

# Nvidia Nemo를 활용한 한국어 End-to-end ASR fine-tuning

서강대학교 컴퓨터공학과 김지환

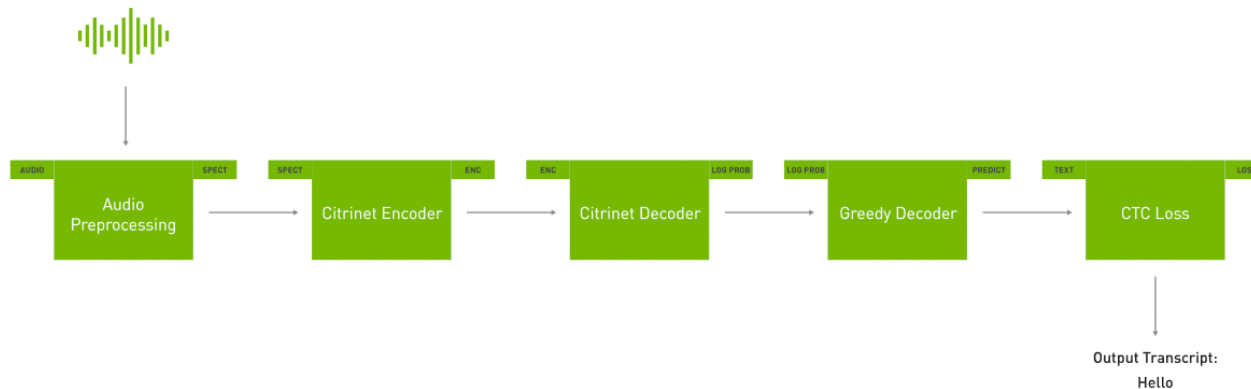
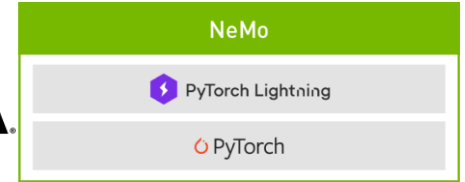
# Table of contents

- 1. Nemo 소개
- 2. Google colab 소개
- 3. Nemo 설치 및 환경 세팅 (Colab 기준)
- 4. Nemo ASR 실습
- 5. 결론

# Nemo 소개

## ■ NeMo (NVIDIA, 2019)

- <https://developer.nvidia.com/nvidia-nemo> (homepage)
- <https://github.com/NVIDIA/NeMo> (source)
- NVIDIA NeMo™ is an open-source framework for developers to build and train state-of-the-art (SOTA) conversational AI models.
- PyTorch, PyTorch Lightning을 기반으로 작성된 E2E toolkit
- SOTA model들의 pretrained model을 제공
  - ASR pretrained models : <https://catalog.ngc.nvidia.com/>



# Google colab 소개



## ■ Google Colaboratory (Google)

- <https://colab.research.google.com/>
- 웹 브라우저에서 파이썬을 작성하고 실행할 수 있는 서비스
- 클라우드 기반의 주피터 노트북 개발환경
- 기본적으로 파이썬을 사용가능
  - Tensorflow, PyTorch, matplotlib, scikit-learn, pandas 등의 ML/DL에 사용하는 라이브러리들을 기본적으로 지원
- K80 GPU를 무료로 사용 가능
  - 사용량의 제한이 있으나 일반적으로 교육용으로 사용하기에는 문제 없음

	Colab Free	Colab Pro	Colab Pro +
Guarantee of resources	Low	High	Even Higher
GPU	K80	K80, T4 and P100	K80, T4 and P100
RAM	16 GB	32 GB	52 GB
Runtime	12 hours	24 hours	24 hours
Background execution	No	No	Yes
Costs	Free	9.99\$ per month	49.99\$ per month
Target group	Casual user	Regular user	Heavy user

# Nemo 설치 및 환경 세팅 (Colab 기준)

## ■ Nemo 설치

- PIP를 이용한 설치 (colab 권장)
  - `$pip install nemo_toolkit['all']`
- Source code를 이용한 설치
  - `$apt-get update && apt-get install -y libsndfile1 ffmpeg`
  - `$git clone https://github.com/NVIDIA/NeMo`
  - `$cd NeMo`
  - `$/reinstall.sh`

# Nemo 설치 및 환경 세팅 (Colab 기준)

## ■ Nemo 설치 (실습)

- Pip 명령어를 통해 nemo 설치
- 주요 library import
  - omegaconf: yaml, json 등의 configuration 파일을 읽고 쓸 수 있는 lib. Nemo에서 기본적으로 사용함
  - nemo.collections.asr: Nemo ASR class
  - nemo.utils.exp\_manager: 학습 로그, conf 등에 사용되는 lib
  - datasets.load\_dataset: 학습 및 테스트 데이터 관리 lib로, huggingface에서 사용됨

## Prerequisites

```
[1] !pip install nemo_toolkit['all']
```

숨겨진 출력 표시

```
[2] import copy  
from omegaconf import OmegaConf, open_dict
```

```
[3] import nemo  
import nemo.collections.asr as nemo_asr  
from nemo.utils import exp_manager
```

[NeMo W 2022-07-07 05:02:01 optimizers:55] Ape:

```
!pip install datasets  
from datasets import load_dataset
```

# Nemo ASR 실습

## ■ 실습 순서

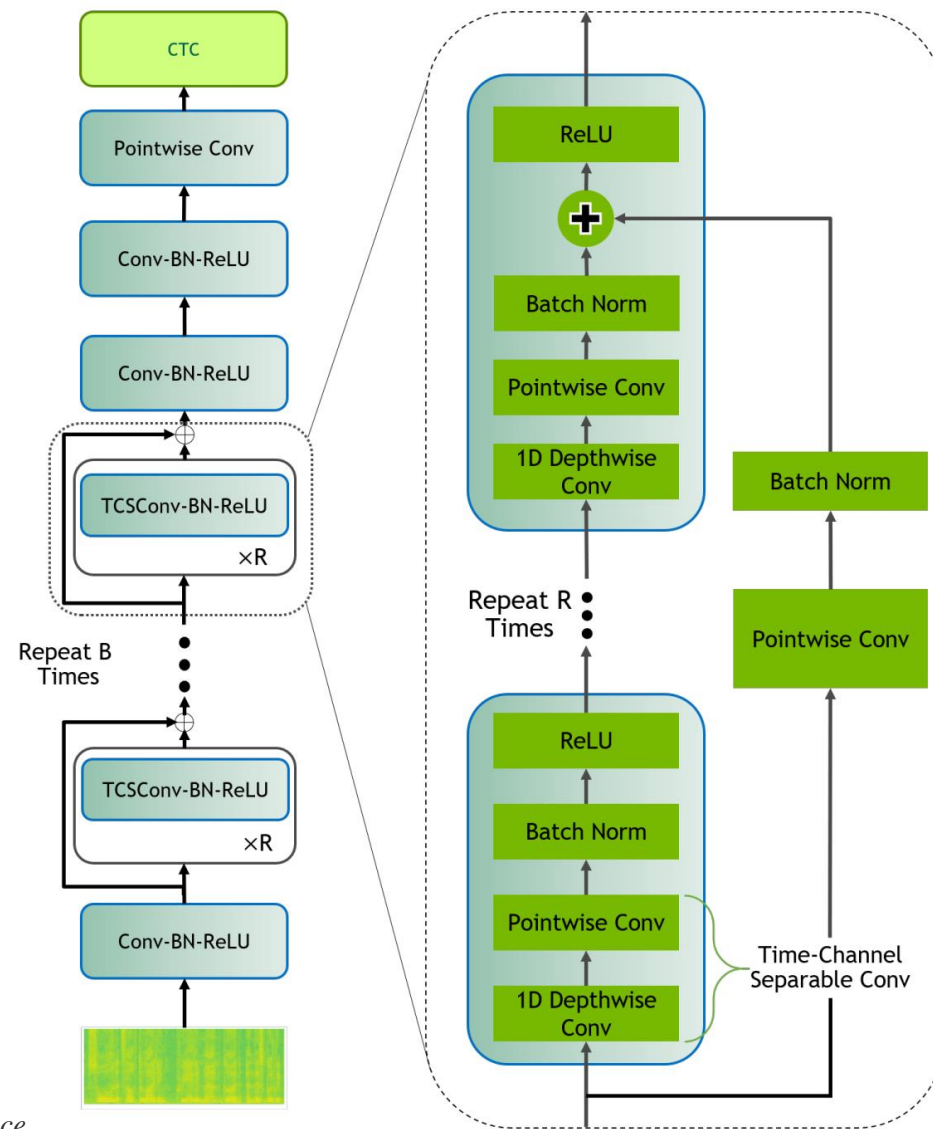
- 1) Pre-trained 모델 불러오기 (영어)
- 2) Small data를 통한 모델 확인 (영어)
- 3) Train/test 데이터 불러오기 및 데이터 확인 (한국어)
- 4) 한국어 fine-tuning을 위한 모델 설정
- 5) 한국어 output unit 설정 및 training 세팅
- 6) 학습
- 7) 테스트

# Nemo ASR 실습

## ■ 1) Pre-trained 모델 불러오기 (영어)

- Nemo pretrained catalog에서 모델을 받아 사용 가능
  - <https://catalog.ngc.nvidia.com/models>
- 본 실습에서는 Quartznet을 사용함 (실습용)
  - 1-dimensional, pointwise convolution layer와 batch normalization, ReLU layer들로 구성된 R개의 block을 B개 쌓아서 만들어진 모델
  - SOTA성능을 보이는 모델은 아니지만, 파라미터 수가 작아 low-resourced computing 환경에서 동작이 용이
  - 500MB~1,000MB 사이의 SOTA 수준 모델에 비해 78MB의 적은 용량으로 동작 가능
  - Librispeech test set 기준 4.19%의 word error rate (WER)을 보임

S. Krnan *et al.*, "Quartznet: Deep Automatic Speech Recognition with 1D Time-Channel Separable Convolutions," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6124-6128, 2020





# Nemo ASR 실습

## ■ 1) Pre-trained 모델 불러오기 (영어)

- 모델 불러오기

Pre-trained model

```
[5] char_model = nemo_asr.models.ASRModel.from_pretrained("stt_en_quartznet15x5", map_location='cpu')
```

- 다른 추천 ASR 모델

- Conformer(SOTA), Citrinet, Contextnet

# Nemo ASR 실습

- 2) Small data를 통한 모델 확인 (영어)
  - Huggingface의 librispeech test corpus를 불러와 사용
    - 모델 동작 확인용
  - 임의의 sample에 대해 재생 및 인식 과정 수행
  - 모델에 대한 음성인식
    - {model}.transcribe({listOfFilepath})를 통해 결과 확인 가능

## Pre-trained model

```
[5] char_model = nemo_asr.models.ASRModel.from_pretrained("stt_en_quartznet15x5", map_location='cpu')
```

숨겨진 출력 표시

```
[6] ds = load_dataset("kresnik/librispeech_asr_test", "clean")
```

숨겨진 출력 표시

```
[7] test_ds = ds['test']  
sample = test_ds[0]  
sample
```

숨겨진 출력 표시

```
[8] import IPython
```

```
IPython.display.Audio(sample['file'])
```

▶ 0:04 / 0:04 ———— 🔊 ⋮

```
[9] result = char_model.transcribe([sample['file']])  
results = char_model.transcribe(test_ds['file'][:10])
```

Transcribing: 100%  1/1 [00:12<00:00, 12.23s/it]

Transcribing: 100%  3/3 [00:11<00:00, 3.22s/it]

```
▶ print("Hypothesis: " + result[0])  
print("Reference: " + sample['text'].lower())
```

🔗 Hypothesis: it is sixteen years since john bergson died  
Reference: it is sixteen years since john bergson died

# Nemo ASR 실습

Korean datasets

label set (character set)

- 3) training/test 데이터 불러오기 및 데이터 확인 (한국어)
  - Huggingface의 zeroth\_korean corpus를 불러와 사용
    - Zeroth\_korean: 약 50시간 분량의 휴대폰으로 녹음된 한국어 corpus
      - \* 출처: <http://openslr.org/40/>
  - Training set과 test set을 분리하여 저장함
    - train\_ds: training set (file, audio, text, speaker 정보)
    - test\_ds: test set (file, audio, text, speaker 정보)

```
[12] ds = load_dataset("kresnik/zeroth_korean", "clean")
```

Downloading builder script: 100%

Downloading and preparing dataset zeroth\_korean/clean

Downloading data: 100%

Dataset zeroth\_korean downloaded and prepared to /root

100% 2/2 [00:0

```
[13] train_ds = ds['train']  
test_ds = ds['test']
```

```
test_ds[0]
```

```
{'audio': {'array': array([-3.0517578e-05, -6.1035156e-05, -3.0517578e-05, ...,  
-1.2207031e-04, 1.8310547e-04, -1.2207031e-04], dtype=float32),  
'path': '/root/.cache/huggingface/datasets/downloads/extracted/3c93119fdbcba519e1529c416a7339ed19b67c18686fc5bea5eda3309e01e58e/test_data_01/003/137/137_003_0008.flac',  
'sampling_rate': 16000},  
'chapter_id': 3,  
'file': '/root/.cache/huggingface/datasets/downloads/extracted/3c93119fdbcba519e1529c416a7339ed19b67c18686fc5bea5eda3309e01e58e/test_data_01/003/137/137_003_0008.flac',  
'id': '137_003_0008',  
'speaker_id': 137,  
'text': '보유중인 부동산 자산을 추가로 매각해 총 일 조 사천 억원의 현금을 확보해 재무구조를 개선한다는 방침이다'}
```

# Nemo ASR 실습

## ■ 3) training/test 데이터 불러오기 및 데이터 확인 (한국어)

- Nemo의 ASR data preparation은 3가지의 정보를 필요로 함
  - audio\_filepath, duration, transcription
- 불필요한 화자 정보 등을 제거
- 각 음성 sample 당 duration을 계산하기 위한 함수 정의

```
train_ds = train_ds.remove_columns(["speaker_id", "chapter_id", "id", "audio"])  
test_ds = test_ds.remove_columns(["speaker_id", "chapter_id", "id", "audio"])
```

```
import soundfile as sf  
def get_duration(batch):  
    speech = sf.SoundFile(batch['file'])  
    duration = speech.frames / speech.samplerate  
    batch['duration'] = duration  
    return batch
```

```
train_ds = train_ds.map(get_duration)  
test_ds = test_ds.map(get_duration)
```

# Nemo ASR 실습

## ■ 3) Training/test 데이터 불러오기 및 데이터 확인

- 양식에 맞게 기존 정보를 재조정
- 저장된 prepared\_data 정보를 json 형식으로 변환
- 변환된 json 파일을 불러오기


```
train_ds = train_ds.rename_column(original_column_name='file', new_column_name='audio_filepath')  
test_ds = test_ds.rename_column(original_column_name='file', new_column_name='audio_filepath')
```

te manifest

```
import os  
  
train_json_path = os.path.abspath('train.json')  
test_json_path = os.path.abspath('test.json')  
  
train_json = train_ds.to_json(train_json_path)  
test_json = test_ds.to_json(test_json_path)
```

Creating json from Arrow format: 100%  3/3 [00:00<00:00, 1.78ba/s]

Creating json from Arrow format: 100%  1/1 [00:00<00:00, 27.13ba/s]

Reading manifest data:  22263/? [00:00<00:00, 74981.26it/s]

Reading manifest data:  457/? [00:00<00:00, 7310.05it/s]

```
import json  
from tqdm.auto import tqdm  
  
def read_manifest(path):  
    manifest = []  
    with open(path, 'r') as f:  
        for line in tqdm(f, desc="Reading manifest data"):  
            line = line.replace("#n", "")  
            data = json.loads(line)  
            manifest.append(data)  
    return manifest  
  
train_manifest = read_manifest('train.json')  
test_manifest = read_manifest('test.json')
```

# Nemo ASR 실습

## ■ 4) 한국어 fine-tuning을 위한 모델 설정

- 영어로 학습된 모델을 소용량의 한국어를 이용하여 원활히 tuning하기 위해
  - model의 encoder 정보 (speech representation) 는 그대로 유지한 채
  - Model의 decoder 정보 (sequence of output unit representation) 을 재학습하는 것이 효율적
- 충분한 양의 데이터를 확보하지 못했거나, computing환경이 부족할 때 사용하는 방법임
- 하지만, encoder 전체를 학습하지 않는 경우 normalization 문제가 발생할 수 있음
  - Ex> 원본 모델이 학습한 음성 데이터와 새로운 학습 데이터의 볼륨(소리 크기) 차이가 많이 나는 문제
  - 이를 방지하기 위해 batch normalization 부분은 freeze하지 않음

# Nemo ASR 실습

## ■ 4) 한국어 fine-tuning을 위한 모델 설정

- Encoder freeze

- freeze() method를 통해 설정
- Batch normalization 부분은 unfreeze로 설정하기 위해, method를 정의하여 반영함

```
import torch
import torch.nn as nn

def enable_bn(m):
    if type(m) == nn.BatchNorm1d:
        m.train()
        for param in m.parameters():
            param.requires_grad_(True)
```

```
char_model.encoder.freeze()
char_model.encoder.apply(enable_bn)
```

# Nemo ASR 실습

## ■ 5) 한국어 output unit 설정 및 training 세팅

- 영어 알파벳으로 정의된 모델을 한국어 음절로 변경하는 작업

- `print(OmegaConf.to_yaml(char_model.cfg))` 로 확인 가능

- 한국어 음절 추출

- Training/test dataset의 transcription을 추출하여 모델의 output unit으로 정의

```
def extract_all_chars(batch):  
    all_text = " ".join(batch["text"])  
    vocab = list(set(all_text))  
    return {"vocab": [vocab], "all_text": [all_text]}
```

```
vocab_train = train_ds.map(extract_all_chars, batched=True, batch_size=-1, keep_in_memory=True, remove_columns=train_ds.column_names)  
vocab_test = test_ds.map(extract_all_chars, batched=True, batch_size=-1, keep_in_memory=True, remove_columns=test_ds.column_names)
```

```
vocab_list = list(set(vocab_train["vocab"][0]) | set(vocab_test["vocab"][0]))
```

```
vocab_list
```

- Change\_vocabulary method를 통해 모델의 output unit을 변경함

- \* `char_model.change_vocabulary(new_vocabulary=vocab_list)`



# Nemo ASR 실습

## ■ 5) 한국어 output unit 설정 및 training 세팅

### ● Fine-tuning training을 위한 configuration 세팅

- 생성한 단어 사전을 모델에 적용
- 기존 모델의 configuration 정보를 복사
- 데이터와 학습 자료에 대한 내용을 update
  - \* 학습 자료의 위치, batch size, vocab list 등
  - \* 한국어는 대소문자가 없으므로 normalize\_transcripts는 false

```
char_model.cfg.labels = vocab_list
```

```
cfg = copy.deepcopy(char_model.cfg)
```

```
# Setup train, validation, test configs
```

```
with open_dict(cfg):
```

```
    # Train dataset (Concatenate train manifest cleaned and dev manifest cleaned)
```

```
    cfg.train_ds.manifest_filepath = f"{train_json_path}"
```

```
    cfg.train_ds.labels = vocab_list
```

```
    cfg.train_ds.normalize_transcripts = False
```

```
    cfg.train_ds.batch_size = 16
```

```
    cfg.train_ds.num_workers = 2
```

```
    cfg.train_ds.pin_memory = True
```

```
    cfg.train_ds.trim_silence = True
```

```
# Validation dataset (Use test dataset as validation, since we train using train + dev)
```

```
cfg.validation_ds.manifest_filepath = test_json_path
```

```
cfg.validation_ds.labels = vocab_list
```

```
cfg.validation_ds.normalize_transcripts = False
```

```
cfg.validation_ds.batch_size = 8
```

```
cfg.validation_ds.num_workers = 8
```

```
cfg.validation_ds.pin_memory = True
```

```
cfg.validation_ds.trim_silence = True
```

```
# setup data loaders with new configs
```

```
char_model.setup_training_data(cfg.train_ds)
```

```
char_model.setup_multiple_validation_data(cfg.validation_ds)
```

# Nemo ASR 실습

## ■ 5) 한국어 output unit 설정 및 training 세팅

- Optimizer & augmentation 설정

- Quartznet fine-tuning 논문을 참조하여 parameter 세팅함
- Spec\_augment는 기존의 것을 그대로 사용함

```
# Original optimizer + scheduler
print(OmegaConf.to_yaml(char_model.cfg.optim))
```

```
name: novograd
lr: 0.01
betas:
- 0.8
- 0.5
weight_decay: 0.001
sched:
  name: CosineAnnealing
  warmup_steps: null
  warmup_ratio: null
  min_lr: 0.0
  last_epoch: -1
```

```
with open_dict(char_model.cfg.optim):
    char_model.cfg.optim.lr = 0.01
    char_model.cfg.optim.betas = [0.95, 0.5] # from paper
    char_model.cfg.optim.weight_decay = 0.001 # Original weight decay
    char_model.cfg.optim.sched.warmup_steps = None # Remove default number of steps of warmup
    char_model.cfg.optim.sched.warmup_ratio = 0.05 # 5 % warmup
    char_model.cfg.optim.sched.min_lr = 1e-5
```

```
print(OmegaConf.to_yaml(char_model.cfg.spec_augment))
```

```
_target_: nemo.collections.asr.modules.SpectrogramAugmentation
rect_freq: 50
rect_masks: 5
rect_time: 120
```

```
char_model.spec_augmentation = char_model.from_config_dict(char_model.cfg.spec_augment)
```

# Nemo ASR 실습

## ■ 6) 학습

- 학습 과정을 표시하는 metric 설정
  - Character error rate (CER)을 사용
    - \* 정답과 비교하여 음절 단위의 오류율을 측정
  - Pytorch\_lightning을 이용하여 trainer 생성
    - \* 모델 학습 & 테스트를 단순화하는 PyTorch library.

```
char_model._wer.use_cer = True  
char_model._wer.log_prediction = True
```

```
import torch  
import pytorch_lightning as pl  
  
if torch.cuda.is_available():  
    gpus = 1  
else:  
    gpus = 0  
  
EPOCHS = 10  
  
trainer = pl.Trainer(gpus=gpus,  
                    max_epochs=EPOCHS,  
                    accumulate_grad_batches=1,  
                    checkpoint_callback=False,  
                    logger=False,  
                    log_every_n_steps=50,  
                    check_val_every_n_epoch=10)  
  
# Setup model with the trainer  
char_model.set_trainer(trainer)  
  
# Finally, update the model's internal config  
char_model.cfg = char_model._cfg
```

# Nemo ASR 실습

## ■ 6) 학습

- 학습 결과와 로그를 저장할 수 있는 경로 설정
- 학습 진행 상황 확인을 위한 TensorBoard 설정

```
try:
    from google import colab
    COLAB_ENV = True
except (ImportError, ModuleNotFoundError):
    COLAB_ENV = False

# Load the TensorBoard notebook extension
if COLAB_ENV:
    %load_ext tensorboard
    %tensorboard --logdir /content/experiments/lang/ASR-Char-Model-Korean/
else:
    print("To use tensorboard, please use this notebook in a Google Colab environment.")
```

```
os.environ.pop('NEMO_EXPM_VERSION', None)

config = exp_manager.ExpManagerConfig(
    exp_dir=f'experiments/lang/',
    name=f"ASR-Char-Model-Korean",
    checkpoint_callback_params=exp_manager.CallbackParams(
        monitor="val_wer",
        mode="min",
        always_save_nemo=True,
        save_best_model=True,
    ),
)

config = OmegaConf.structured(config)

logdir = exp_manager.exp_manager(trainer, config)
```

# Nemo ASR 실습

## ■ 6) 학습

- fit() method를 통해 학습

```
%%time
trainer.fit(char_model)
```

```
... LOCAL_RANK: 0 - CUDA_VISIBLE_DEVICES: [0]
[NeMo | 2022-07-07 06:23:30 modelPT:579] Optimizer config = Novograd (
  Parameter Group 0
    amsgrad: False
    betas: [0.95, 0.5]
    eps: 1e-08
    grad_averaging: False
    lr: 0.01
    weight_decay: 0.001
  )
[NeMo | 2022-07-07 06:23:30 lr_scheduler:837] Scheduler "<nemo.core.optim.lr_scheduler.CosineAnnealing object at 0x7fc478db06d0>"
will be used during training (effective maximum steps = 13870) -
Parameters :
(warmup_steps: null
warmup_ratio: 0.05
min_lr: 1.0e-05
last_epoch: -1
max_steps: 13870
)
```

	Name	Type	Params
0	preprocessor	AudioToMelSpectrogramPreprocessor	0
1	encoder	ConvASREncoder	18.9 M
2	spec_augmentation	SpectrogramAugmentation	0
3	_wer	WER	0
4	decoder	ConvASRDecoder	1.2 M
5	loss	CTCLoss	0

```
-----
1.3 M   Trainable params
18.8 M   Non-trainable params
20.1 M   Total params
80.515   Total estimated model params size (MB)
[NeMo W 2022-07-07 06:23:31 nemo_logging:349] /usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py:490: UserWarning: This DataLoader will create 8 wo
cpuset_checked))

[NeMo | 2022-07-07 06:23:32 wer:253]

[NeMo | 2022-07-07 06:23:32 wer:254] reference:보유중인 부동산 자산을 추가로 매각해 총 일 조 사천 억원의 현금을 확보해 재무구조를 개선한다는 방침이다
[NeMo | 2022-07-07 06:23:32 wer:255] predicted: 이 이 이 이 이 이 컷 컷 있니다
[NeMo | 2022-07-07 06:23:32 wer:253]

[NeMo | 2022-07-07 06:23:32 wer:254] reference:장발적 손해배상제가 피해액 세 배까지 배상액을 제한한 것을 세 배 이상으로 강화하겠다고 했다
[NeMo | 2022-07-07 06:23:32 wer:255] predicted: 이 이 이 컷 있니다

Epoch 5: 63%
```

# Nemo ASR 실습

■ 6) 학습

- 학습 초반 상황

Epoch 19: 80%

[NeMo I 2022-07-08 15:13:38 wer:253]

[NeMo I 2022-07-08 15:13:38 wer:254] reference: 그는 이런 사태를 예견하고 일 년 전부터 하베르 코레라는 일 인 뉴스 사이트를 열어 언론 활동을 계속하고 있다

[NeMo I 2022-07-08 15:13:38 wer:255] predicted: 총발바신람꼭뵈븐속엮얏복해풍경실써개신해현통개객프부영채넉냈들벗프꼭중프독배총곡런쓰개건결의꾼헐췌태스개꼭헐불갠음벧췌큰뚝둑크닐릉헛꼭당뵈온꺾극결해분담분쪽팔삿개관권험름뵈얏쪽온얏전얏건전통얏지췌탈부찌릉말쪽뉴렛복배름전걸너익췌험두북통멧흔영쪽통멧은넷쪽쪽통멧흙은솔럼섯문뒤단쪽전개진알싯단개갈프론헤뒤것푸롬움달개업력뒤프쪽택덜툼채물건결올집꼭새채류에택람심줄무극실단쪽별택췌채프푸스프꼭쓰겔흔쪽올집개크쪽의택뭇합뽀단저첸세헌전헐채양극통울퓰헤영트앤시스몽꼭꾸근무겔멧하삼진영채개업즐거긴통달헐헐개

[NeMo I 2022-07-08 15:14:03 wer:253]

[NeMo I 2022-07-08 15:14:03 wer:254] reference: 방잠수선 인근해역에는 서해해양경비안전본부 해양환경관리공단의 방제선 여덟 척과 상하이 셀비지 작업선 아홉 척이 유실 등을 막기 위해 겹겹으로 에워싸고 있다

[NeMo I 2022-07-08 15:14:03 wer:255] predicted: 단출꼭풍운택된쓴

[NeMo I 2022-07-08 15:14:27 wer:253]

[NeMo I 2022-07-08 15:14:27 wer:254] reference: 그마저도 남편이 다른 회사로 옮기면서 차가 필요해 중고차를 구매하자 지원대상에서 제외됐다

[NeMo I 2022-07-08 15:14:27 wer:255] predicted:

[NeMo I 2022-07-08 15:14:52 wer:253]

# Nemo ASR 실습

## ■ 6) 학습

### ● Epoch 10

Epoch 10, global step 15256: val\_wer was not in top 3

[NeMo I 2022-07-08 17:20:46 wer:253]

[NeMo I 2022-07-08 17:20:46 wer:254] reference:강아지가 용변을 보도록 유도하는 말을 가르칠 수도 있다

[NeMo I 2022-07-08 17:20:46 wer:255] predicted:강화지가 용변대보도록 이유도 하는 말을 가르치수도 있다

[NeMo I 2022-07-08 17:21:11 wer:253]

[NeMo I 2022-07-08 17:21:11 wer:254] reference:교회에 가까운 노구를 기어코 평행봉에 거꾸로 세우는 근성과 체력이야말로 질병 등으로 고통스런 삶을 살아온 그의 근 삼십 년 야인 생활을 지탱해준 힘일 테다

[NeMo I 2022-07-08 17:21:11 wer:255] predicted:고이에 가까운 노구를 기육거 평인부의 거꾸로세우는 근선과 치력이야 말러 출병등으로 보통스운 설을 사라운 구 근 삼십 년 년인상을를 지다가된 마해다

[NeMo I 2022-07-08 17:21:35 wer:253]

[NeMo I 2022-07-08 17:21:35 wer:254] reference:그러나 한편으로는 젊은 세대 성악도들을 향해 조금은 성급한 것 같다

[NeMo I 2022-07-08 17:21:35 wer:255] predicted:그러나 한평으로는 젊문세대 성학도들을 명해 조금의 상각한것됐다

[NeMo I 2022-07-08 17:22:00 wer:253]

[NeMo I 2022-07-08 17:22:00 wer:254] reference:기동민 민주당 원내대변인은 이날 논평을 통해 테니스 치고 사 대 강변에서 자전거 타던 이 전 대통령께서 낮 뜨거움 운운하며 입을 열었다며 이같이 비난했다

[NeMo I 2022-07-08 17:22:00 wer:255] predicted:기동민 민주한 원내 대변이는 이날 농병을 통해 테래서 제고 사대감면에서 자정거 다한 이이 전대 통령 께서 나뜨 거음 우는 나며 입을 열었다며 이갈치 비난했다

[NeMo I 2022-07-08 17:22:24 wer:253]

# Nemo ASR 실습

## ■ 6) 학습

- Epoch 18

```
Epoch 18, global step 26352: val_wer reached 0.28204 (best 0.28204), saving model to "/data1/hosung/works/tutorials/experiments/lang/ASR-Char-Model-Korean/2022-07-08"
[NeMo I 2022-07-08 18:53:07 wer:253]

[NeMo I 2022-07-08 18:53:07 wer:254] reference:조폭의 자금으로 굴러가는 지하경제 규모만 백 이십 일 조원에 이르는 것으로 대검찰청은 추정하고 있다
[NeMo I 2022-07-08 18:53:07 wer:255] predicted:조폭의 자금으 굴러가는 지하경제 규모마 면 이 일두원에 이르는 것으로 대검찰정등은 추석하고 있다
[NeMo I 2022-07-08 18:53:32 wer:253]

[NeMo I 2022-07-08 18:53:32 wer:254] reference:당초 이를 근거로 발사에 실패한 것 아니냐는 관측도 나왔지만 우리 군 당국과 북한은 위성이 우주궤도 진입에 성공했다고 발표했다
[NeMo I 2022-07-08 18:53:32 wer:255] predicted:단소위를근거로 발사에 실패한 것 아니냐는 관측도 나왔지만 우리군 당국과 북한은 우선의 오조과고에 지입에 성공했다고 밝혔다
[NeMo I 2022-07-08 18:53:56 wer:253]

[NeMo I 2022-07-08 18:53:56 wer:254] reference:김현정 혹시 매장에서 벌어진 일이 아니더라도 대표님께서 저건 정말 아닌데 라고 느껴졌던 경험이 있나 모르겠어요
[NeMo I 2022-07-08 18:53:56 wer:255] predicted:김현정 수 여장에서 고로진이가 아이더라도 대표님께서 적원 처아 아닌 대어 못두게졌던 경어이 다고르겠어
[NeMo I 2022-07-08 18:54:21 wer:253]

[NeMo I 2022-07-08 18:54:21 wer:254] reference:우리는 멕시코에 사는 레질리 케슬린 알론드라라고 해요
[NeMo I 2022-07-08 18:54:21 wer:255] predicted:우리는 매시코에사는 레질리카술린 안론들라라고 해요
[NeMo I 2022-07-08 18:54:45 wer:253]
```



# Nemo ASR 실습

## ■ 6) 학습

- Epoch 50

```
[NeMo I 2022-07-08 19:01:20 wer:254] reference:그러면 동네 아주머니가 창문 밖으로 고개를 내밀어 안드레아를 부르고는 날이 무더진 칼 몇 개를 들고 옵니다
[NeMo I 2022-07-08 19:01:20 wer:255] predicted:그러면 동네 아주머니가 창문 밖꾸로 보개를 내밀어 안들리마를 그어고는 나이 무여진 칼 역개를 들고 옵니다
[NeMo I 2022-07-08 19:01:45 wer:253]

[NeMo I 2022-07-08 19:01:45 wer:254] reference:일반적으로 반죽을 만들 때는 우유나 물 또는 우유와 물을 섞어서 사용합니다
[NeMo I 2022-07-08 19:01:45 wer:255] predicted:이그한적으로 반주를 면들 되는 오유나을 돈는 우유와보를 사것서나 해 건니다
[NeMo I 2022-07-08 19:02:10 wer:253]

[NeMo I 2022-07-08 19:02:10 wer:254] reference:그리고 이내 우리 아들 리언에게도 하느님의 말씀에 순종하는 것이 얼마나 유익한지 가르쳐 주기 시작했어요
[NeMo I 2022-07-08 19:02:10 wer:255] predicted:그리고 인에 우리 아들 미원에게도 하느님의 말씀에 순종하는 것이 어마나이미 한지 가르저중기 시작했어요
[NeMo I 2022-07-08 19:02:34 wer:253]
```

- 학습 종료 후, save\_to() method를 통해 모델 저장

```
char_model.save_to(save_path="ASR.nemo")
```


# Nemo ASR 실습

## ■ 7) 테스트

- 저장된 모델을 `restore_from()` method로 불러온 뒤,
- `transcribe` method를 통해 음성인식 결과를 출력함
- Jiwer에서 CER(Character Error Rate)를 가져와서 정답 script와 비교해 성능 측정
- 50 epoch 기준 약 20%의 CER을 보임

```
char_model = nemo_asr.models.ASRModel.restore_from("ASR.nemo")

result = char_model.transcribe(test_ds['audio_filepath'])

Transcribing: 100%  115/115 [00:06<00:00, 16.37it/s]

from jiwer import cer

cer(test_ds['text'], result)

0.2000235552763819
```

## ■ 추가 성능 개선 방안

- Output unit 변경

- 본 실습은 26개의 알파벳을 대상으로 학습한 모델에 대해 1203개의 한국어 음절로 tuning한 모델임
  - \* 26개 중에 하나를 맞췄던 모델로 1203개 중 하나를 맞추라고 하는 상황
  - \* Output unit을 변경하면 성능 개선을 기대할 수 있음
    - Ex> grapheme ('ㄱ', 'ㅏ', 'ㄴ')

- 언어 모델 적용

- End-to-end 모델은 음향 모델의 역할만을 수행하는 모델임
- 좋은 성능을 보이는 언어 모델을 적용하면 성능 개선을 기대할 수 있음
  - \* Ex> shallow fusion, deep fusion

- Nemo에서 full recipe로 학습을 수행하는 방법
  - 임의의 데이터에 대해 같은 모델을 처음부터 학습하는 방법
  - Colab에서는 동작이 어려움 or 시간 소요가 큼
- `python example/asr/asr_ctc/speech_to_text_ctc.py \`  
    `--config-path=<path to dir of configs> \`  
    `--config-name=<name of config without .yaml> \`  
    `model.train_ds.manifest_filepath="<path to manifest file>" \`  
    `model.validation_ds.manifest_filepath="<path to manifest file>" \`  
    `trainer.devices=1 \`  
    `trainer.accelerator='cpu' \`  
    `trainer.max_epochs=50`