KOREA UNIVERSITY

Artificial Intelligence Laboratory

Voice Conversion Experiment

Artificial Intelligence Laboratory

Department of Computer Science and Engineering, Korea University

INDEX

• 코드 베이스 & 데이터 세트 준비

- 코드 베이스 다운:
  git clone https://github.com/hpjang/CycleVAE.git
  cd CycleVAE

- 가상환경 생성
  virtualenv env
  source env/bin/activate

- 학습 데이터 세트 다운: https://datashare.is.ed.ac.uk/handle/10283/3061
  - training data for building parallel and non-parallel VC systems released to participants (117.0Mb) 다운로드 후 압축 해제
    wget https://datashare.ed.ac.uk/bitstream/handle/10283/3061/vcc2018_database_training.zip
    vcc2018_training 폴더를 corpus 폴더에 train이라는 이름으로 저장

  - evaluation data (source speaker's data) released to participants (31.79Mb) 다운로드 후 압축 해제
    wget https://datashare.ed.ac.uk/bitstream/handle/10283/3061/vcc2018_database_evaluation.zip
    vcc2018_evaluation 폴더를 corpus 폴더에 test라는 이름으로 저장

  - 한국어 데이터 다운
    https://drive.google.com/drive/folders/1ZqzwDvtHx8Gob7uQdbXyZ4ihtU8qZhGj?usp=sharing

- **데이터 전처리**

- python3 preprocess/preprocess-vcc2018.py

```
(venv) hpchang@speech004:~/speech004/course/VAE_experiment$ python3 preprocess/preprocess-vcc2018.py
/home/hpchang/.local/lib/python3.6/site-packages/numba/errors.py:137: UserWarning: Insufficiently rec
ound. Numba requires colorama >= 0.3.9
  warnings.warn(msg)
Loading VCC2SF1 Wavs...
Processing 10026
Processing 10058
Processing 10048
Processing 10070
Processing 10024
Processing 10053
Processing 10037
Processing 10050
Processing 10041
Processing 10073
Processing 10052
Processing 10028
Processing 10051
Processing 10008
Processing 10016
```

- **학습 -> 변환**

- nohup bash run_all.sh > vae_exp_0520.txt &
  코드를 백그라운드로 실행 하고 터미널 창에 출력되는 log를 vae_exp_0520.txt 파일에 저장 (이후 loss graph 출력에 필요)

```
.....................
EPOCH: 998
Train: rec_loss : 1.0582 / kl_loss : 1.0234 / total_loss : 1.1606 /
DEV: rec_loss : 1.0749 / kl_loss : 1.0092 / total_loss : 1.1758 /
.....................
EPOCH: 999
Train: rec_loss : 1.0627 / kl_loss : 1.0032 / total_loss : 1.1631 /
DEV: rec_loss : 1.0806 / kl_loss : 1.0193 / total_loss : 1.1825 /
.....................
EPOCH: 1000
Train: rec_loss : 1.0598 / kl_loss : 1.0077 / total_loss : 1.1606 /
DEV: rec_loss : 1.0772 / kl_loss : 1.0025 / total_loss : 1.1775 /
.....................
***********************************
Model name: VAE3
TIME PER EPOCH: 0.34576888803716427
Final Epoch: 870 1.1703
***********************************
```

학습 완료

```
Processing VCC2SF1
        Convert 30023.wav ...
convert.py:112: RuntimeWarning: divide by zero encountered in log
  logf0_norm = (np.log(f0)-logf0_m_s) / logf0_s_s
convert.py:116: UserWarning: Creating a tensor from a list of numpy
the list to a single numpy.ndarray with numpy.array() before conver
c/utils/tensor_new.cpp:201.)
  y = torch.Tensor(style).float().cuda().contiguous()
        Convert 30033.wav ...
        Convert 30012.wav ...
        Convert 30014.wav ...
        Convert 30020.wav ...
        Convert 30027.wav ...
        Convert 30008.wav ...
        Convert 30034.wav ...
        Convert 30018.wav ...
        Convert 30004.wav ...
        Convert 30016.wav ...
        Convert 30025.wav ...
        Convert 30006.wav ...
```

학습된 모델로 convert wav 생성

학습 시간 (1000epoch 기준)

- 화자 4명(Many-to-Many)
  VAE만 학습시: cpu: 21분, gpu: 6분 / CycleVAE 추가 학습시: cpu: 120분, gpu: 30분
- 학습된 모델 저장 위치: model

- **평가**

- bash run_all.sh

```
SRC:  VCC2SM1  TAR:  VCC2SF1  MCD_MEAN:  7.593974159259488  MCD_STD:  0.350554100361027
SRC:  VCC2SM1  TAR:  VCC2SF1  MSD_MEAN:  1.902580215049465  MSD_STD:  0.029426708477566312
SRC:  VCC2SF1  TAR:  VCC2SM1  MCD_MEAN:  6.980727672878368  MCD_STD:  0.37203821625817995
SRC:  VCC2SF1  TAR:  VCC2SM1  MSD_MEAN:  1.9736052798898303  MSD_STD:  0.08441122685197619
SRC:  VCC2SM2  TAR:  VCC2SF1  MCD_MEAN:  7.65695623433364  MCD_STD:  0.4807572187494584
SRC:  VCC2SM2  TAR:  VCC2SF1  MSD_MEAN:  1.9495598935602498  MSD_STD:  0.06279243745901182
SRC:  VCC2SF2  TAR:  VCC2SF1  MCD_MEAN:  7.386005141834283  MCD_STD:  0.4595518749754665
SRC:  VCC2SF2  TAR:  VCC2SF1  MSD_MEAN:  1.934054872803487  MSD_STD:  0.05058298172151311
SRC:  VCC2SM2  TAR:  VCC2SF2  MCD_MEAN:  7.105956656891425  MCD_STD:  0.49194933297739646
SRC:  VCC2SM2  TAR:  VCC2SF2  MSD_MEAN:  1.9423796324210352  MSD_STD:  0.06807869985790602
SRC:  VCC2SM2  TAR:  VCC2SM1  MCD_MEAN:  6.823073823310213  MCD_STD:  0.33703384743779297
SRC:  VCC2SM2  TAR:  VCC2SM1  MSD_MEAN:  1.9757439332943996  MSD_STD:  0.1050774254623631
SRC:  VCC2SF2  TAR:  VCC2SM2  MCD_MEAN:  7.111056955018541  MCD_STD:  0.5083103145752539
SRC:  VCC2SF2  TAR:  VCC2SM2  MSD_MEAN:  1.9406500737873822  MSD_STD:  0.06267347860553019

------------------------------
total   MCD_MEAN:  7.084930050622954  MCD_STD:  0.5147675500656881
f2f   MCD_MEAN:  7.119929312382638  MCD_STD:  0.5211972090083633
f2m   MCD_MEAN:  6.96107014162409  MCD_STD:  0.4683025064832528
m2f   MCD_MEAN:  7.282704613306037  MCD_STD:  0.5575358742507932
m2m   MCD_MEAN:  6.902101481494829  MCD_STD:  0.33822615540337303

------------------------------
total   MSD_MEAN:  1.9377954234729828  MSD_STD:  0.06847091851848633
f2f   MSD_MEAN:  1.9373253187841328  MSD_STD:  0.05682726863161494
f2m   MSD_MEAN:  1.953213429716386  MSD_STD:  0.06961763955119313
m2f   MSD_MEAN:  1.9211578334760746  MSD_STD:  0.05715204352268758
m2m   MSD_MEAN:  1.9407046956688434  MSD_STD:  0.0873875535043844
```

MCD 확인 방법: MCD 측정이 완료되면 stats 폴더에 다음과 같이 정리된 MCD 결과 확인

• 평가 및 Hyperparameter 조정

- 듣기 평가: 오디오 재생
  result 폴더 내 .wav 파일 확인

- Loss 그래프
  run_graph.sh 파일 내 log_path 변수에 학습 로그 파일(슬라이드 p.5 참고) 파일명 입력
  -> graph 폴더 내 .csv 파일을 다운 받아 엑셀 그래프로 생성

- Hyperparameter 조정 방법
  train_further.py 파일 151 line 에서 원하는 Hyperparameter 값을 수정

```
"rec": 1.0, "cyc": 1.0, "si": 0.1, "i": 0.1, "li": 1.0, "ac": 1.0, "sc": 0.1, "kl": 0.1
```

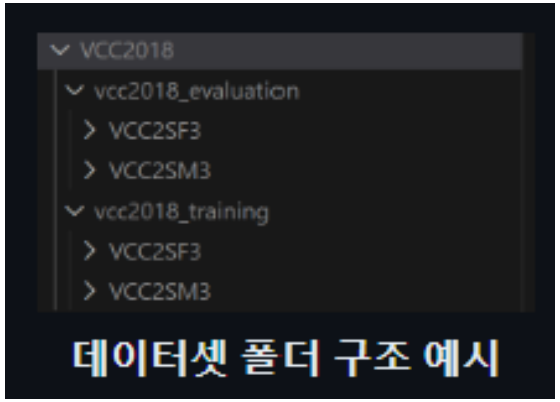
- 수정해 볼 Hyperparameter
  rec: reconstruction loss / cyc: cycle consistency loss / kl: KL-divergence loss

• 코드 베이스 & 데이터 세트 준비

- 코드 베이스 다운:
  git clone https://github.com/chaksseu/MaskCycleGAN-VC.git
  cd MaskCycleGAN-VC

- 아나콘다 환경 설정 및 필요 패키지 설치
  conda env create -f environment.yml
  conda activate MaskCycleGAN-VC

- 학습 데이터 세트 다운: https://datashare.is.ed.ac.uk/handle/10283/3061
  - training data for building parallel and non-parallel VC systems released to participants (117.0Mb) 파일 다운로드 및 압축해제
  - evaluation data (source speaker's data) released to participants (31.79Mb) 파일 다운로드 및 압축해제
  - evaluation data (target speaker's data) used as reference in listening tests (15.98Mb)파일 다운로드 및 압축해제
  - vcc2018폴더를 생성하고 압축해제한 폴더들을 넣습니다.
  - vcc2018_reference 폴더 내의 파일을 모두 vcc2018_evaluation으로 옮긴 후 폴더를 삭제합니다.
  - 개별적인 데이터 학습을 위해서는 vcc2018_evaluation 폴더와 vcc2018_training 폴더 아래에 화자 폴더를 생성한 후、 그 안에 wav파일을 넣으시면 됩니다.



깃허브 링크: https://github.com/chaksseu/MaskCycleGAN-VC

- **데이터 전처리**

- training dataset 전처리
- python data_preprocessing/preprocess_vcc2018.py --data_directory vcc2018/vcc2018_training --preprocessed_data_directory vcc2018_preprocessed/vcc2018_training --speaker_ids VCC2SF2 VCC2SM2

- evaluation dataset 전처리
- python data_preprocessing/preprocess_vcc2018.py --data_directory vcc2018/vcc2018_evaluation --preprocessed_data_directory vcc2018_preprocessed/vcc2018_evaluation --speaker_ids VCC2SF2 VCC2SM2

- --speaker_ids 뒤에 적은 화자의 데이터셋이 전처리됩니다.

- 원하시는 화자의 이름을 speaker_ids 뒤에 넣으실 수 있습니다.

- 전처리 데이터 저장 위치:
  - '/vcc2018_preprocessed/vcc2018_evaluation',
  - '/vcc2018_preprocessed/vcc2018_training'

- **학습**

- <speaker_A_id>를 source로 <speaker_B_id>를 target으로 하는 모델을 학습시킵니다. 최소 수백 epochs 이상의 학습을 권장합니다.

- python -W ignore::UserWarning -m mask_cyclegan_vc.train --name mask_cyclegan_vc_<speaker_id_A>_<speaker_id_B> --seed 0 --save_dir results --preprocessed_data_dir vcc2018_preprocessed/vcc2018_training --speaker_A_id <speaker_A_id> --speaker_B_id <speaker_B_id> --epochs_per_save 10 --epochs_per_plot 10 --num_epochs 6172 --batch_size 1 --decay_after 1e4 --sample_rate 22050 --num_frames 64 --max_mask_len 25 --gpu_ids 0

- 학습된 모델 저장 위치:
  - '/results/mask_cyclegan_vc_<speaker_id_A>_<speaker_id_B>/ckpts'

- 학습/테스트 데이터 에러 그래프 생성/확인 방법:
  - /results/mask_cyclegan_vc_<speaker_id_A>_<speaker_id_B>/mask_cyclegan_vc_<speaker_id_A>_<speaker_id_B>.log 파일의 모든 값을 복사하여 loss_graph.py 파일의 data에 붙여넣기
  - python loss_graph.py

- 노트북/서버에서 각각 학습 시간 (6172epoch 기준)
  - 화자 2명(one-to-one), 화자당 81문장의 경우:
    - 노트북(cpu): 대략 520시간, 서버(gpu): 52시간

- **모델 테스트(오디오 생성)**
- 훈련시킨 MaskCycleGAN-VC 모델을 evaluation dataset으로 test합니다.
- converted audio는 다음 위치에 저장됩니다. 'results/<name>/converted_audio'
- python -W ignore::UserWarning -m mask_cyclegan_vc.test --name mask_cyclegan_vc_<speaker_A_id>_<speaker_B_id> --save_dir results/ --preprocessed_data_dir vcc2018_preprocessed/vcc2018_evaluation --gpu_ids 0 --speaker_A_id <speaker_A_id> --speaker_B_id <speaker_B_id> --ckpt_dir results/mask_cyclegan_vc_<speaker_A_id>_<speaker_B_id>/ckpts --load_epoch <recent epoch> --model_name generator_A2B
  - <speaker_A_id>: source 화자
  - <speaker_B_id>: target 화자
  - <recent epoch>: 마지막으로 저장된 모델의 epoch 수
- MCD 확인 방법:
  - # 새로운 conda 환경 생성 및 python 파일 실행
  - conda create -n mcd python==3.8
  - conda activate mcd
  - pip install pymcd tqdm
  - # 'cal_pymcd.py'에서 GT path와 Converted path 설정 후
  - python cal_pymcd.py
- 결과 파일 재생 방법:
  - results/mask_cyclegan_vc_<speaker_id_A>_<speaker_id_B>/converted_audio에서 원하는 wav파일을 다운받아 재생

• 코드 베이스 & 데이터 세트 준비

- 코드 베이스 다운:
  git clone https://github.com/chaksseu/DiffVC.git
  cd DiffVC

- 아나콘다 환경 설정 및 필요 패키지 설치
  conda create -n DiffVC python==3.6.13
  conda activate DiffVC
  pip install -r requirements.txt
  pip install torchaudio

- 학습 데이터 세트 다운: https://datashare.ed.ac.uk/handle/10283/2651
  - 데이터셋의 크기가 상당히 크기 때문에 다른 개별적인 데이터를 학습에 이용하셔도 좋습니다. (이전의 vcc2018 training 데이터셋도 사용 가능)
  1. 'data' directory 생성 '
  2. data' directory 아래에 "wavs", "mels", "embeds" 이름의 폴더 생성
  3. 학습 시킬 raw 오디오 파일(.wav)들을 "wavs" 폴더에 삽입
  4. get_mels_embeds.py 또는 inference.ipynb 파일을 이용하여 mels, embeds 계산

완성된 데이터셋 폴더 구조(예시)

```
├─data
│  ├─embeds
│  │  ├─p229
│  │  └─p238
│  ├─mels
│  │  ├─p229
│  │  └─p238
│  └─wavs
│     ├─p229
│     └─p238
```

- 전처리 데이터 저장 위치: '/data/embeds', '/data/mels'

깃허브 링크: https://github.com/chaksseu/DiffVC

- **사전 학습모델 다운로드**

- 이번 실험에서는 Decoder만을 학습시킬 것이기에 vocoder 및 encoder는 사전학습 된 모델을 다운받아 사용합니다.

- 1. 사전 학습된 HiFi-GAN vocoder 다운로드
  - https://drive.google.com/file/d/10khlrM645pTbQ4rc2aNEYPba8RFDBkW-/view?usp=sharing
  - 위 링크에서 다운로드 후, checkpts/vocoder/에 넣으시면 됩니다.

- 2. 사전 학습된 Encoder 다운로드
  - https://drive.google.com/file/d/1JdoC5hh7k6Nz_oTcumH0nXNEib-GDbSq/view?usp=sharing
  - "logs_enc" directory를 생성하여, 위 링크에서 다운 받은 파일을 넣으시면 됩니다.

- **디코더 학습**

- python train_dec.py

- 학습된 모델 저장 위치:
  - 'logs_dec/'

- 학습/테스트 데이터 에러 그래프 생성/확인 방법:
  - 'logs_dec/train_dec.log' 파일의 값을 복사하여 'loss_graph.py' 파일의 data에 붙여넣기
  - python loss_graph.py

- 노트북/서버에서 각각의 학습 소요 시간 (200epoch 기준):
  - 화자 100명, 화자당 180-350문장의 경우:
    - 노트북(cpu): X(사실상 불가능), 서버(gpu): 100시간
  - 화자 2명, 화자당 100문장의 경우:
    - 노트북(cpu): 60시간, 서버(gpu): 40분

- **모델 테스트(오디오 생성)**
- _inference.py 및 inference.ipynb 참고
- 학습시킨 모델을 사용하여 Voice Conversion을 진행
- (경로 설정 예시)
  - vc_path = 'logs_dec/vc_200.pt' (학습시킨 모델 경로)
  - src_path = 'data/p226/p226_005_mic1.wav' (source wav 파일 경로)
  - tgt_path = 'data/p229/p229_002_mic1.wav' (target wav 파일 경로)
  - output_path = 'converted_audio/converted.wav' (변환된 음성이 저장될 경로)
- 결과 저장 위치:
  - 'converted_audio/converted.wav'
- MCD 확인 방법:
  - # 새로운 conda 환경 생성 및 python 파일 실행
  - conda create -n mcd python==3.8
  - conda activate mcd
  - pip install pymcd tqdm
  - # 'cal_pymcd.py'에서 GT path와 Converted path 설정 후
  - python cal_pymcd.py
- 결과 파일 재생 방법:
  - '/converted_audio'에서 원하는 wav파일을 다운받아 재생