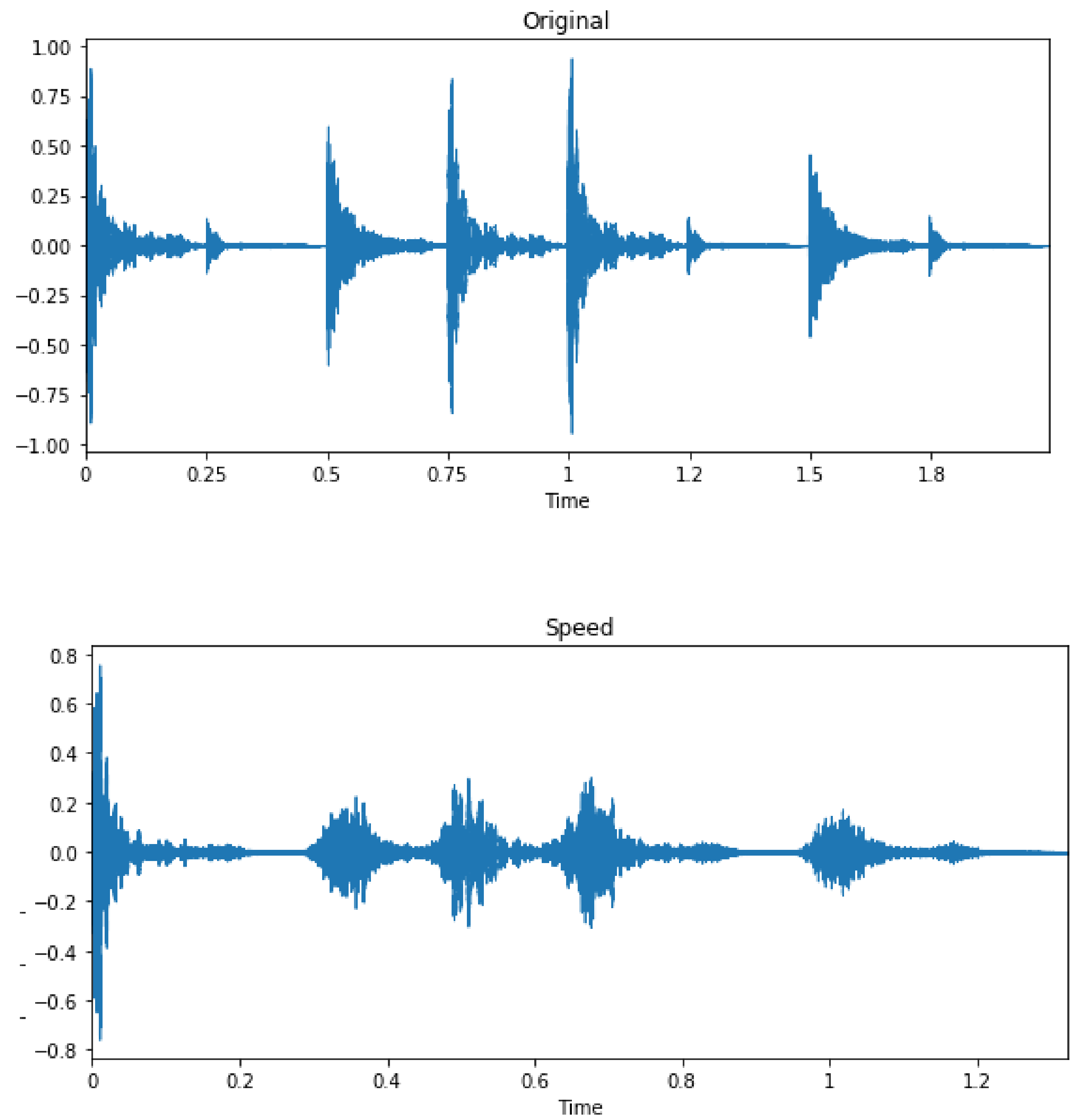
- Augmentation
  - Shifting time
    - 임의의 초로 오디오를 왼쪽 / 오른쪽으로 이동함
    - n초를 사용하여 오디오를 왼쪽 (빨리 감기)으로 이동하면 처음 n초는 0(무음)으로 표시됨
    - n초를 사용하여 오디오를 오른쪽 (뒤로 앞으로)으로 이동하면 마지막 n초가 0(무음)으로 표시됨





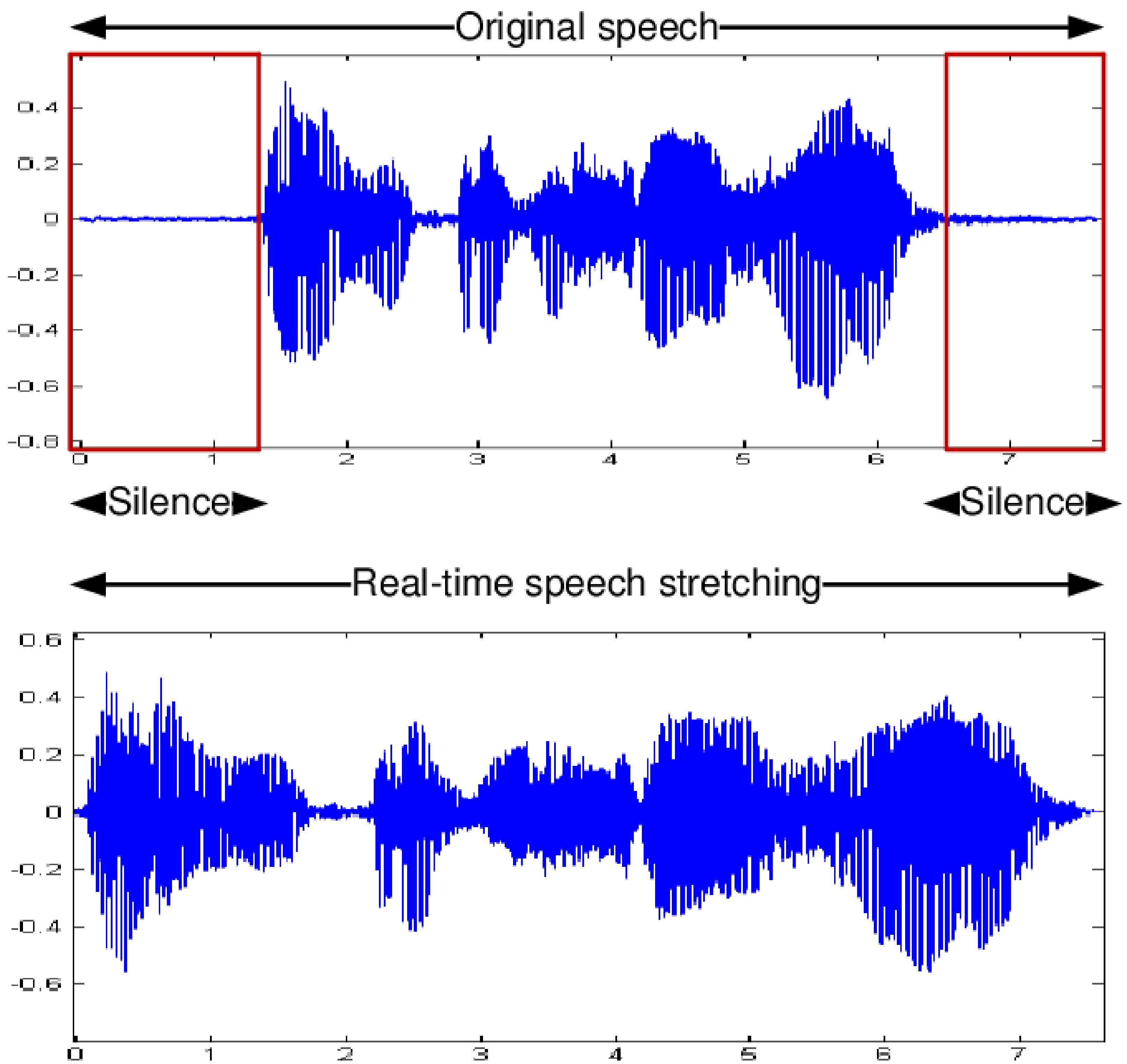
Data augmentation

- Augmentation
  - Changing speed
    - 고정 비율로 시계열을 늘릴 수 있음



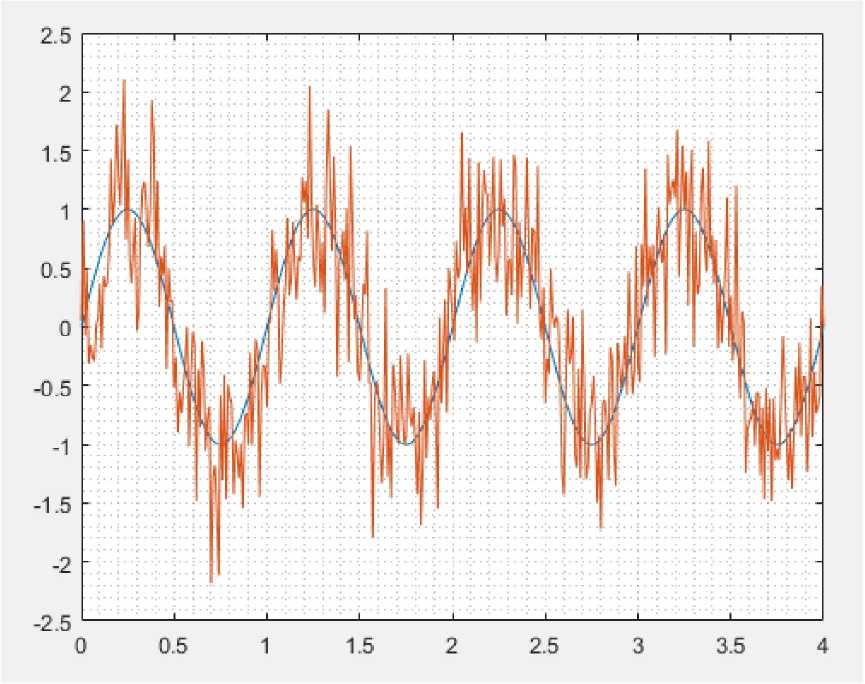
Data augmentation

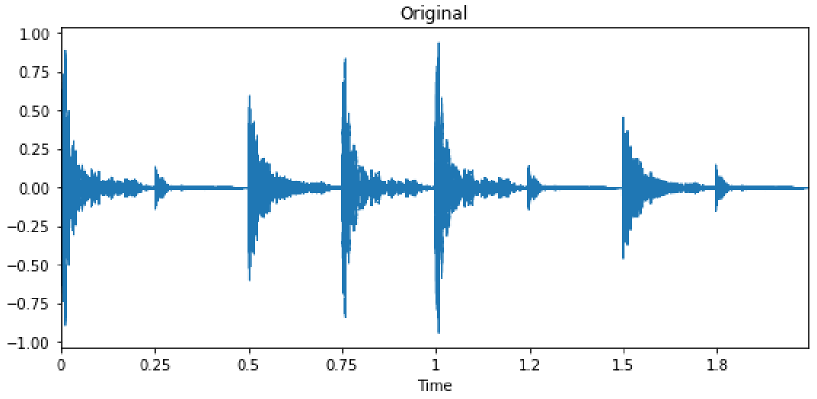
- Augmentation
  - Time stretching

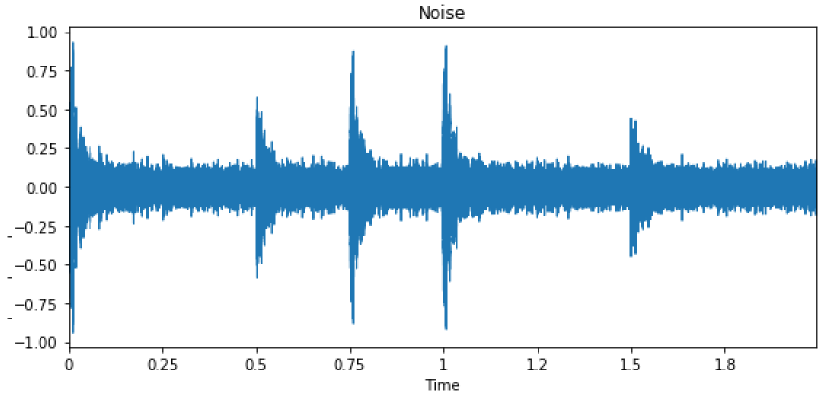


Data augmentation

- Additive Gaussian Noise

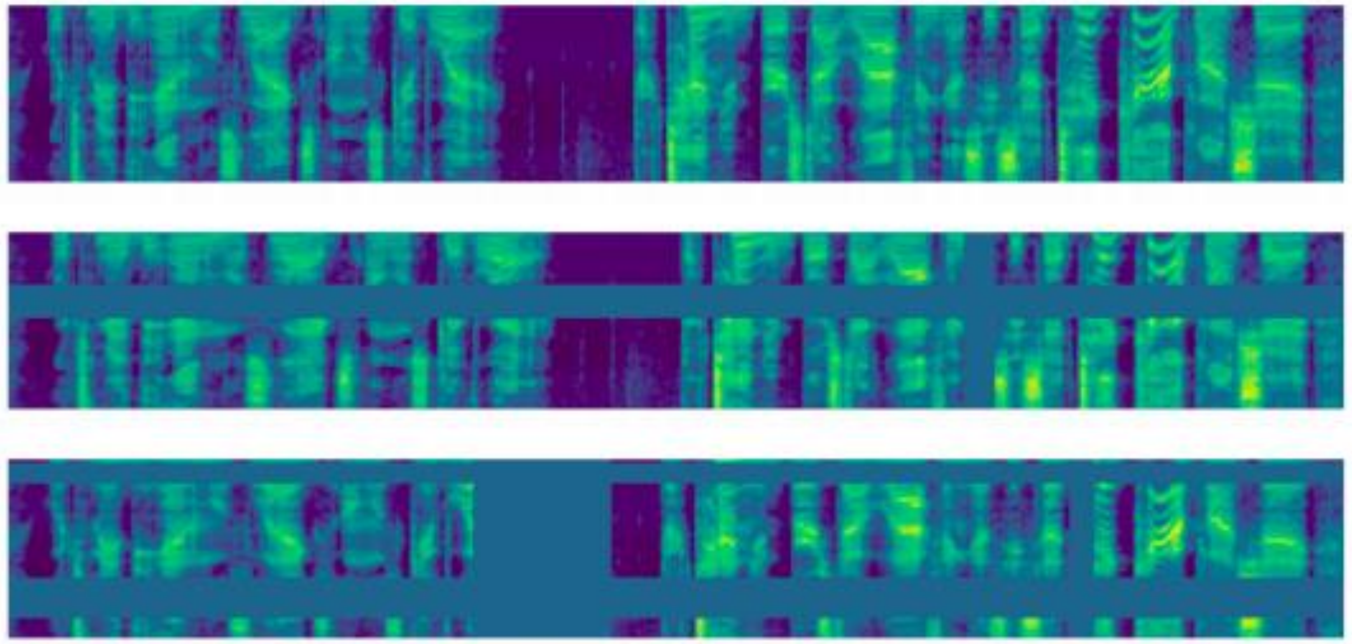






Data augmentation

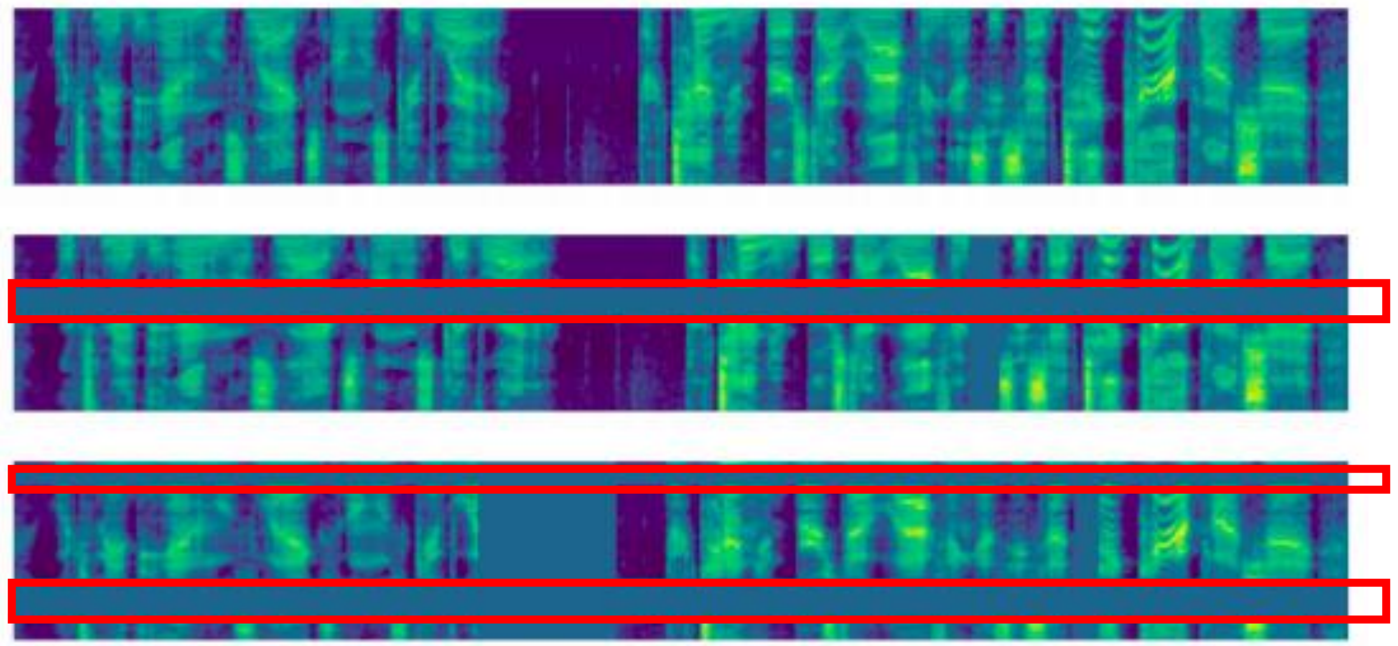
- Augmentation
  - SpecAugment



Park, Daniel S., et al. "Specaugment: A simple data augmentation method for automatic speech recognition." *arXiv preprint arXiv:1904.08779* (2019).

Data augmentation

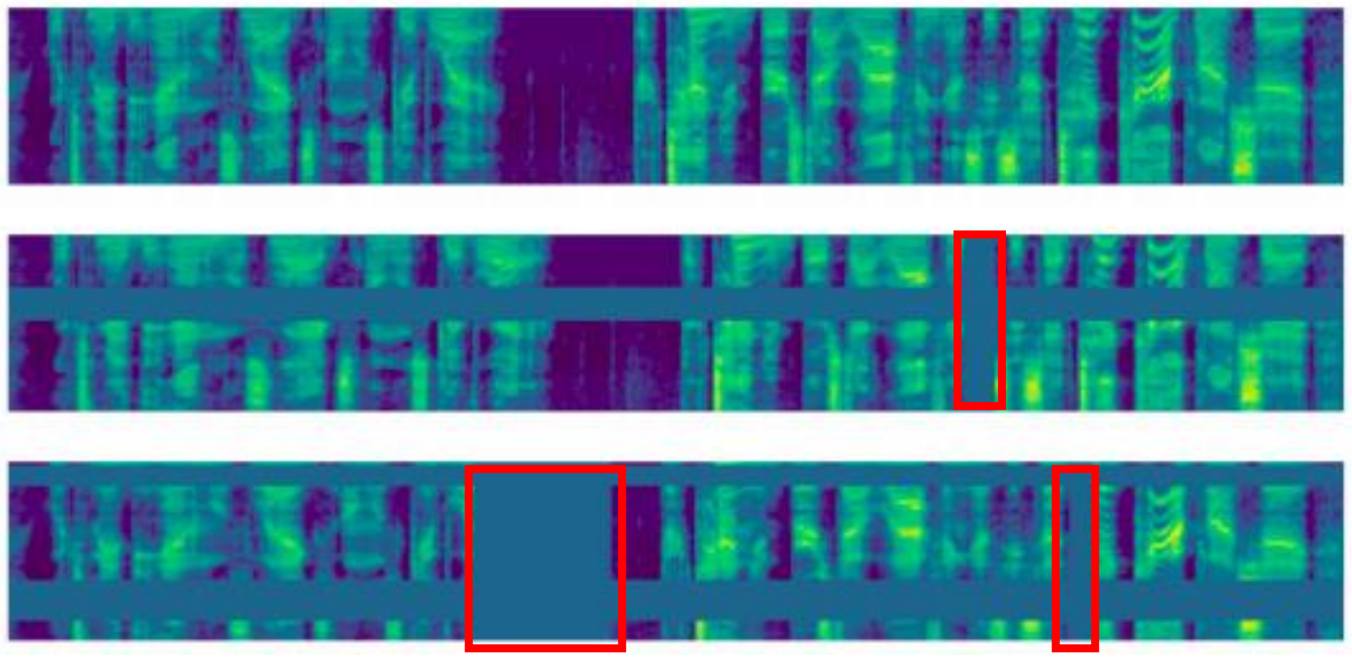
- Augmentation
  - SpecAugment
    - 임의의 주파수 축에 0 값으로 Masking



Park, Daniel S., et al. "Specaugment: A simple data augmentation method for automatic speech recognition." *arXiv preprint arXiv:1904.08779* (2019).

Data augmentation

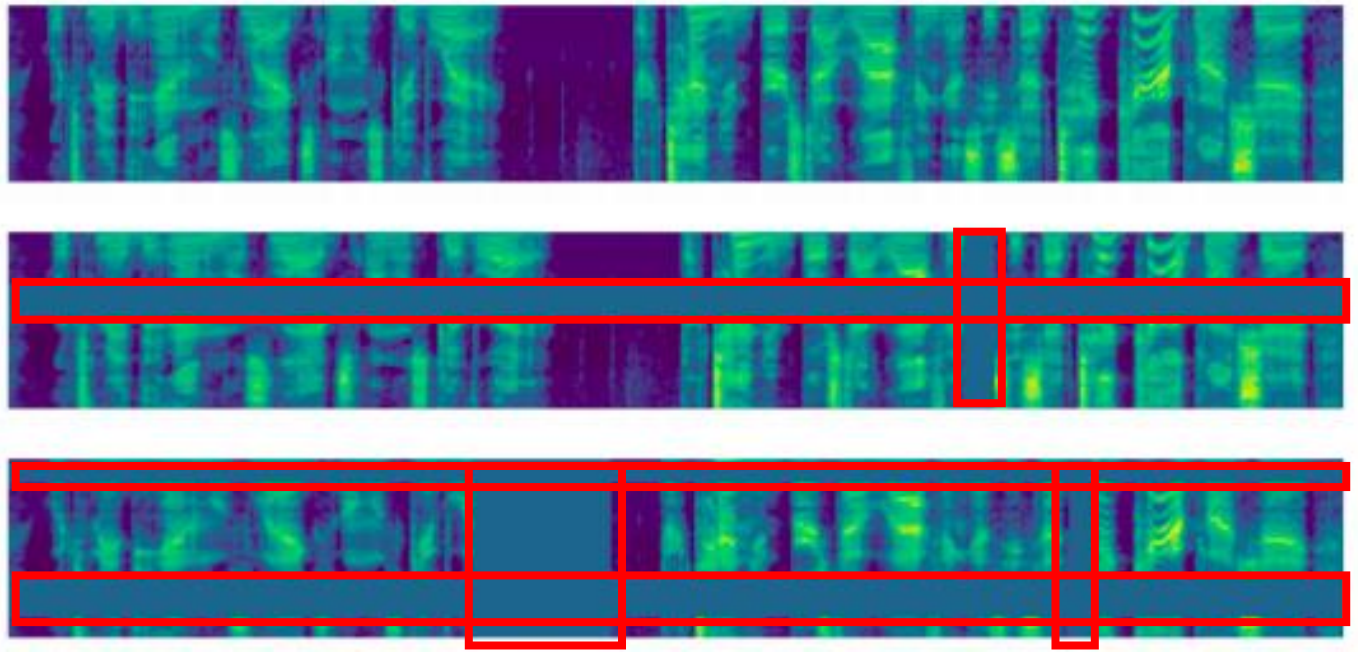
- Augmentation
  - SpecAugment
    - 임의의 시간 축에 0 값으로 Masking



Park, Daniel S., et al. "Specaugment: A simple data augmentation method for automatic speech recognition." *arXiv preprint arXiv:1904.08779* (2019).

Data augmentation

- Augmentation
  - SpecAugment
    - 주파수 + 시간축 0값 Masking
    - 각 1번 / 2번의 조합

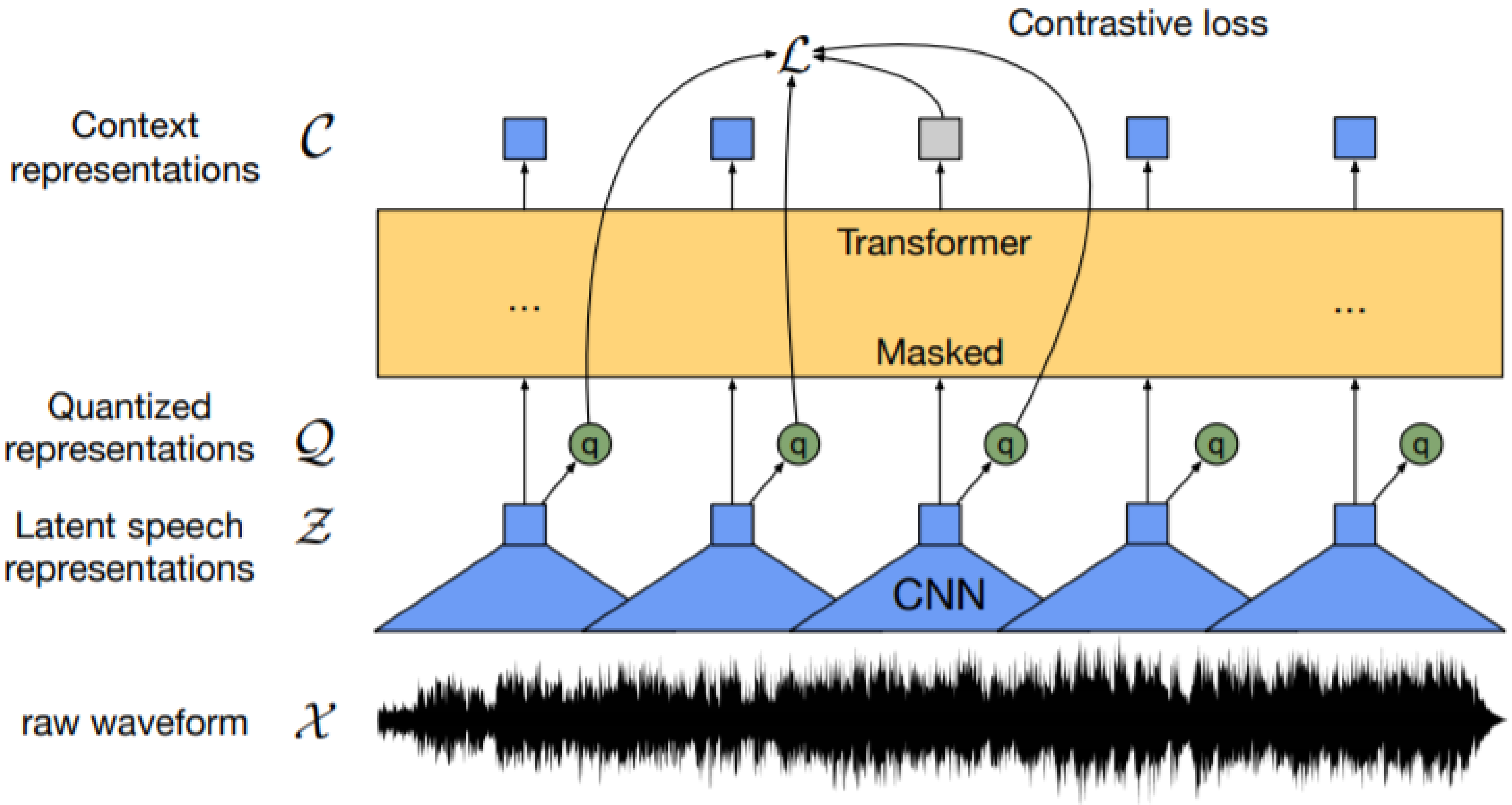


Park, Daniel S., et al. "Specaugment: A simple data augmentation method for automatic speech recognition." *arXiv preprint arXiv:1904.08779* (2019).

Pre-trained wav2vec 2.0을 활용한 음성인식 실습

wav2vec 2.0 (2020)

- Using contrastive learning method to get powerful representation for self-supervised learning
  - Adapting the masking method (BERT) to the speech input in the latent space
  - Using CNN + transformer architecture for representation learning
  - Recording LibriSpeech clean 4.8% and noisy 8.2% Word Error Rate (WER) with just 10 minutes of data (40 utterances)



Baevski, Alexei, et al. "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations." *Advances in Neural Information Processing Systems* 33 (2020).

wav2vec 2.0 실습

- 목표
  - 영어 기반으로 학습된 공개된 wav2vec 2.0 모델을 활용하여 음성인식 진행

wav2vec 2.0 실습

- https://colab.research.google.com/drive/15LN8JMXgCZCHR4PEhu_BsuiRcnSS-1xc?usp=sharing

- 혹은 kaen2891 tistory 블로그 검색

Thank you!

Presentation finished