1. AI R&D 실무자양성과정 프로젝트 (18/07/20 ~ 18/09/14)

● 개요: Singing Voice Conversion을 주제로, 가수 볼빨간사춘기의 노래를 박효신으로 또는 화자 손석희, 유인나가 부르는 것처럼 들리도록 변환
● 역할:
"Singing Style Transfer Using Cycle-BEGAN"의 논문을 참고해 "Cycle-GAN Voice Converter"의 코드에 BEGAN과 같이 학습하도록 모델 수정(Discriminator과 Loss function 수정, 하이퍼파라미터 추가). Vocal Separation 모델 조사, 데이터수집, 데이터 전처리.
https://github.com/NamSahng/SingingStyleTransfer
● 환경: 네이버 클라우드 tensorflow ubuntu 서버, jupyter notebook, python(tensorflow)

[1] Singing Style transfer CBEGAN:
- 데이터: log-magnitude spectrogram 1,024-sample, 1/4 overlapping windows for the STFT for songs sampled at 44.1kHz. we view a T × F 2D spectrogram as a T × 1 image with F channels, and use 1D convolution
- 모델 특징: Cycle Gan에 G: U net D: U net , BEGAN의 학습기법 , GRU 추가
- 시도: 남자가수 목소리 <-> 여자가수 목소리 변환

[2] C-GAN VC: NTT통신사 연구논문
- 데이터: MCEPs 와 f0값
- 모델 특징: C: 1D CNN Auto-Encoder with ResNet D: 2D CNN discriminator로 1과 0으로 분류
  Identity Loss & GLU for activation Function.
  a. Identity loss가 없으면 degradation이 일어남 (Linguistic Structure가 collapse됨)
     박효신으로 변환하는 Generator에 박효신을 넣어 박효신노래가(거의 그대로) 나오도록
  b. RNN을 사용하면 좋겠지만 parallel하게 다룰 수 있는 모델이 아니므로 시퀀셜데이터에 병렬화를 허용하는 gated CNN을 사용. gated CNN에서 GLU를 활성함수로 사용했으며 이는 하나의 레이어를 하나는 sigmoid로 보내고 그 둘을 element wise곱을 해, 시퀀셜함을 유지하면서 데이터의 정보가 GLU를 들어오기 이전에 의존해 선택적으로 전파되도록 하는 것으로 알고 있다.

$$\boldsymbol{H}_{l+1} = (\boldsymbol{H}_l * \boldsymbol{W}_l + \boldsymbol{b}_l) \otimes \sigma(\boldsymbol{H}_l * \boldsymbol{V}_l + \boldsymbol{c}_l),$$

- 변환: world decompose로 wav를 읽고 이를 통해 MCEP와 f0를 얻는다. 변환할 때 f0를 pitch conversion이라는 메소드로 f0는 변환 시키고(학습하지 않음), MCEP만 우리의 모델로 학습변환하는 것이다.

$$\mathcal{L}_{full} = \mathcal{L}_{adv}(G_{X\to Y}, D_Y) + \mathcal{L}_{adv}(G_{Y\to X}, D_X) + \lambda_{cyc}\mathcal{L}_{cyc}(G_{X\to Y}, G_{Y\to X}) + \mathcal{L}_{id}(G_{X\to Y}, G_{Y\to X})$$

$$\mathcal{L}_{adv}(G_{X\to Y}, D_Y) = \mathbb{E}_{y\sim P_{\text{Data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x\sim P_{\text{Data}}(x)}[\log(1 - D_Y(G_{X\to Y}(x)))].$$

[3] C-BEGAN Vocal Converter

- 데이터 표현: C-GAN VC를 따랐다. 128 frame(0.5초) 에서 I expanded frames to 512 바꾸었다.

- BEGAN:
  a. W-거리: 이렇게 바꾸어서 D를 지나면 이미지가 나오며, 가짜로 인식해야하는 이미지는 많이 변형되고, 진짜로 인식해야하는 이미지는 최대한 변형이 안되게. 이것이 W-GAN과 EB-GAN에서 한 것. data distribution에 맞춰 학습하는 것이아니라 오토인코더의 loss-distribution에 맞춰 학습
  b. Equilibrium: kt: GD의 학습에서 GENERATION에 집중하도록 D Loss 부분의 generation loss를 조금 약하게 조정. weight decay처럼 감마라는 파라미터에 따라 바뀜
  감마는 GD의 Loss를 또 고려
  c. 또한 Convergence measure를 통해 최종 평형에 이르었는지 mode collapse인지 확인하는 척도
  이를 통해 G가 image(sound) Generation에 더 집중할 수 있도록 한다.

**3-2. Loss Function**

$$\mathcal{L}_{full} = \mathcal{L}_{adv}(G_{X\to Y}, D_Y) + \mathcal{L}_{adv}(G_{Y\to X}, D_X) + \lambda_{cyc}\mathcal{L}_{cyc}(G_{X\to Y}, G_{Y\to X}) + \mathcal{L}_{id}(G_{X\to Y}, G_{Y\to X})$$

where

$$\mathcal{L}_{adv}(G_{X\to Y}) = [D_Y(G_{X\to Y}(x)) - (G_{X\to Y}(x))]$$
$$\mathcal{L}_{adv}(D_Y) = [D_Y(y) - y] - k_t\,[D_Y(G_{X\to Y}(x)) - (G_{X\to Y}(x))]$$

$$k_{t+1} = k_t + \lambda_k\,[\gamma(D_Y(y) - y) - \mathcal{L}(G_{X\to Y})]$$
$$\mathcal{M}_{global} = [D_Y(y) - y] + |\gamma(D_Y(y) - y) - \mathcal{L}(G_{X\to Y})|$$

- 이전 preprocess들:
  a. pydub library을 통한 silence removal 일종의 vad
  b. U-NET + IRM, IBM :
  의료분야에서 셀의 경계 값을 이용해 segmentation을 위해 사용하는 것. 잡음이 심함 musdb 데이터 사용.
  IRM, IBM: 오라클 방법으로 처음에는 하나의 방법인줄 알고 연구했다가 분리된 보컬과 inst가 필요..
  UNET은 일종의 필터를 학습한다.

- 개선점, 아쉬운 점:
  MCEPs 이외의 mag-spec으로 변환 실패 시간부족, 더욱 큰 변화 예상, 기존
  MCEPs는 feature 24 * 512 // spectrogram은 512 * 960

BEGAN학습 기법을 적용하는 것을 발표 1주일 정도 전에 끝났고, spectrogram으로도 하면은 더 많은 변화가 예상되어 꼭 하고 싶었다. 이렇게 비교하는 것이 맞는지는 모르지만, 메그스펙트럼의 feature가 거의 40배 정도 많았다.

하지만 다시 음성으로 복원시키기 위해서 phase를 같이 싸클간에 변환해야하는지 그냥써도 되는건지, 어떻게 처리할지 몰라 조교님께 여쭈고, 조교님께서 모두의 연구소 소장님과 텐플 코리아에 문의를 한 결과, 다양한 답변을 들을 수 있엇지만 그리핀림으로 phase를 복원하는 방법이 있다는 것이 기억에 남는다.

질문 :
Cycle-gan을 응용해서 voice style transfer하는데, 입력이 unpaired한 상태에서도 성능이 잘나오도록 하는 것이 목적입니다. MCEP음성 표현을 사용해보니 성능이 잘 안나와서 stft를 사용하려고 하는데, stft의 magitude만 사용하는게 좋을지, 아니면 phase까지 같이 사용하는 것이 좋을지 궁금합니다.
현재 입력 포멧:
* MCEP :
[batch_size, time, num_features]
* stft예상포멧(magnitude만 사용한 경우) :
[batch_size, time, mag_features]

답변 :
1. phase정보를 있는 그대로 사용하는것은 거의 의미가 없구요, 여러 전처리 과정이 필요한데 저라면 일단 magnitude를 사용했을 때 성능을 점검해보겠습니다. 그리고 목소리가 이상하거나 기계잡음같은 부분은 아마도 STFT magnitude를 변조하는 것 보다, STFT magnitude를 time-series audio signal로 바꾸는 변환 과정에서 일어나는 문제가 치명적일 것 같습니다. 따라서 이 부분의 성능 개선에 집중하는게 어떨까싶습니다. 물론 전부 짐작입니다!!

2. STFT를 input feature로 사용해서 cyclegan 돌려본적이 있는데요, 당시에는 griffin-lim으로 phase를 복원했고 남녀변환정도는 꽤 잘됐었습니다. Griffin-lim의 경우 적절하지 않은 magnitude가 주어지면 잘 동작하지 않습니다. 추가로, phase의 경우 주어진magnitude에 맞는 phase패턴이 존재하기 마련입니다. Phase를 unwrap하고 주파수축으로 group delay을 계산하면 랜덤한 패턴이었던게 매그니튜드와 비슷한 패턴이 나타납니다. Tacotron2와 같은 논문에서 주어진 melspectrogram condition으로부터 적절한 wave를 생성해내는 것이나, 혹은 최근 audio source separation논문 등에서 magnitude와 noisyphase를 사용하여 clean phase를 예측하는 등의 방식들도 있습니다. 물론 가장 straight forward한 방식은 t,f representation을 사용하지 않고 time domain raw audio만을 사용하는 방식일텐데, 쉬워보이진 않습니다만 wave gan등에서 성공한 사례가 있으니 도전해볼만한 영역인 것같기도 합니다.

2. L.point Big Data Competition (17/12/11 ~ 18/01/26)

● 개요: 롯데 그룹의 14개 계열사의 1년간 고객의 상품구매 패턴 및 이용 업종 데이터를 활용해 고객의 성향 및 라이프 스타일 파악, 고객의 니즈와 취향에 맞는 상품 및 서비스를 제안. CLV와 구매스타일 클러스터링을 활용한 STP전략, 장바구니 분석을 활용한 Cross-Selling.
● 역할: 팀장으로서 분석 방향, 파생변수 도출, 데이터 전처리, 업무분담.
● 환경: R studio

● 결과 A.
- 매출액과 CLV를 통한 6가지 STP 잠재 우량고객, Old VIP, 중점관리고객
  매출액 ⇩ CLV ⇧ : 잠재 우량 고객 : 단기적읶 관점이 아닌 장기적이고 지속적인 마케팅 실행
  매출액 ⇧ CLV ⇩ : OLD VIP : 주요 수익원 현재의 마케팅 방식 유지
  매출액 ⇧ CLV ⇧: 중점 관리 고객 : 다른 고객군에 비해 마케팅에 더 많은 노력

- CLV(Customer Lifetime Value) : 공헌 마진, 활동확률, 할인율을 고려한 파생변수
  공헌 마진 : 고객이 처음 관계를 시작한 시점부터 현재까지 그 고객이 기여한 총 가치
  활동확률 : 연령대별 방문 빈도 비율 고려한 활동확률, 지수분포 고려
  할인율 : 미래에 발생하게 될 고객 가치를 현재 가치로 환산하기 위해 필요한 비율

● 결과 B.
- 당시의 파생변수:
식료품, 잡화, 남성의류, 여성의류, 스포츠, 아동, 골프, 명품, 가정/가전/가구, 식당가(푸드코트, 매장 내 카페 등), 기타(위 분류 외 나머지)로 물품으로 분류

- 새로운 파생변수: 20개
stay time, 근무 시간대 백화점 이용 비율, 주말 이용빈도 비율 , REGENDER(남편카드를 사용하는 여자 여성들이 살 수밖에 없는 물품), 명품의 기준 재정립, 서울시 상권데이터를 활용해 거주지역 식료품, 의류 생활용품 지출 대비 롯데그룹에서 이용비율 고려 충성도

유아 존재여부 , 아동 존재여부 , 건강보조품 이용정도
웰빙식품 이용정도 , 다이어트제품 이용정도 , 펫 존재여부
차량 보유여부 , 골프 이용정도, 스포츠 이용정도, 식당가 이용정도, 명품 이용정도

- 최종 세그먼트
Husband 고객층, Accessorier 고객층, Fashionista 고객층, AllThatLiving 고객층, Sporty 고객층, LUXURY 고객층, Foodie 고객층, 3rd Floor 고객층(여성의류)

- 세부 젼략
스페셜 트리트먼트 증진 및 커뮤니티 프로그램, 로열티 프로그램, 의 프로모션, 세일기간 등을 활용한 Up-Selling 방식의 마케팅, 컨텐츠

업셀링: 보다 업그레이드된 상품을 구매하도록 유도
크로스셀링: 추가적으로 더 구매하도록 유도

● 결과 C.

거래 상품을 Apriori 알고리즘으로 분석하여 돼지고기, 여행용품, 채소의 수요를 확인하여 1~2인용 / 3~4인용 / MT용으로 구성하는 Cross Selling을 제안.

- 세부 개요

당시 큰 방향으로 사람을 어떻게 어떤 변수로 세그멘테이션하고 전략을 세울지, 백화점에서 유통하는 제품을 어떻게 하면 더 잘 팔릴 수 있을지, 이러한 것들의 특정 매장에서는 어떤 고객이 주를 이루니 어떠한 방향으로 나아가야하는지 고민했었다. 매장을 찾는 것은 당시 2015년의 데이터로 추정되는데 매장 점포의 아마 51개로 기억하는데 그 숫자와 받은 데이터의 매장점포 코드가 맞지 않아 진행하지 못했다.

Lpoint 공모전에서 이렇게 데이터마이닝을 위해 고민했던 것에 흥미를 느껴 4학년 1학기에 경영학과 전공필수 과목인 마케팅원론과 소비자행동론을 수강했었다. 이러한 개념들이 STP전략 그리고 이를 수립하기위한 4p믹스 마케팅믹스로서 보다 세밀하고 정확하게 전략을 수립하는 것을 확인할 수 있었다.

- STP

· 시장세분화: 시장세분화를 위한 세분화 기준변수 파악 → 각 세분시장의 프로파일 개발
  - 시장세분화의 기준변수: 지리적변수, 인구통계적변수, 심리도식적변수, 행동적변수

· 표적시장선정: 세분시장 매력도 평가를 위한 측정정변수 개발 → 표적시장 선정
  - 세분시장의 규모와 성장률, 세분시장의 구조, 기업의 목표와 자원

· 포지셔닝: 각 표적시장별 포지셔닝을 위한 위치파악 → 각 표적시장별 마케팅 믹스 개발
  - 소비자 인식속 자사의 제품의 상대적 위치인 제품포지션을 고려해 차별과 가능요인과 차별점을 선택해 마케팅믹스를 수립하여 포지션을 전달

- 마케팅믹스**(4P, 마케팅도구들의 집합)**:

· 가격계획(price), 제품계획(product), 촉진계획(promoition), 유통계획(place)

3. Naver Data Science Competition (18/07/26 ~ 18/07/29)

● 개요: TensorFlow Speech Recognition Challenge의 Speech Commands Dataset 분류 (31개의 단어 speech 데이터 약 65,000개를 14개의 단어와 20개의 Unknown 그리고 silence로 분류)

● 역할: 데이터 전처리. Kaggle의 Kernel 모델 AlexNet으로 수정 및 비교. 발음상 유사한 2개의 단어를 위한 Metrics를 활용한 모델 분석.

● 환경: 네이버 클라우드 tensorflow ubuntu 서버, jupyter notebook, python(tensorflow)

- 배경: '음성 인터페이스는 다양한 디바이스의 스크린 중독으로부터 벗어날 수 있는 해독제이다.' 예를 들어 '카카오'라는 단어를 말하면 응답하도록 설정하였을 때, 지인과 통화 도중 여러 단어를 카카오 미니 자신을 부르는 것으로 착각하곤 했었다.

- 전처리: 모두 1초 이내의 데이터로 0.003초가량을 추가하는 방법과 librosa library를 활용해 늘리는 방법. 활용.
  음성데이터를 클래스 별로 FFT한 후 PCA를 통해 2차원으로 나타낸 뒤 3 sigma 수준 이상의 점을 제거
  PCA: 분산이 최대가 되도록 하는 축들을 (다 모으면 주성분임 이것이) SVD로 찾는다.
  Singular value decomposition (특이값분해)
- 결과: 0.003초 추가 방법/ 늘리는 방안, spectogram / log-mel-spec / 2개의 모델 조합으로 84 ~ 93까지 다양
  모델 파라미터 커널: 11만 / 2천만
  알렉스 넷에서 conv할 때 padding = same으로 한 것 정보손실로 -> 93% 가 최고
  커널 모델: 축소된 VGG와 비슷 (3x3 커널, stride 1)

- metrics!: 또 다른 결과로 "go-no", "Three-tree" , "on-one"와 같은 발음상의 유사한 단어를 놓고,
  단어를 인식했지만 반응을 하지 않아도 되는 단어 Recall 로 놓고 모델 평가
  단어를 인식하고 반응해야하는 것이 좋은 단어 ;; 다시봐도 애매하네
  RECALL: VIP를 분류할 때, TP / TP + FN "카카오"
  실제 이거로 말하면 나온건 다 인식 or 반응 x
  PRECISION: SPAM을 분류할 때 , TP / TP + FP
  이거라고 예상 한거는 다 맞는 게 좋음

| | | Actual Value | |
|---|---|---|---|
| | | True | False |
| Estimated Value | Positive | True Positive | False Positive |
| | Negative | False Negative | True Negative |

- AlexNet 설명:

1. Local Response Normalization: 일종의 Regularization 테크닉으로 CNN에서 output으로 Convolutional Feature Map이 나오면 그 맵에서 일정부분만 높게 Activation되게 하는 것. 사람의 뇌가 실제로 이렇게 작동함

2. Data augmentation과:
   a. flip하는 것 / b. crop하는 것.(224 smaller patch활용) 이러한 스킬은 Label이 보존되게 바꿔야함
   c. Color Variation: 그냥 노이즈를 섞는 것이 아니라 해당 라벨의 RGB 변화량에 비례하게 RGB값을 더함
3. Drop out: output으로 레이어가 나오면 노드 중 일정 퍼센트를 0으로 만들어 주는 것. 여기서는 그냥 0.5를 곱했는데 이 방식은 이 것이 처음이자 마지막;;

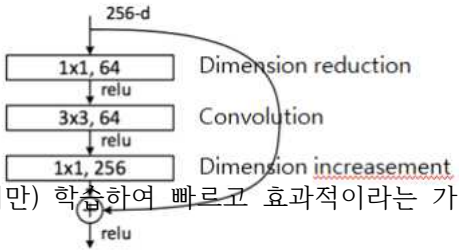
- 다른 Modern CNN설명:

VGG: 3 x 3 feature map stride 1로 다 통일해서 간단하게 했다는 것. layer 19와 16을 많이 활용.

GoogLeNet:
  Inception 모듈에서 다양하게 convolution하고 그전에, 1x1 Conv으로 채널을 줄여 차원축소 효과를 보아
  VGG보다 더 깊지만 파라미터를 줄임
  * 깊은 것이 항상 좋으냐
  아니다. 그때 그때 다르다. Vanishing , Exploding 때문에
  그러나 Better initialization methods / batch normalization / ReLU 이걸하면 위문제는 없을 수 있다.
  오버피팅은?

오버피팅 Training acc는 감소하는데 test acc가 증가하는거 -> 얼스탑으로 해결
Degradation: acc 둘 다 잘나와오는데 성능이 잘안나오는거  여기선 시파(50층) 이미지넷 30층쯤 발생

ResNet: 입력이 들어오면 입력에 출력을 더함. 입출력에서 디멘션이 같아야하는 것만이 제약
resiual building block을 만들고, 중간에 건너 뛴 있는 레이어는
기존 입력과 (desirable)원하던 입력과의 차이를 학습할 것이다.
degradation을 100층으로 밀음. 1000단이면 다시 디그라데이션




이는 중간에 있는 레이어가 x라는 입력과 원하는(desirable) 입력과 차이(나머지만) 학습하여 빠르고 효과적이라는 가정으로 출발했으며, 실험결과 실제로 빠르고 효과적인 모델이 만들었다.

실제 음성 인식: ASR(automatic speech recognition), STT(Speech 2 text)
GMM-HMM,  LSTM, CTC(Connectionist Temporal Classification )-RNN

6. 서비스경영 팀 프로젝트 (18/10/18 ~ 18/12/06)

● 개요: 롯데백화점, 챗봇 샬롯 서비스 분석

● 역할: 샬롯의 추천시스템 분석, “Deep Learning based Recommender System: A Survey and New Perspectives”을 통한 개선방안 분석. QA 시스템을 활용한 문제점 개선방안 도출

샬롯을 분석한 결과 크게 2가지 문제점과 개선방안을 생각했습니다. 실질적인 코드로 분석한 것이 아니라 개념적으로 접근해 문제를 분석하고 개선방안을 제시한 것입니다.

샬롯은 개인맞춤, 상황맞춤, 스타일추천, 사진판별, 유행정보를 활용한 추천, 선물 추천 등 다양한 추천시스템이 있었지만 이 추천 상품의 설명력이 부족했습니다. 추천해주는 상품의 구매평 X, 평점정보 X, Q&A X 등등의 문제 콜드스타트일 등의 문제 또는 알고리즘 문제라고 생각했습니다.

그리고 샬롯의 추천시스템 알고리즘을 조사했습니다. 이를 통해 IBM watson의 알고리즘인 pyspark의 matrixfactorization 모델과 ALS알고리즘을 활용한다는 것을 확인했습니다.

MF는 선형대수의 SVD와 관련이 있으며, user item의 joint latent factor space에 매핑해 모델링하는 것으로 까지만 알고, 이를 통해나온 손실함수를 학습한는 방법으로 SGD와 ALS가 있는데 ALS가 SGD보다 느리고 복잡하지만, 병렬처리, 그리고 암시적(Implict) 데이터를 중심으로 하는 시스템에 유리한 것까지만 안다.

그리고 이것이 머신러닝 기반의 모델이므로 이를 Deep Learning based Recommender System의 서베이 논문을 읽고, 딥러닝 기반으로 바꾸면 어떠겠냐 제안. 서베이 논문은 전체적인 추천시스템의 개요, 딥러닝 모델들을 뉴럴 빌딩 블락이나 데이터 도메인등에 따라 구분하고, 딥러닝 기반이 왜 좋은지 등의 긴 논문이었습니다.

이어서 샬롯에서 나온 문제점으로 문화센터 지점 및 몰링 문화 서비스 안내 -> 홈페이지로 유도 -> 지점찾고 프로그램 다시 다 찾아야하는 불편함 ->  QA 시스템으로 문서화하면 이를 해결할 수 있지 않을까 제안 했습니다.

- 딥러닝추시 요약.

추천시스템은 사용자 기호(User Preferences), 상품 특성(Item Features), 사용자-상품 과거의 거래내역, 추가정보(시계열 특성, 공간적 특성(Spatial ex)POI))을 활용하여 추천리스트 생성하는 것이다. 또한 사용자와 상품의 interaction(상호작용)과 명시적/암묵적 피드백을 활용하는 협업필터링(CF)과 유저간의 상품간의 보조정보의 비교로 추천하는 콘텐츠 기반 추천시스템. 그리고 이를 합친 하이브리드 추천 시스템으로 나뉜다.

Content-based는 아이템이나 유저를 분석하여 비슷한 아이템을 추천한다. 아이템과 유저간의 액션을 분석하는 것이 아니라 컨텐츠 자체를 분석하기 때문에 많은 양의 유저의 액션을 요구하지 않는다는 것이 장점. 즉 CF의 문제점인 cold start가 없다. 이에반해 협업필터링은 여러 사람들의 평가정보를 활용하고, 다양한 범위의 추천이 가능하지만 많은 데이터가 필요하다.

샬롯의 추천시스템으로 유추되는 Matrix Factorization은 BPR(Bayesian Personalized Ranking), CML(Collaborative Metric Learning)과 함께 머신러닝 기반의 Standard 모델이다.

A. Nonlinear Transformation:

ReLu, Sigmoid와 같은 비선형 활성화 함수로 비선형 모델링이 가능하다. 샬롯에서 사용하는 MF은 선형모형으로 선형가정을 사용하여 지나치게 단순해 모델의 표현력을 제한한다. 하지만 비선형성을  활용한 딥러닝은 복잡한 사용자와 상품간의 상호 패턴을 더욱 잘 잡아낼 수 있다.

B. Representation Learning (Feature Learning: RBM, PCA, 오토인코더 등등으로 함)

노동적인 Feature engineering을 딥러닝은 비지도 지도학습을 통해 자동적으로 할 수 있게 하는 것과 text, images, audio and even video 의 정보들도 사용가능하다.

C. Sequence Modeling

추천시스템의 모델링에서 사용자와 상품의 시계열 특성은 굉장히 중요한데 RNN과 CNN의 성능으로 다음 장바구니 예

측(Next-Item/Basket Prediction), 세션기반 추천(Session Based Recommendation)과 같은 모델링이 가능하다.
(세션: 단기 기록, 소규모 기록)

D. Flexibility:
뉴럴구조를 결합하고 교체하기 쉬우며 이로 인해 다양한 특성과 요인을 동시에 포착하는 하이브리드 모델을 구축할 수 있다. 딥러닝에는 텐서플로우, 케라스, 카페 파이 토치등 다양한 딥러닝 프레임워크가 있다.

넷플릭스는 고객의 80% 영화 상영은 추천목록으로부터, 유튜브의 60% 클릭수는 추천페이지에서 이루어 졌다고 한다. 또한 구글 플레이스토어는 MLP 아키텍처를 활용한 Wide Deep Model을 Yahoo 뉴스는 RNN을 활용한 추천시스템을 활용하고 있다.

- QA넷 요약

기존의 BiDAF의 모델을 개선한 것으로 Recurrent를 제거하고 Conv와 self attention을 Encoder의 Building Block으로 사용. Encoder는 Q와 Context 각각 인코딩
-> 결과적으로: 빠르다 -> 학습을 많이하면 정확성 업
-> Translation(영불영)을 활용하여 Paraphrase 하여 Data Augmentation 하면 좋다.

Reading Comprehension Model의 대부분의 Standard Model이다.

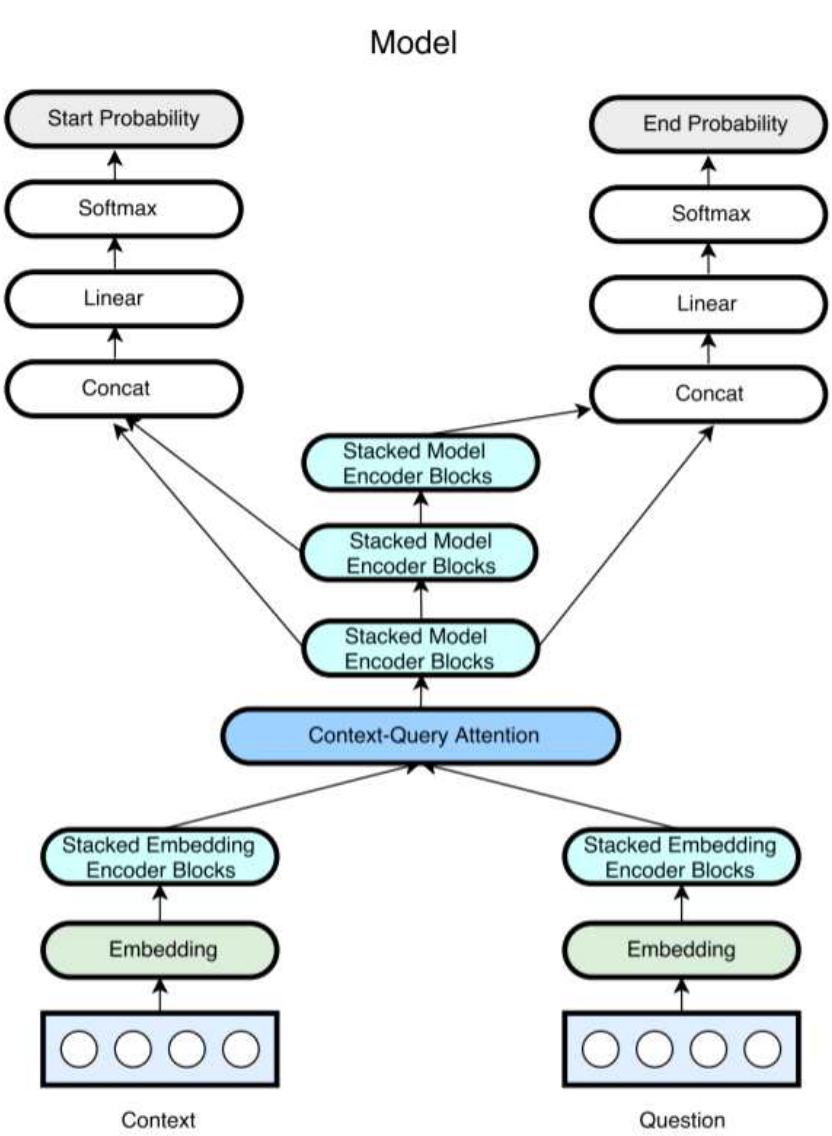
A. Input Embedding Layer
1) word embedding:
2) character embedding

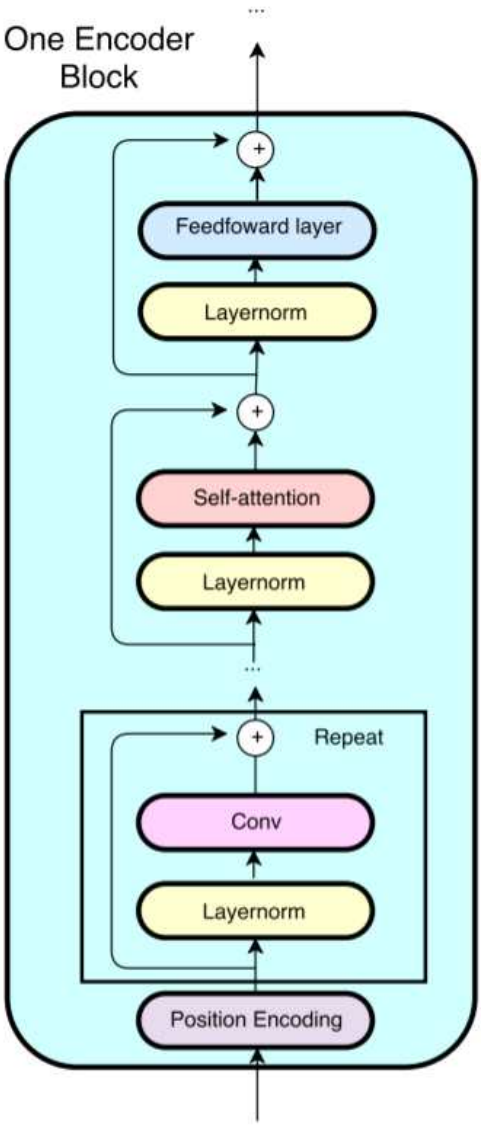
B. Embedding Encoder Layer.
[convolution-layer × # + self-attention-layer + feed-forward-layer] 이 쌓인 의 형태
“Attention is all you need”의 multi-head attention mechanism 그리고 이것이
For the self-attention-layer, we adopt the multi-head attention mechanism defined in which, for each position in the input, called the query, computes a weighted sum of all positions, or keys, in the input based on the similarity between the query and key as measured by the dot product.을 한다.

C. Context-Query Attention Layer
computer the similarities between each pair of context and query words, rendering a similarity matrix S 이를 또 많이 계산하는데 사용한다.

D. Model Encoder Layer. : BiDAF와 거의 비슷

E. Output layer.
- This layer is task-specific. (task에 따라 다르다)
- 객관식 문제에서 고르기와 같은 문제같은 other comprehension tasks로 customize 가능하다,


Model
Start Probability
Softmax
Linear
Concat
End Probability
Softmax
Linear
Concat
Stacked Model Encoder Blocks
Stacked Model Encoder Blocks
Stacked Model Encoder Blocks
Context-Query Attention
Stacked Embedding Encoder Blocks
Embedding
Stacked Embedding Encoder Blocks
Embedding
Context
Question


One Encoder Block
Feedfoward layer
Layernorm
Self-attention
Layernorm
Repeat
Conv
Layernorm
Position Encoding

4. 산업공학종합설계 팀프로젝트 (18/09/17 ~ 18/12/03)

● 개요: CNN을 활용한 폐타이어 역물류 시스템 개선 및 아레나를 통한 시뮬레이션 비교.

● 역할: 이미지넷으로부터 데이터 수집 전처리, ResNet과 Inception V2 모델 분석, Transfer Learning을 활용한 분류

● 환경: Google Colaboratory, Pytorch

- 배경: 폐타이어 야적장에서 폐타이어가 너무 쌓이는 바람에 국가에서 환경적으로 오염된 구역으로 지정하였다. 이를 해결해보자.

- 전처리:

받은 데이터는 전처리 다음과 같이 한다. 받은 데이터를 torchvision.transforms.RandomResizedCrop(224), torchvision.transforms.RandomHorizontalFlip()과 같은 방법을 활용해 데이터를 늘린다. (Data Augmentation) 이렇게 이미지를 증강 시키는 면 오른 쪽 그림과 같이 데이터가 늘어남을 확인할 수 있다. 이는 과적합(오버피팅)을 방지하는 방법인 Regularization의 하나로 ALEXNET에서 처음 사용되었었다.

- Inception

기존 CNN 모델은 네트워크가 깊어지면 깊어질수록 성능은 좋아지지만 Gradient Vanishing과 같은 문제들이 발생한다. 따라서 한번의 Convolution Layer에서 1x1, 3x3, 5x5 Filter를 활용하여 연산을 각각 수행하고, 1x1 Convolution을 사용하여 이미지의 차원을 축소시켜 parameter 개수를 효율적으로 줄일 수 있는 모델이다.

Inception 모듈은 왼쪽 그림과 같이 생겼으며, 이는 Network in Network라고도 불린다. 차원 축소는 오른 쪽과 같은 형태로 진행된다. 오른쪽 그림에서 왼쪽이 차원축소를 안하고 Convolution을 진행할 때 이며, 이때 사용된 파라미터 개수는 3 * 3 * 18 * 30 = 약 5000 개임을 확인할 수 있다. 이에 비해 1x1 convolution을 활용한 오른쪽 그림은 (1*1*18 +3*3*5) *30 = 약 1300 개로 모델이 깊지만 파라미터는 줄어드는 Dimension Reduction의 효과를 보았다.

- transfer learning

위에 말한 Transfer learning을 하기위해선 pytorch.models.resnet18 을 활용해 다운받고, 그 아래 줄 코드를 활용해 파라미터를 Freeze하고 마지막 단에 우리가 분류해야할 이미지 클래스(2개)를 연결하여 마지막 층은 학습하게 준비한다.

Transfer learning은 사람들이 각자 자신이 가진 데이터네에서 이미지를 분류하거나 학습을 하기 위해 데이터가 적은 경우, 활용하는 것이다. 우리가 사용한 Resnet, Inception 이외에 ALEXNET, DenseNet 등 다양한 모델과 학습된 파라미터들을 제공한다. Transfer Learning의 방법은 구현된 모델의 학습파라미터를 불러와 데이터 사이즈에 맞게 마지막 층만 학습하는 것이며, 이 때 Learning rate는 기존의 Learning rate의 0.1정도로만 한다.

- 결과

평균적으로 ResNet이 빠르고 더 정확한 모델임을 실험을 통해 확인하며 Residual Building Block의 성능을 체감할 수 있었다. 평균 분류시간은 200장의 이미지를 분류하는데 걸리는 시간임으로 하나의 타이어를 분류하는데 시간은 더욱 짧을 것으로 예상되며 아래 그림과 같이 학습한 데이터의 이미지에서 타이어의 위치가 굉장히 뒤죽박죽임에도 불구하고 잘 분류하는 것을 보아 우리가 실제로 적용할 폐타이어나 일반 타이어 분류는 컨베이어 벨트라는 완벽한 제약조건에서 수행될 때 분류 성능 또한 더욱 높아질 것으로 예상.

| 모델 | 평균학습시간 | 평균분류시간 | Acc 평균 | Best Acc. |
|---|---|---|---|---|
| Inception V2 | 6m 54s | 5s | 90.45 % | 94% |
| ResNet | 4m 20s | 3s | 93.55 % | 94.5% |

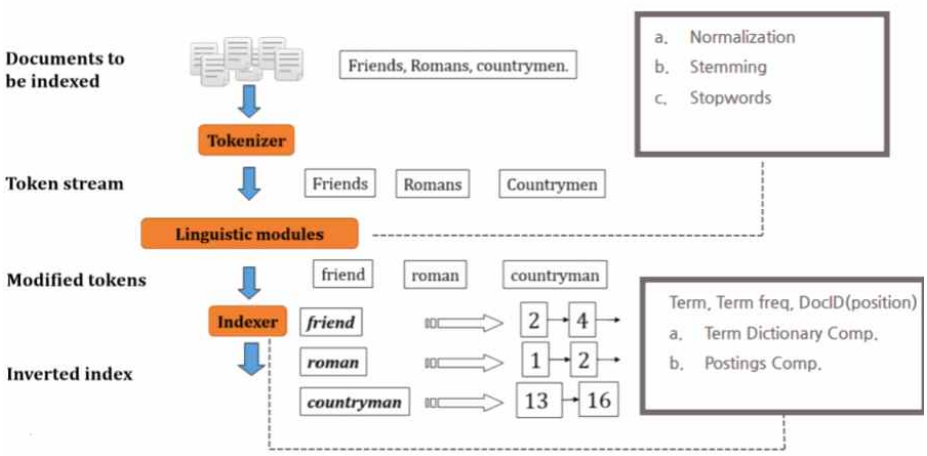
5. 소프트웨어실습 프로젝트 (17/10/20 ~ 17/12/01)

● 개요: 소프트웨어 실습 수업의 개인 텀프로젝트. 정보검색론을 수강하며 배운 정보시스템 구현.

● 역할: Gutenberg의 무료 도서데이터의 단어에서 Tokenizer, Linguistic modules, Indexer를 거쳐 AVL tree의 형태 Inverted Index를 구축. 단어를 검색 시 단어의 도서, 도서 내의 해당 부분, 그리고 요약된 도서 내용을 볼 수 있도록 구현.

● 환경: Pycharm, Python

- 프로젝트 개요: 소프트웨어실습 수업의 개인 텀프로젝트로 당시 정보검색론을 수강하면서 배운 정보검색시스템을 구현했습니다. Gutenberg의 무료 도서데이터의 단어에서 Tokenizer, Linguistic modules, Indexer를 거쳐 AVL tree의 형태 Inverted Index를 구축했습니다. 그리고 단어를 검색하면 단어의 도서, 도서 내의 해당 부분, 그리고 요약된 도서 내용을 볼 수 있도록 구현했습니다.




- 자연어처리 개요:
  Inverted Index를 구축하기위해 먼저 토큰화하고
- Tokenization: Cut character sequence into word tokens / Deal with “John’s”, a state-of-the-art solution
  Linguistic모듈에서 N, S, S를 거쳐
- Normalization: Map text and query term to same form / You want U.S.A. and USA to match
- Stemming: We may wish different forms of a root to match / authorize, authorization
- Stop words(불용어): We may omit(생략하다) very common words (or not) / the, a, to, of

  AVL Tree에 저장하게 하였다. 노드에 key, (tf_list or tf-idf_list), (location_list)
  사실 AVL Tree가 아닌 트리에 리프노드에 리스트를 추가하는 형식으로 저장되어 검색도 어느 정도 빠르고
  저장 공간 효율을 높이는 형태로 저장

a. tf-idf (tf: doc d에서 term t가 나온 횟수, idf $idf_t = \log_{10}(N/df_t)$ )
   Doc에서 Term이 나타난 횟수, Collection에서의 term의 희소성에 따라 증가

- AVL TREE: Balanced binary tree로 검색, 삽입, 삭제, 높이: log(n) 회전을 통해 높이 재구성

- 요약: 가장 간단한 요약법으로 문서 내에서 단어를 토큰화, 정규화한 후 가장 빈도수가 높은 단어가 포함된 문장을 출력한다.

7. Big Contest 퓨처스리그 (17/08/07 ~ 17/09/29)

● 개요: 약 5년간의 영화 데이터를 활용하여 새로 개봉하는 3편의 2주 동안의 누적관객 수를 DNN을 활용해 예측.

● 역할: 데이터 정제, 추가 변수, 파생 변수 조사

● 환경: Pycharm, Python, Excel

- 변수들: **감독파워 (해당 영화 전작 3편의 평균 누적 관객수), 배우파워, 공휴일 수(상영일 수 동안 해당된 공휴일 + 문화의 날 수), 찜하기, 개봉전 평점, 평점 참여인원, 보고 싶어요, 글쎄요, 예고편 수, 블로그, 뉴스, 날짜 겹치는 영화 수, 원작도서 유무, 속편**

- 아쉬운 점 & 개선점:

영화 개봉 전의 트위터 반응 자연어 처리를 통한 예측

+ Trailer와 같은 영상을 통한 예측 -> 멀티모달을 통한 예측?