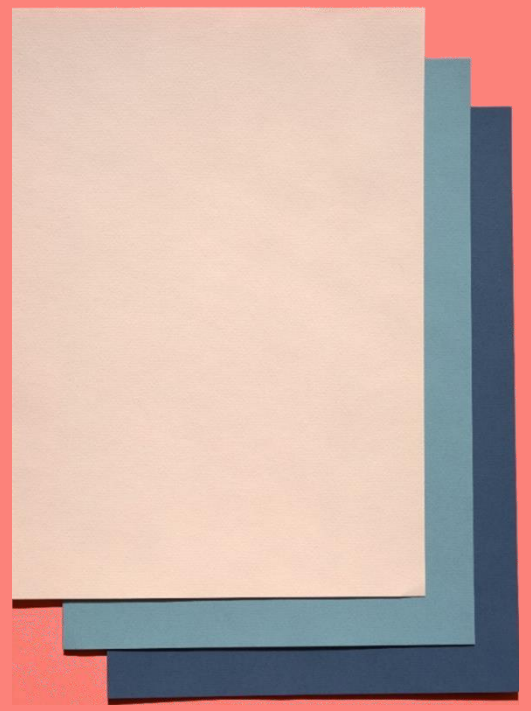
2021 외국인 발화
한국어 음성 인식률 향상 해커톤

너의 목소리가 들려 - 김다희, 오형석, 이현진

목 차



1. 음성인식 공부

1. 음성인식 공부 _SKT 음성인식 강의

음성인식 과정

음성임을 인식

사람의 목소리 특정하는 주파수 추출 및 잡음 제거

특징 추출

녹음된 음성의 변화 특징 수치화 후 특징벡터 추출

딥러닝을 통한 음향 모델링

학습데이터의 경우는 수작업으로 확보!

특징 추출로 얻은 벡터와의 비교는 개별 음소에 매칭

=> 언어 모델을 이용하여 음성 인식결과가 문맥에 맞는 단어로 매칭되도록 통계적 확률값 구하기

많은 문장이 언어모델로 학습될수록 문맥 고려한 음석 인식 ↑

음성인식 방식

- 실시간 처리 가능한 sequential 방식(회사)
- 성능 좋은 batch 방식(학교, 연구소)

현재

- 발화된 시그널이 있고 그 시그널에 대한 레이블 정보만 있다면 모델링 한꺼번에 하는 End to End 방법으로 발전 중

성능 측정

WER : Word Error Rate 사람의 경우 WER 4%정도 -> WER 10% 이하의 음성인식 기술이 상용화 가능

1. 음성인식 공부_SpeechRecognition

Speech To Text(STT)

- STT는 음성을 입력받아 그에 해당하는 구문(문자열)을 얻는 기술
- 딥러닝을 이용한 STT 모델에는 DeepVoice, WaveNet 등 존재
- 음성 인식 기술은 음성 데이터가 대부분 큼
- 관련 기술 코드도 상당히 길어 구현이 힘듬
- SpeechRecognition으로 모델을 학습시키지 않고 간편한 구현 가능

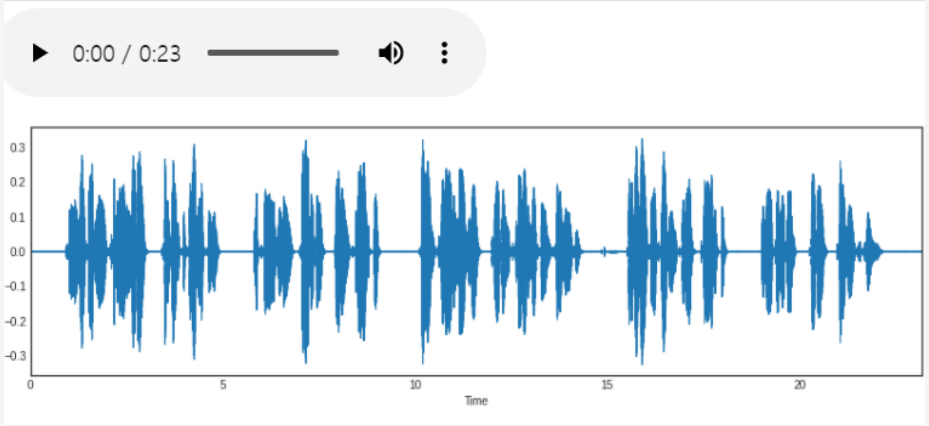
- 여러 기업에서 제공하는 음성 인식 기술 사용 가능
  - `recognize_google()` - Google Web Speech API
  - `recognize_google_cloud()` - Google Cloud Speech
  - `recognize_bing()` - Microsoft Bing Speech
  - `recognize_houndify()` - SoundHound Houndify
  - `recognize_ibm()` - IBM Speech to Text
  - `recognize_wit()` - Wit.ai
  - `recognize_sphinx()` - CMU Sphinx (Sphinx는 오프라인으로 동작하며, 나머지 모든 함수는 인터넷 연결이 되어야만 사용 가능)

```
!pip install SpeechRecognition

import speech_recognition as sr

r = sr.Recognizer()

fig = plt.figure(figsize=(14, 4))
korean_wav, rate = librosa.core.load('korean.wav')
librosa.display.waveplot(korean_wav, sr=rate)
ipd.Audio(korean_wav, rate=rate)
```




```
korean_audio = sr.AudioFile('korean.wav')

with korean_audio as source:
  audio = r.record(source)

r.recognize_google(audio_data=audio, language='ko_KR')
```

```
'죽는 날까지 하늘을 우러러 한점 부끄럼이 없기를 잎새에이는 바람에도 나는 괴로워했다
별을 노래하는 마음으로 모든 죽어가는 것을 사랑해야지 그리고 나한테 주어진 길을 걸어
야겠다 오늘 밤에도 별이 바람에 스치운다'
```

1. 음성인식 공부 _Zeroth ENTERPRISE

Zeroth 칼디(Kaldi)를 기반
실시간 모듈, 2,000시간 음향 모델을 지원
어떠한 비즈니스에도 최적화하여 적용할 수 있는 음성인식 솔루션

음성인식 | Automatic Speech Recognition

- STT(Speech-to-Text)
- 음성 스트리밍 인식 (Speech Stream Recognizing)
- 멀티 채널 지원 (Multi-channel Support)
- 구문 힌트 (Phrase Hints)

자연어 이해 | Natural Language Understanding

- 개체명 추출 (Named Entity Extraction)
- 의도 감지 (Intent Detection)
- 유형 분류 (Document Classification)
- 검색 기반 추천 답변 시스템 (Retrieval-based Recommender System)

강점

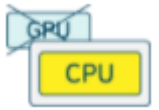

도메인 최적화

실시간 음성 인식


다양한 결과 출력 형식


최신 기술 업데이트 가능


경량화 시스템



CPU만으로 디코딩 가능

1. 음성인식 공부_Kospeech

2020 국어 정보 처리 시스템 경진 대회 일반분야 참가 팀 SOTA-X의 출품작
End-to-End 한국어 음성인식 툴킷 'KoSpeech'

https://github.com/sooftware/KoSpeech

pytorch 기반의 딥러닝 모델로, 한국어만 지원

최근까지 업데이트 및 에러 fix 지속적
개발자와의 소통 원활

전처리를 지원

데이터의 특징에 따라
전사규칙을 삭제하고
문장 부호(?!,. 등)를 삭제하는 등의
전처리를 진행 과정 공개

End to End 모델

음소나 음절의 처리, 문법, 발음 등을 모두 학습

Deep Speech 2, Speech Transformer 등 다양한 알고리즘 기반의 모델을 지원

Supported Models

Acoustic Model	Notes	Citation
Deep Speech 2	2D-invariant convolution & RNN & CTC	Dario Amodei et al., 2015
Listen Attend Spell (LAS)	Attention based RNN sequence to sequence	William Chan et al., 2016
Joint CTC-Attention LAS	Joint CTC-Attention LAS	Suyoun Kim et al., 2017
RNN-Transducer	RNN Transducer	Ales Graves. 2012
Speech Transformer	Convolutional extractor & transformer	Linhao Dong et al., 2018
Jasper	Fully convolutional & dense residual connection & CTC	Jason Li et al., 2019
Conformer	Convolution-augmented-Transformer	Anmol Gulati et al., 2020

2. KoSpeech 공부

2. KoSpeech 공부

(1) Prepare

```
# google mount
from google.colab import drive
drive.mount('/content/drive')

# KoSpeech git clone
!git clone https://github.com/sooftware/kospeech

# LAS git clone
!git clone https://github.com/switiz/las.pytorch
```

- https://github.com/sooftware/kospeech
- https://githubmemory.com/repo/switiz/deepspeech2.pytorch
- https://github.com/switiz/las.pytorch

- 구글 마운트 및 깃 클론 후 테스트용으로 사용할 데이터셋을 업로드 하는 준비 단계
- KsponSpeech: AIHub에서 다운받은 데이터셋
- Pretrained.pt: LAS 모델 학습 시 가져올 pre-trained weight 파일

2. KoSpeech 공부

(2) Preprocess

```
[ ]  # directory setting
     %cd /content/kospeech/dataset/kspon

     # install files
     !pip install -U -r requirements.txt
     !pip install -e .
     !pip install .

     # preprocess
     !bash preprocess.sh
```

	id	char	freq
0	0	<pad>	0
1	1	<sos>	0
2	2	<eos>	0
3	3		525
4	4	.	57
...	...	...	...
317	317	갚	1
318	318	갖	1
319	319	갔	1

- 음성 데이터의 transcripts.txt 및 aihub_labels.csv 파일을 생성하는 전처리 단계
- preprocess.sh 파일에서 본인 환경에 맞는 경로 수정 필요
- 전처리가 끝나면 우측 사진과 같은 라벨링 확인 가능

2. KoSpeech 공부

(3) Configuration

```
# directory setting
%cd /content/las.pytorch

# install packages
!pip install -U PyYAML
!pip install Levenshtein
!pip install FullLoader
```

config.yaml

```
#traiin dataset
root :  /content/drive/MyDrive/icanhearyourvoice/OurSpeech
script_data_path: /content/kospeech/dataset/kspon/transcripts.txt
vocab_path: /content/drive/MyDrive/icanhearyourvoice/OurSpeech/aihub_labels.csv

#train
batch_size: 8
epochs: 5
use_cuda: True
cer_every: 500
teacher_forcing_step: 0.005
teacher_forcing_ratio: 0.99
min_teacher_forcing_ratio: 0.7
label_smoothing: 0.1

#model_save_load
resume: False
save_every: 1
#inference
inference: False
use_val_data: True
weight_path: /content/drive/MyDrive/icanhearyourvoice/pretrained.pt
```

- 우리의 학습 환경에 맞게 config.yaml 파일을 수정하는 설정 단계
- root: dataset root 디렉토리 경로
- Script_data_path: preprocess로 생성된 script 파일 경로
- Vocab_path: preprocess로 생성된 vocab 파일 경로
- weight_path: pre-trained model의 가중치 파일 경로

2. KoSpeech 공부

(4) Train

```
!python train.py
#!python train.py --resume # 이어서 학습할 시
```

```
=========================================
[2021-11-26 05:26:37] - Train : epoch 4 loss:187.49 train_cer:3.02 val_cer:0.00
=========================================
```

```
-----------------------------------
ListenAttendSpell(
  (listener): Listener(
    (pLSTM_layer0): pBLSTMLayer(
      (BLSTM): LSTM(80, 256, batch_first=True, bidirectional=True)
    )
    (pLSTM_layers): ModuleList(
      (0): pBLSTMLayer(
        (BLSTM): LSTM(1024, 256, batch_first=True, dropout=0.1, bidirectional=True)
      )
      (1): pBLSTMLayer(
        (BLSTM): LSTM(1024, 256, batch_first=True, dropout=0.1, bidirectional=True)
      )
    )
  )
  (speller): Speller(
    (rnn): LSTM(512, 512, num_layers=2, batch_first=True, dropout=0.3)
    (emb): Embedding(323, 512)
    (attention): MultiHeadAttention(
      (scaled_dot): ScaledDotProductAttention()
      (query_projection): Linear(in_features=512, out_features=512, bias=True)
      (value_projection): Linear(in_features=512, out_features=512, bias=True)
      (out_projection): Linear(in_features=1024, out_features=512, bias=True)
    )
    (mlp): Sequential(
      (0): Linear(in_features=1024, out_features=512, bias=True)
      (1): Tanh()
      (2): Linear(in_features=512, out_features=323, bias=True)
    )
    (softmax): LogSoftmax(dim=-1)
  )
)
-----------------------------------
```

- Pre-trained LAS 모델로 학습하는 단계
- 이미 kspon 데이터로 학습된 모델로 우수한 성능을 보임을 확인
- LAS 모델 구조: Listener(Acoustic Model) – Speller(Language Model)

2. KoSpeech 공부

(5) Inference

```
#!python inference.py
!python inference.py --audio_path "audio file 경로"
```

```
=========================================
=========================================
[2021-03-18 00:31:02] - Inference Start
=========================================

아 뭔 소리야? 그것도
=========================================
[2021-03-18 00:31:03] - Inference complete
Elapsed time: 0:00:00.845489
=========================================
```

```
RuntimeError: Error(s) in loading state_dict for ListenAttendSpell:
        size mismatch for speller.emb.weight: copying a param with shape torch.Size([2001, 512]) from checkpoint, the shape in current model is torch.Size([323, 512]).
        size mismatch for speller.mlp.2.weight: copying a param with shape torch.Size([2001, 512]) from checkpoint, the shape in current model is torch.Size([323, 512]).
        size mismatch for speller.mlp.2.bias: copying a param with shape torch.Size([2001]) from checkpoint, the shape in current model is torch.Size([323]).
```

- 학습된 모델로 실제 음성을 가지고 텍스트를 추론하는 단계
- 이때 size mismatch error가 발생하는데, 기존 모델의 size와 임의로 줄여서 넣은 데이터로 구축한 모델 size가 맞지 않기 때문
- 이는 train으로 생성된 새로운 pretrained.pt를 사용함으로써 해결하기로 함

3. Train / Inference

3. Train / Inference

(1) Json -> Text

(2) Tunning KoSpeech

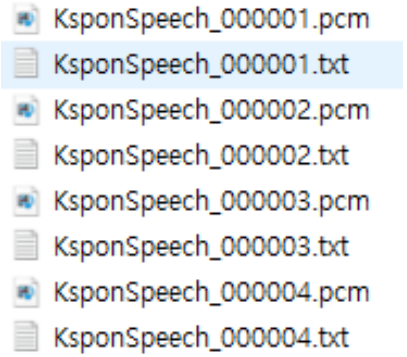
(3) Test / Inference

(1) Json To Txt

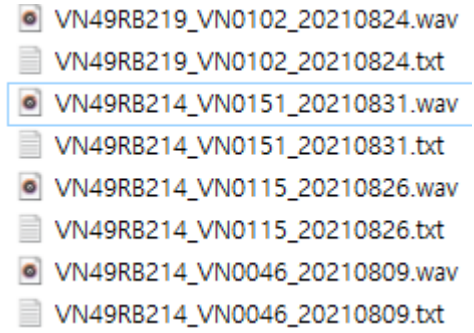

```
{
    "fileName": "CN11RC001_CN0001_20210827.wav",
    "file_info": {
        "speakerID": "CN0001",
        "sentenceID": "CN11RC001",
        "recordUnit": "ios",
        "recordQuality": "16bit 16kHz MONO",
        "recordDate": "2021-08-27 17:53:07",
        "recordTime": "8.416"
    },
    "transcription": {
        "Reading": "잠시만. 내 지갑이 어디 갔지? 유민아, 미안한데 내 지갑이 없어진 거 같아.",
        "ReadingLabelText": "잠시만 내 지갑이 어디 갔지 유민아 미안한데 내 지갑이 없어진 거 같아",
        "Question": "",
        "AnswerLabelText": "",
        "SentenceSpeechLV": "중"
    },
    "SpeakerID": "CN0001",
    "basic_info": {
        "gender": "F",
        "birthYear": "2000",
        "eduBackground": "고졸"
    },
    "residence_info": {
        "country": "CN",
        "residencePeriod": "1년 이상 3년 미만",
        "residenceCity": "KR-11"
    },
    "skill_info": {
        "languageClass": "중국어",
        "motherTongue": "중국어",
        "selfAssessment": "상",
        "topikGrade": "5",
        "LearningPeriod": "48",
        "learningSource": "학원"
    }
}
```


제공받은 *.JSON 파일을 KoSpeech에
Training 시키기 위하여, *.txt 형식으로 바꾼 후 (encoding = 'ANSI')
DataSet 구성

KoSpeech



Our TrainDataSet



예시)

아/ 몬 소리야, 그건 또. b/

KoSpeech 에선 Noise를 '/ b/ **등등**' 을
이용하여 표현하였는데, 제공되어지는

JSON **에서 Noise에 존재 유무를**
확인할 수 없었음.

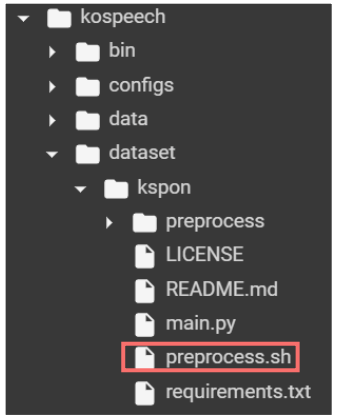
잠시만 내 지갑이 어디 갔지 유민아

미안한데 내 지갑이 없어진 거 같아

30000개의 Train DataSet 구성

(2) Tunning KoSpeech

Preprocess



2가지 변수 수정

```
DATASET_PATH="SET_YOUR_DATASET_PATH"
VOCAB_DEST='SET_LABELS_DESTINATION'
OUTPUT_UNIT='character'
PREPROCESS_MODE='phonetic'
VOCAB_SIZE=5000

echo "Pre-process KsponSpeech Dataset.."

python main.py ₩
--dataset_path $DATASET_PATH ₩
--vocab_dest $VOCAB_DEST ₩
--output_unit $OUTPUT_UNIT ₩
--preprocess_mode $PREPROCESS_MODE ₩
--vocab_size $VOCAB_SIZE ₩
```

Our Dataset_voca(label).csv

id	char	freq
0	<pad>	0
1	<sos>	0
2	<eos>	0
3		417973
4	이	39321
5	요	37635
6	에	29201
7	는	28039
8	가	27545
9	어	27106

1. 사전학습을 통해,
DataSet의 Vocab(Label)을 만든다.

2. PreProcess 후 생긴
DataSet의 Vocab(Label)

(2) Tunning KoSpeech

Train

4가지 변수 수정

```
#train dataset
root : C:/SpeechRecognitionDataset/Dataset/AI_hub
script_data_path: C:/Users/sanma/PycharmProjects/las.pyt
vocab_path: C:/Users/sanma/PycharmProjects/las.pytorch/d

#train
batch_size: 8
epochs: 5
use_cuda: True
cer_every: 500
teacher_forcing_step: 0.005
teacher_forcing_ratio: 0.99
min_teacher_forcing_ratio: 0.7
label_smoothing: 0.1

#model_save_load
resume: False
save_every: 1
#inference
inference: False
use_val_data: True
weight_path: C:/Users/sanma/PycharmProjects/las.pytorch/
```

root : Dataset root 디렉토리
script_data_path : script 파일
vocab_path : preprocess로 생성된 vocab 파일(label)
weight_path : #pre-trained model의 가중 파일

Epoch 5

```
[2021-12-09 17:54:58] - Train : epoch 4 loss:76.78 train_cer:0.09 val_cer:0.00

[2021-12-09 17:54:58] - Train Complete
Elapsed time: 2:11:25.365222
```

Epoch 20

```
100%|
[2021-12-10 10:00:22] - validation complete

[2021-12-10 10:00:22] - Train : epoch 19 loss:89.71

[2021-12-10 10:00:22] - Train Complete
Elapsed time: 11:08:08.926872
```

최종 Epoch 50으로 선정

pretrained.pt

(3) Test / Inference

Inference

```
import os
from glob import glob
import pandas as pd
os.chdir('/content/drive/MyDrive/Project/sound/NIA_contest/icanhearyourvoice/UnzipTest')   # test파일 경로 넣기
audio_lists = glob('*.wav')    # .wav로 하기
path = '/content/drive/MyDrive/Project/sound/NIA_contest/icanhearyourvoice/UnzipTest/'   # test 파일 경로 지정

result = []

for i in audio_lists :
  #!python inference.py
  %cd /content/drive/MyDrive/Project/sound/NIA_contest/las.pytorch
  filename = path + i
  a = !python inference.py --audio_path {filename}
  result.append(a[45])

final = pd.read_csv('/content/drive/MyDrive/Project/sound/NIA_contest/icanhearyourvoice/submission.csv')

for i in range(len(result)) :
  index = final[final['fileName']==audio_lists[i]].index.values[0]
  final.iloc[index, 1] = result[i]
```

결과

```
=======================================
[2021-12-12 02:22:34] - Inference Start
=======================================

그럼
=======================================
[2021-12-12 02:22:34] - Inference complete
Elapsed time: 0:00:00.556308
=======================================
```

?

모든 TEST에 '그럼' 만
나오는 결과를 얻었다.

(4) Result

Problem

1. RunTime 및 구글드라이브에 잦은 오류로 인해, 시간이 부족했고 파일 업로드에서도 지속적인 오류가 발생하여 충분한 try를 해보지 못함

2. 데이터를 고려한 충분한 모델 튜닝 없이 바로 학습을 진행해서 발생하는 문제인가?

3. 30만개의 DATASET을 다 처리하지 못하고, 3만개만 써서 발생하는 문제인가?

4. KoSpeech의 형식에 맞게 전처리를 했다고 생각했지만, 그 과정에서 발생하는 오류인가?

5. RunTime의 제한으로 인한 EPOCH 수를 늘리지 못해서 발생하는 오류인가?

6. 50번의 EPOCH로 얻은 Pretrained.pt 의 값이 형편없어서 발생하는 오류인가?

4. 서비스 모델 제안

4. 서비스 모델 제안

(1) 기존 음성 인식 모델

- 기존 음성 인식 시스템의 대부분은 성인 남녀 음성 기준
- 효과적인 외국인 발화 음성 인식을 위한 제안: 새로운 End-to-End 방식의 모델 적용
- End-to-End: 입력에서 출력까지 파이프라인 네트워크 없이 신경망으로 한 번에 처리하는 방식.
- 즉, 모델의 모든 매개변수가 하나의 손실함수에 대해 동시에 훈련되는 경로가 가능한 네트워크로 신경망의 입력 및 출력을 직접 고려하여 네트워크 가중치를 최적화 할 수 있음
- 이를 통해 제한된 도메인(성인 남녀 음성)에서 학습하더라도 새로운 도메인(외국인 음성)에서도 잘 적용될 수 있는 유연한 모델 구축 가능

4. 서비스 모델 제안

(2) 제안된 음성 인식 모델

- 기존 End-to-End 모델: LAS, ESPnet 등
- 제안 End-to-End 음성인식 모델: SEQFORMER 구조의 end-to-end 방식
- SEQFORMER: 재귀적 뉴럴 네트워크 기반의 인코더 + Transformer 디코더
- Encoder: CNN-LSTM 형태의 인코더 + 음향적 데이터 증강 기법(ADA) -> 효과적인 음향모델 학습
- Decoder: Transformer 디코더 + masked 언어정보 예측(MLP) -> 언어 정보 표현을 위한 모델 강화
- 음향적 다양성과 언어 표현의 다양성을 학습하여 강인한 음성인식 엔진을 구현할 것으로 기대
- 참고 논문: https://www.koreascience.or.kr/article/JAKO202009135472564.page

4. 서비스 모델 제안

(3) SEQFORMER 구조

1. CNN-LSTM 형태의 인코더

음향적 잠재 변별 정보 학습

2. 음향적 데이터 증강 기법(ADA)

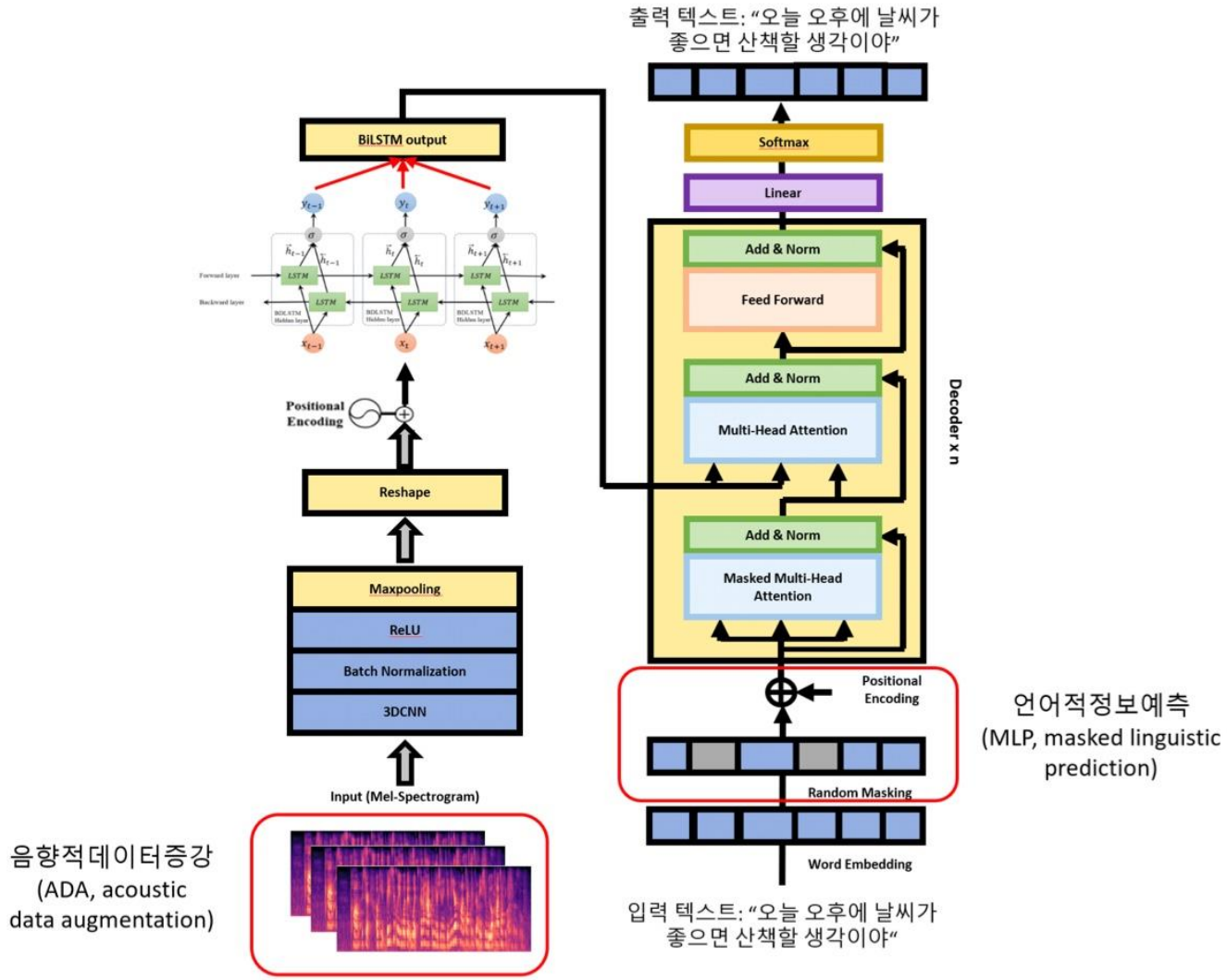
다양한 음향적 특성 표현

3. Transformer 디코더

음성 입력에 대한 출력 문자열의 예측 학습

4. masked 언어정보 예측(MLP)

다양한 언어정보 표현을 위한 모델 강화



Thank You