MINDs Lab

English TTS

AICS 윤서영 매니저

maum.ai

MINDs Lab

1. 학습 준비

maum.ai

1. 필요 기본 지식

1. 음성 합성을 위해 필요한 데이터 구성

Wav	txt
22k (22050), wav format	UTF-8 without BOM

2. 학습을 위해 필수로 알아야 하는 리눅스 명령어

1) cd.. : 이전 디렉토리로 이동
2) cd 경로 : 해당 경로로 이동
3) cp 파일 경로/이름: 원하는 파일을 원하는 경로에 붙여넣기
4) mv 파일 이동경로: 원하는 파일을 원하는 경로에 옮기기
5) ls: 현재 디렉토리의 파일 보기
6) pwd: 현재 위치한 경로 확인
7) Ctrl + c : 종료 / 나가기 / 취소
8) vi : python, json, yaml 파일 등을 보고 수정하는 명령어

2. TTS docker 설정

1. Docker 버전 확인 방법

- 버전 확인 페이지 (repo) : https://docker.maum.ai:8443/

2. Docker image 받기

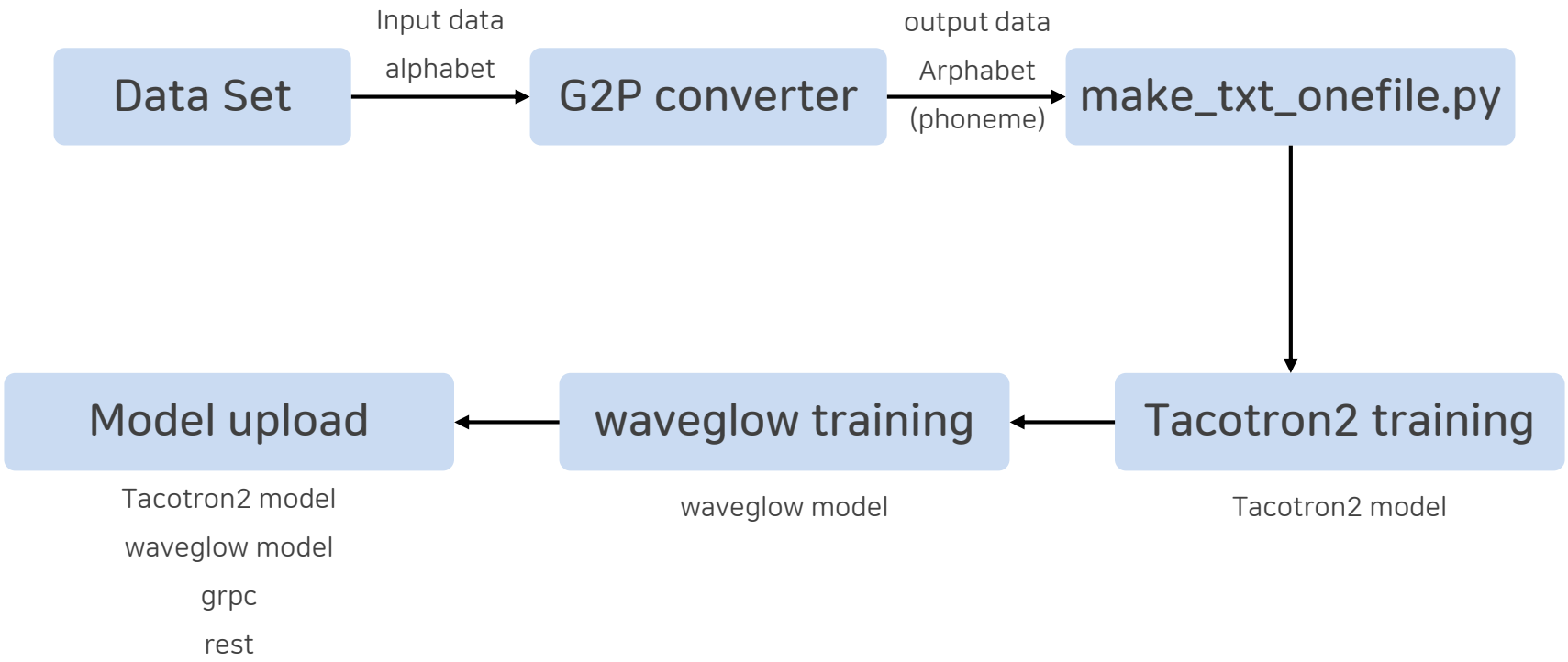
1) Tacotron2 : docker pull docker.maum.ai:443/brain/tacotron2:1.2.7
2) Waveglow : docker pull docker.maum.ai:443/brain/waveglow:1.2.7

※ 1.2.7 = image version

참고) Docker 명령어

1) docker images : 현재 서버에 설치 되어있는 docker image list를 보여줌
2) docker ps : 현재 서버에 떠있는 docket container list를 보여줌 (running)
3) docker ps –a : 서버에 있는 모든 docker container list를 보여줌
4) docker stop : 현재 떠있는 docker container를 죽임
5) docker rm : 죽어있는 docker container를 삭제 (죽어있는 container에만 적용됨)
6) docker exec –it **container_name** bash : 컨테이너 접속
7) Ctrl + p / q : container에서 나가기

3. English TTS pipeline



2. TTS 학습

1. TTS docker 설정

1. Docker container 만들기

1) tacotron2

docker run -itd --ipc host --gpus '"device=**0,1**"' -v **/DATA1**:/DATA1-p **6006**:6006 -e LC_ALL=C.UTF-8 --name **container_name** docker.maum.ai:443/brain/tacotron2:1.2.7

2) Waveglow

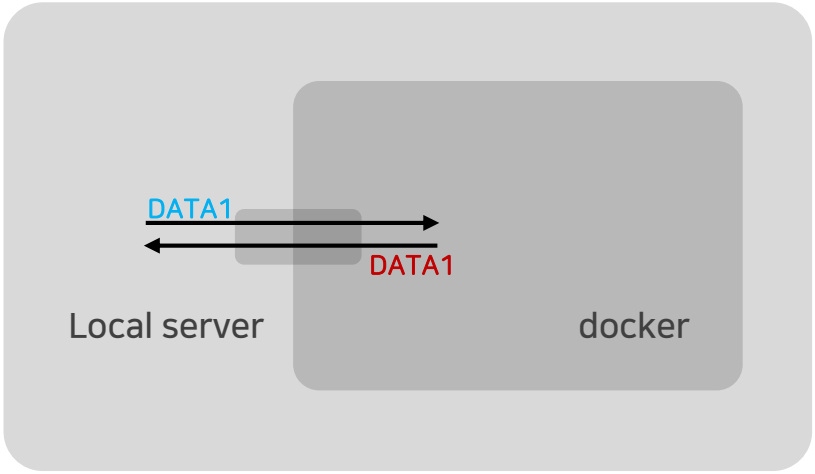
docker run -itd --ipc host --gpus '"device=**0,1**"' -v /DATA1:/DATA1 -p **6007**:6006 -e LC_ALL=C.UTF-8 --name **container_name** docker.maum.ai:443/brain/waveglow:1.2.7

3) g2p

docker run -d --gpus '"device=0"' -p 19001:19001 -e LC_ALL=C.UTF-8 --name **eng_g2p** docker.maum.ai:443/brain/g2p/eng:1.2.0-server

- gpus : 사용할 gpu number
- v : local server와의 공유 폴더
- p : local server와의 공유 포트
- name : 사용할 container 이름

※ local server 및 container 이름은 겹치면 안됨



2. 영어용 데이터 준비

1. g2p_convert.py

- alphabet으로 구성되어 있는 본문 데이터를 stress값 (강세)이 없는 phoneme (arphabet) 데이터로 변경이 필요
- **alphabet → phoneme (arphabet) → stress값 빼기 = 학습용 데이터**
- python g2p_converter.py -i **/input data full path** -o **/output data full path**
- 데이터 변경 예시)

  변경전 Let's count! > 변경 후 {L EH T S} {K AW N T}!

2. English baseline 바꾸기

- 기존 baseline을 사용하지 않고 convert_eng2cmu.py를 돌린 baseline을 사용
- checkpoint_74000_22k_eng_base_eva_sg_cmu

3. make_txt_onefile.py

- 학습 데이터를 train / test / validation으로 나누는 과정이며, 해당 파일을 활용하여 자동으로 분류 가능
- 수정 필요 사항

  1) filelist = 원본 txt 파일의 절대 경로 작성

  2) file_name = os.path.join(~) → 데이터 폴더명과 분류될 아웃풋의 이름 설정

  ※ 모두 통일해주면 헷갈리지 않고 좋다

3. TTS docker 학습

1. tacotron2 학습

- tacotron2의 경우 1, 2차 학습으로 나눠짐
- 경로) cd /root/tacotron2
- python -m multiproc -l **log_path** -o **ckpt_path** trainerd.py -c **baseline_root** **--warm_start** --hparams=distributed_run=True,resource_root='**~resources_root**','training_files='**data_folder**/train.txt',validation_files='**data_folder**/val.txt',speaker=['**speaker ID**'],mask_padding=True,use_eos=False,batch_size=**16**
- 1차 학습시 '—warm_start'를 붙여 학습하며, 보통 checkpoint 500 ~ 1000 사이에 학습 종료
  2차 학습시 '—warm_start'를 제거 후, 1차 학습에서 생성된 checkpoint를 baseline으로 변경하여 학습
- l : log를 쌓을 경로
- o : checkpoint를 쌓을 경로
- c : 사용할 baseline 경로
- resources_root : 원본 데이터 경로 ~ resources/
- training_files : ~train.txt가 있는 상대경로 (예, X/X_train.txt)
- validation_files : ~val.txt가 있는 상대경로 (예, Y/Y_train.txt)
- speaker = 사용할 speaker ID

※ baseline : 2019_09_02_fp16_22k_kor_base_2_178000

참고) sample rate error

- sample rate error가 발생한다면, wav 데이터가 있는 내부 폴더에서 하단 명령어 실행

for i in *wav; do ffmpeg -i "$i" -ar 22050 -hide_banner -loglevel panic /output경로/"$i"; done

3. TTS docker 학습

1. waveglow 학습

- fp_16 > false 변경
- python distributed.py –c **train/config.json** –l **log_path** –o **checkpoint_path**
- l : log를 쌓을 경로
- o : checkpoint를 쌓을 경로

```
    "checkpoint_path": "/DATA1/yoong/tts/trained/waveglow/waveglow_194000_m_re"
},
"data_config": {
    "resource_root": "/DATA1/yoong/tts/resources/",
    "training_files":"/DATA1/yoong/tts/resources/Eric/Eric_train.txt",
    "segment_length": 16000,
    "sampling_rate": 22050,
    "filter_length": 1024,
    "hop_length": 256,
    "win_length": 1024,
    "mel_fmin": 0.0,
    "mel_fmax": 8000.0,
    "load_mel_from_disk": false
},
"validation_files": "/DATA1/yoong/tts/resources/Eric/Eric_val.txt",
```

참고) baseline

- 남성) waveglow_194000_m_re
- 여성) waveglow_226000_kss_kva_22k_re / waveglow_260000_eng_22k_re

MINDs Lab

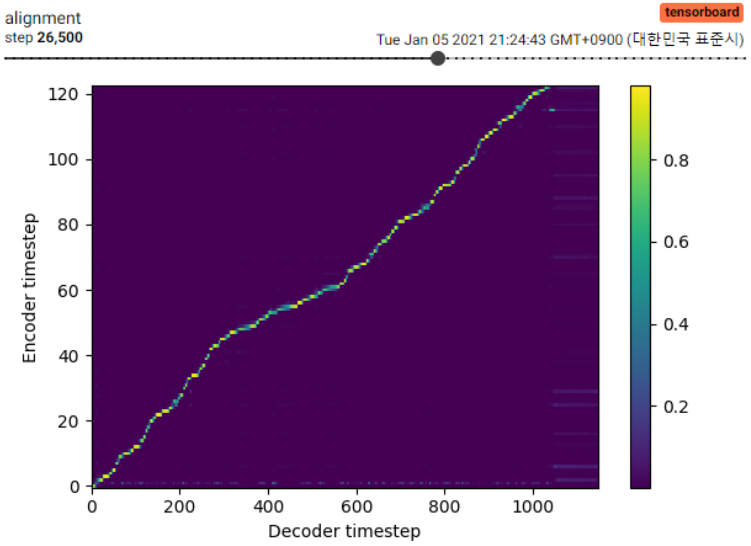
3. tensorboard

maum.ai

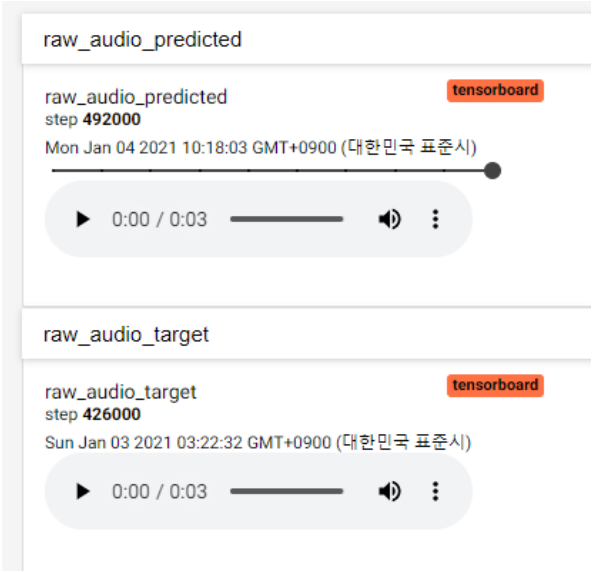
1. tensorboard 띄우기

1. Tensorboard 띄우기

- 경로 : 띄우고자 하는 모델의 log 경로
- tensorboard --samples_per_plugin=scalars=500,images=0 --logdir=.



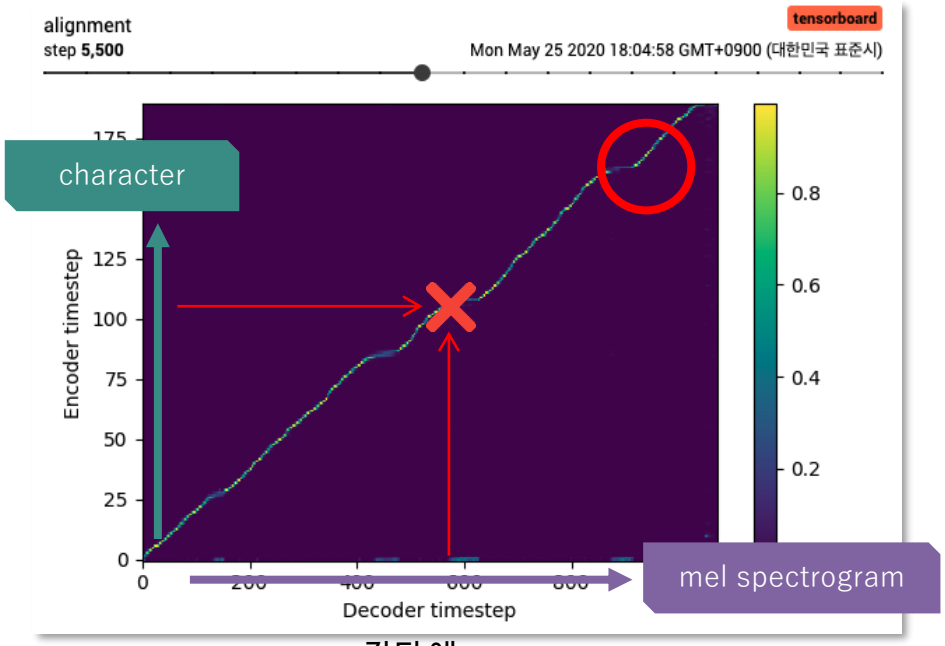
Tacotron2 tensorboard 예시



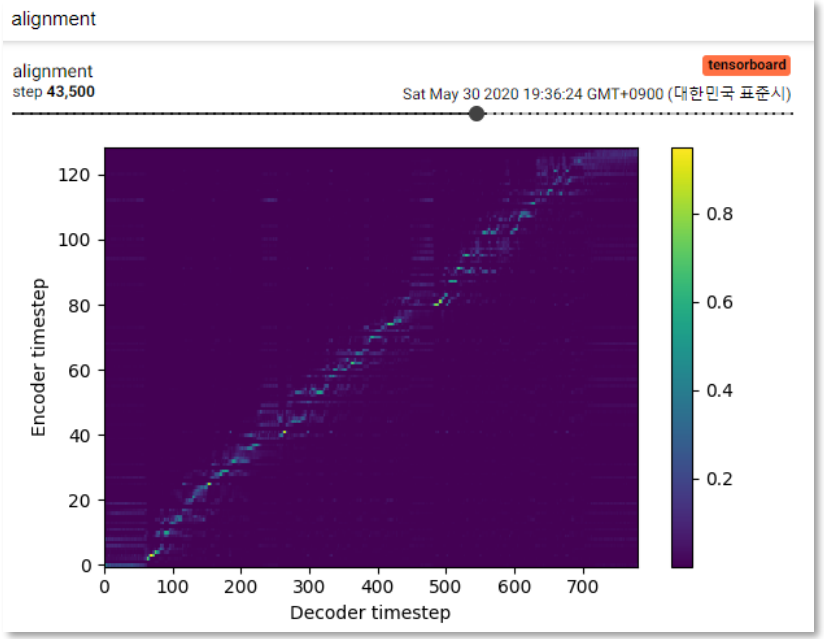
waveglow tensorboard 예시

2. TACOTRON2

1. tensorboard를 통한 최선의 모델 선택 방법 – alignment (attention)



잘된 예

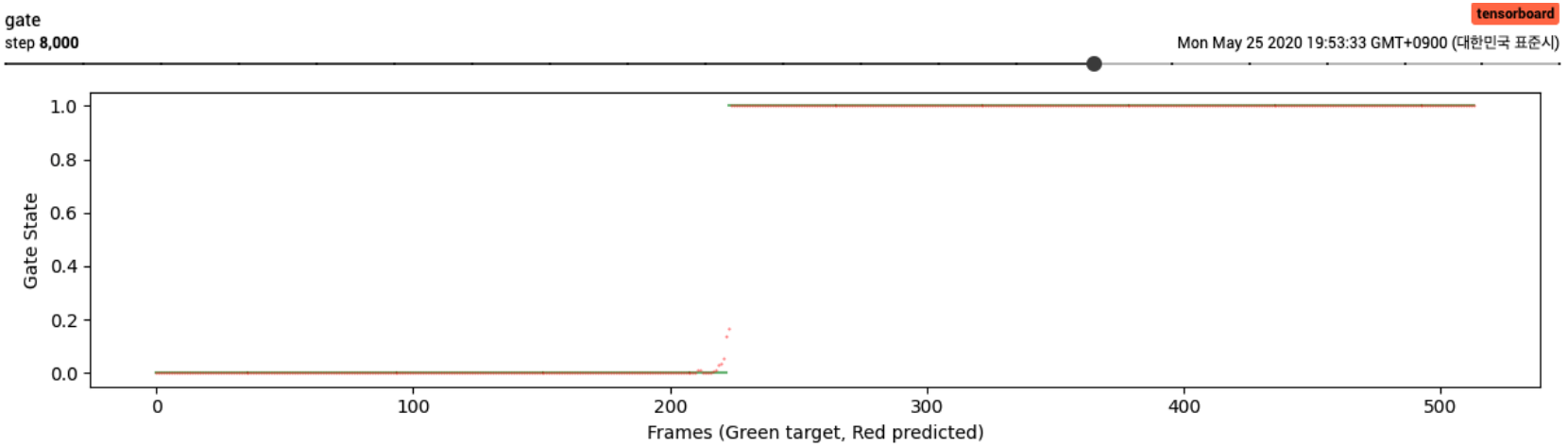


망한 예

- 가늘기 길게 이어져 있을 수록 좋음 (중간에 끊겨버리거나 일자로 줄이 가있는 경우 좋지 않음)
- 상단 동그라미 = 띄어쓰기 구간

2. TACOTRON2

1. tensorboard를 통한 최선의 모델 선택 방법 - gate



- Green Target = 정답지
- Red predict = 학습 결과물

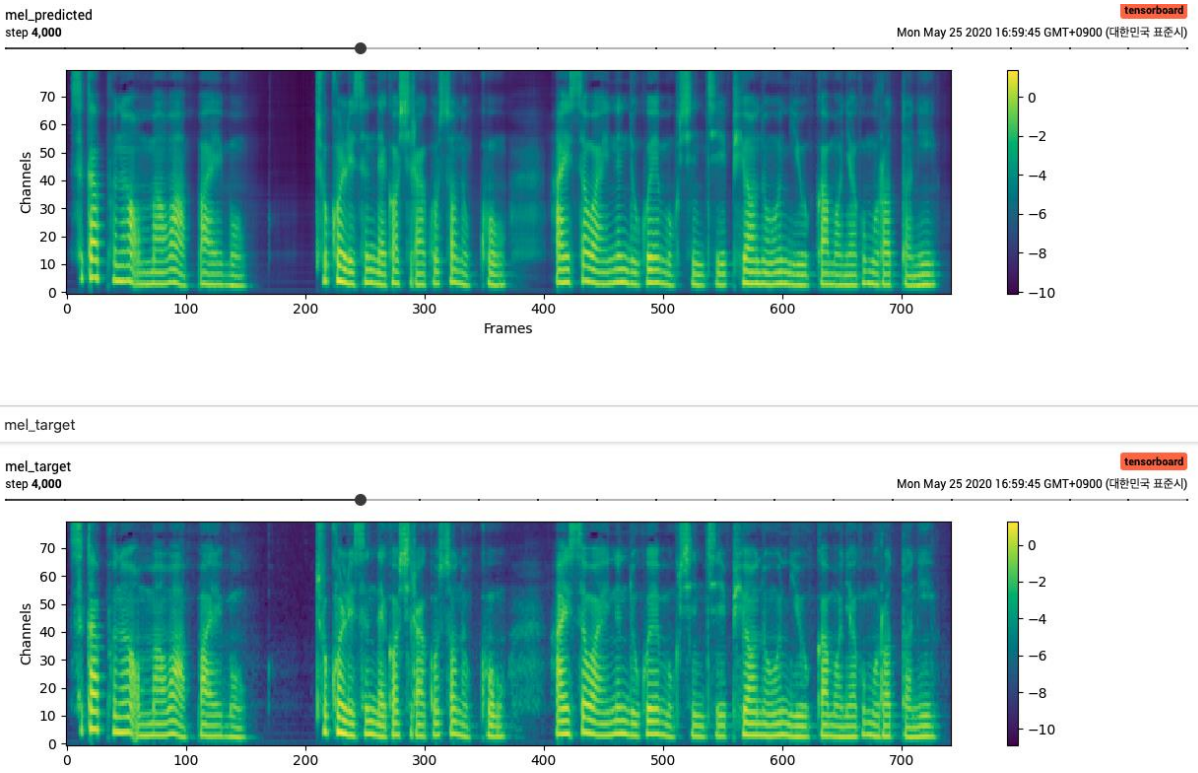
  → 초록색 = 빨간색일 수록 학습이 잘 된 것

2. TACOTRON2

1. tensorboard를 통한 최선의 모델 선택 방법 – mel spectrogram

- Target = 정답지
- Predicted = 학습 결과물

  → predicted = targe일 수록
  학습이 잘나온 것

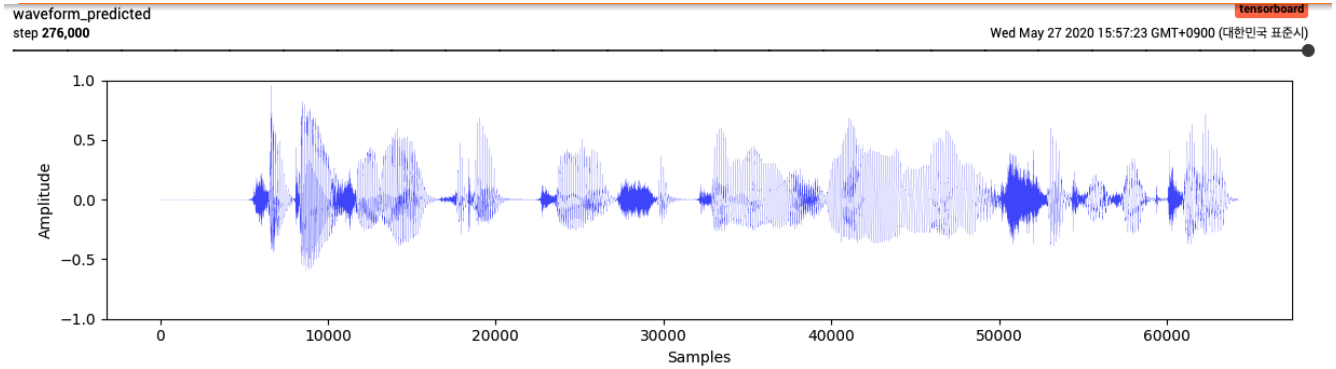


3. waveglow

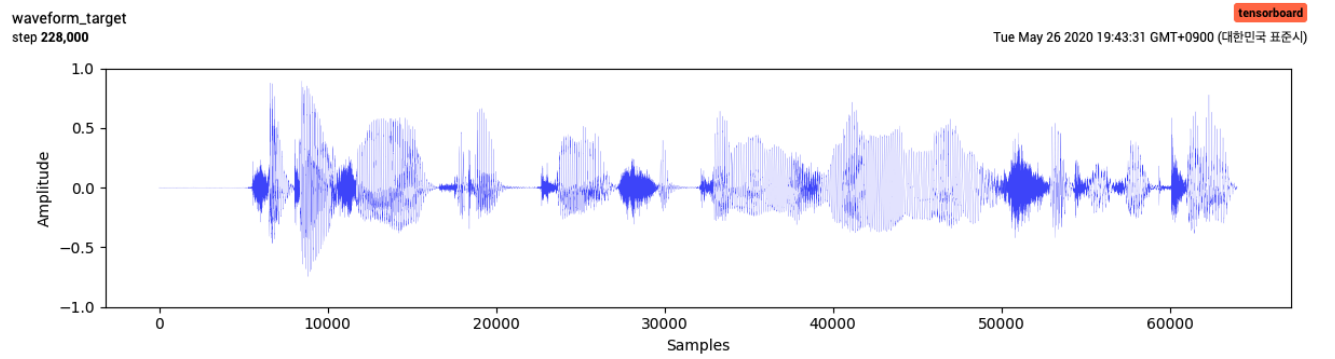
1. tensorboard를 통한 최선의 모델 선택 방법 – waveform

- predicted = 학습 결과물
- target = 정답지

  → predicted = target일 수록 학습이 잘나온 것



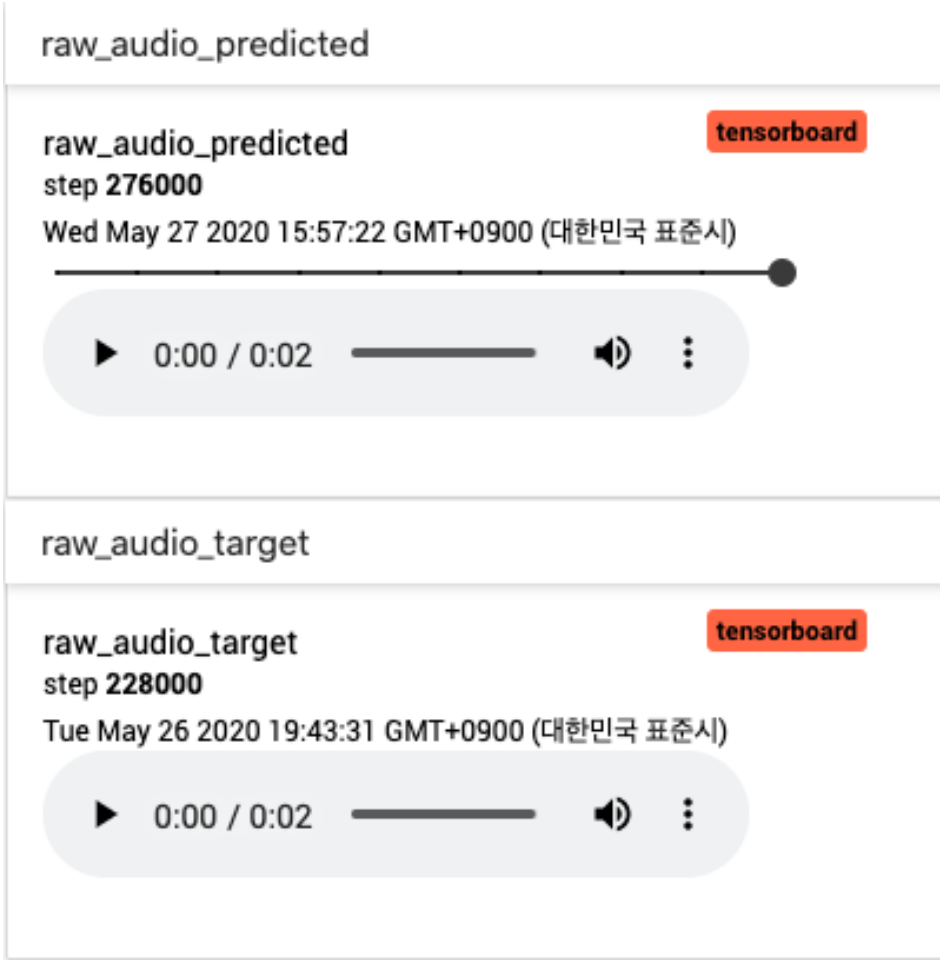
waveform_target



3. waveglow

1. tensorboard를 통한 최선의 모델 선택 방법 – audio

- predicted = 학습 결과물
- target = 정답지
  → predicted = target일 수록 학습이 잘나온 것
- predicted에 금속음, 소음 등이 섞여있을 경우,
  모델 합성 시 98%의 확률로 소음이 섞여 있음
  → predicted에서 최대한 깨끗한 음성 선택 필요

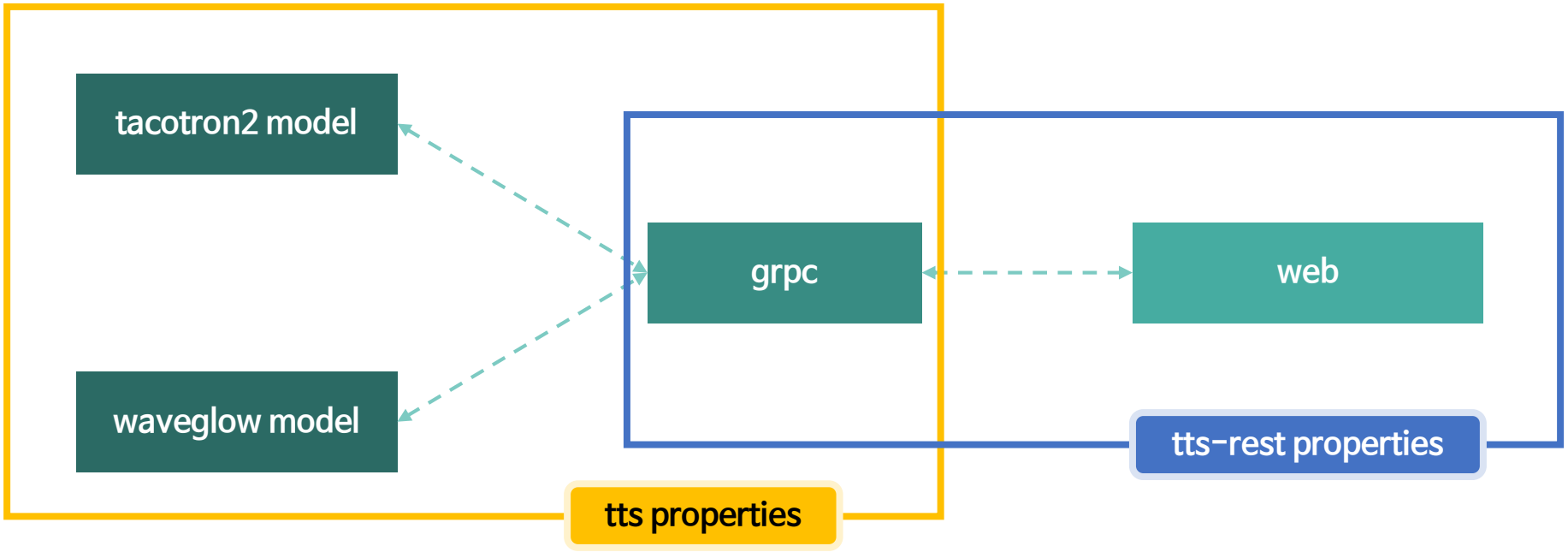


4. Model upload

1. TTS docker model upload

1. 학습 모델 구조

- 학습 모델 구조
  - tacotron2 model, waveglow model, grpc, web으로 구성

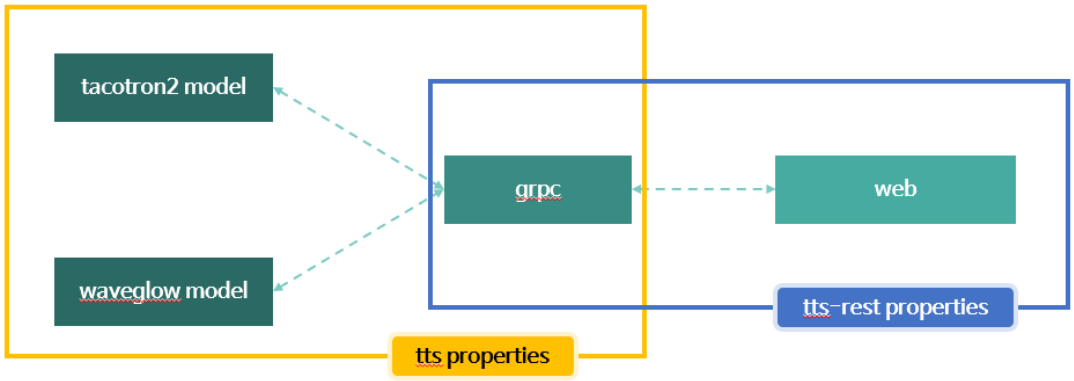


1. TTS docker model upload

1. tacotron2 학습

- Port
  - 2개의 모델과 grpc, web(tts-rest)는 각각의 포트가 필요
  - grpc와 web 포트는 방화벽이 뚫려있어야 함
  - tts properties : taco + wg model과 grpc의 관계 정보
  - tts-rest properties : grpc와 web의 관계 정보

😊 tts properties와 tts-rest properties만 수정해주면 됨



1. TTS docker model upload

1. tacotron2

docker run --gpus '"device=**0**"' -d -v **/DATA1/tts/trained/tacotron2**:/model -p **30101**:30001 -e LC_ALL=C.UTF-8 -e TACOTRON2_MODEL=/model/**CHECKPOINT_NAME** -e TACOTRON2_THRESHOLD=**0.1** -e TACOTRON2_MAX_DECODER_STEPS=**1600** --name **CONTAINER_NAME** docker.maum.ai:443/brain/tacotron2:1.2.7-server

- device : 사용할 gpu num
- v : input data path (:docker model path)
- p : taco server port
- e : checkpoint name / t값 / s값
- name : container name

※ 옵션 기본값

TACOTRON2_THRESHOLD = 0.1

TACOTRON2_MAX_DECODER_STEPS = 1600

2. waveglow

docker run --gpus '"device=1"' -d -v **/DATA1/tts/trained/waveglow**:/model -p **35101**:35001 -e LC_ALL=C.UTF-8 -e WAVEGLOW_MODEL=/model**/waveglow_228000_SON** -e WAVEGLOW_SIGMA=**0.66** -e WAVEGLOW_DENOISER_STRENGTH=**0.01** -e WAVEGLOW_VOLUME=1.0 --name CONTAINER_NAME docker.maum.ai:443/brain/waveglow:1.2.7-server

- 상단과 동일

※ 옵션 기본값

WAVEGLOW_SIGMA = 0.66

WAVEGLOW_DENOISER_STRENGTH = 0.01

1. TTS docker model upload

3. grpc

```
docker run -d -p 9999:9999 -e LC_ALL=C.UTF-8 -e GRPC_ADDR_TACOTRON=172.17.0.1:30101 -e
GRPC_ADDR_WAVEGLOW=172.17.0.1:35101 -e GRPC_ADDR_G2P=172.17.0.1:19001 -e MAX_SPEAKER=1 --name
grpc_test docker.maum.ai:443/brain/tts:1.2.7-server
```

- d : daemon (백그라운드로 실행)
- p : grpc port num
- grpc_addr_taco : taco grpc ip:port
- grpc_addr_wav : waveglow grpc ip:port
- grpc_addr_kong : konglish grpc ip:port
- max_speaker : speaker 개수
- max_length : long text 처리를 위한 기준값

※ IP port 기본값

ENV GRPC_ADDR_TACOTRON=172.17.0.1:30001

ENV GRPC_ADDR_WAVEGLOW=172.17.0.1:35001

ENV GRPC_ADDR_KONGLISH=172.17.0.1:20001

ENV GRPC_ADDR_G2P=172.17.0.1:19001

4. rest

```
docker run -d -p 9998:9998 -e LC_ALL=C.UTF-8 -e GRPC_NAME_TTS=disaster -e GRPC_IP_TTS=172.17.0.1 -e
GRPC_PORT_TTS=9999 --name CONTAINER_NAME  docker.maum.ai:443/brain/tts:1.2.5-rest
```

- GRPC_NAME_TTS = rest 흰페이지에 뜨는 speakerID 값
- p : server port
- name : tts-rest container name

※ 옵션 기본값

WAVEGLOW_SIGMA = 0.66

WAVEGLOW_DENOISER_STRENGTH = 0.01


MINDs Lab
Thank you
maum.ai