이력서: Github io link

경력기술서: Career description link

포트폴리오: Portfolio link

자기소개서: Cover letter link

[프로젝트 목차]

기록 순서는 포트폴리오와 동일하게, 진행했던 시간의 역순서로 기입했습니다.

포트폴리오에 기재된 프로젝트들과 동일하며, 포트폴리오의 내용을 자세히 정리했습니다.

[1] Undefined (page no.2-3)

[1-1] Undefined	개발 팀	FAQ Chatbot 개발 (3 개월)
[1-2] Undefined	개발 팀	대회 규칙 추천 시스템 개발 (2 개월)
[1-3] Undefined	개발 팀	경기 결과 자동 기록 시스템 개발 (1 개월)

[2] 한양대학교 대학원 (page no.3-6)

[2-1] 한양대학교 대학원	ML 시스템 연구실	Graph Data Embedding (2 년 6 개월)
[2-2] 한양대학교 대학원	ML 시스템 연구실	DNN Model Quantization - 1 (1 년 8 개월)
[2-3] 한양대학교 대학원	ML 시스템 연구실	DNN Model Quantization - 2 (3 개월)
[2-4] 한양대학교 대학원	ML 시스템 연구실	Artificial Intelligence Assistant (2 개월)

[3] Kakao (page no.6-10)

[3-1] Kakao	추천 팀	(자동차 동영상 & 만화) 추천 알고리즘 개발 (2 개월)

[4] 한빛소프트 (page no.11)

[4-1] 한빛소프트	AR 팀 인공지능파트	Chatbot 개발 (2 개월)
[4-2] 한빛소프트	AR 팀 인공지능파트	Speech Synthesis 모듈 개발 (4 개월)

[1] Undefined |

Undefined는 iScrim이라는 플랫폼을 개발하는 회사이며, iScrim은 eSports 대회를 쉽게 개최할 수 있도록 돕는 플랫폼입니다. 앱을 이용한다면 복잡한 대회 개최 및 운영 절차들을 간편하게 수행할 수 있습니다. iScrim 플랫폼 상에서 인공지능이 적용된 모듈은 크게 세 가지입니다.

1. 대회 운영자 및 참가자가 대회 관련 정보를 쉽게 질의할 수 있는 FAQ Chatbot 모듈
2. 대회 운영자가 대회를 개최할 때 대회의 세부 규정을 고민하는 수고를 덜어 줄 수 있는 Competition Rule Recommendation 모듈
3. 대회 참가자가 경기 결과 snapshot을 iScrim에 제출할 시, 자동으로 username, rank 등을 포함한 매치 정보를 iScrim 시스템에 기록해주는 OCR 모듈

세 가지 모듈들 각각은 ECS를 이용하여 배포되며, EC2에 배포되는 AI Server를 통해 iScrim의 메인 서버와 교류합니다.

[1-1] Undefined 개발 팀 FAQ Chatbot 개발 (3 개월)

첫 번째인 FAQ Chatbot은 RASA 라는 chatbot framework를 이용했습니다. 저는 사내 데이터셋을 수집/정제 했으며, chatbot의 전체 파이프라인을 설계하고, 모델들을 학습/배포했습니다. 파이프라인 내에서의 주요 ML 모델들은 Multi-lingual BERT와 Starspace입니다. Starspace는 모든 것(*)을 embedding 한다는 개념하에 개발된 기법으로, sentence를 token화 하여 embedding하는 것 또한 가능합니다. iScrim에서는 FAQ 질문을 token화 하여 답변과 매칭하는 형태로 embedding을 학습했습니다.

이후 대회마다 다른 세부 정보를 DynamoDB에서 query하고, 답변에 그 내용을 포함시키는 작업을 수행했습니다. 역시 Starspace를 이용했습니다. 최종적으로 iScrim의 주요 고객층은 해외에 있으므로, 영어와 한글 모델을 각각 개발하여 배포했습니다. 배포는 EC2(AI Server)와 ECS(Chatbot Server)를 이용했습니다. 유지/보수를 위해 사용자의 피드백을 만족/불만족으로 받았고, 불만족의 경우 질문/답변/모델명/confidence 등의 값들을 DB에 저장 및 확인했습니다.

[1-2] Undefined 개발 팀 대회 규칙 추천 시스템 개발 (2 개월)

iScrim의 두 번째 AI 모듈인 Competition Rule Recommendation(CRR) 모듈입니다. 대회 규정들로는 라운드 수/팀 최대 인원/후보 선수 인원/킬 당 점수/매치 순위 점수 등으로 매우 다양하며, 게임마다 다른 형태를 갖고 있습니다. 물론 대회 운영자가 대회 종목(게임)에 대해 세부적으로 파악하고 있다면 대회 개최가 쉽게 이루어질 수 있지만, 그렇지 못하여 대회 개최에 어려움을 겪는 경우들이 많습니다.

CRR은 most popular value를 이용하여 대회 규칙들을 설정해 줍니다. 그리고 사용자가 자신만의 요구사항에 맞게 대회 규칙들 중 일부를 수정했을 때, 수정된 값들을 기반으로 변화가 필요한 다른 규칙들을 예측하여 재설정해 줍니다. 예측 모델로는 Alternative Least Square(ALS)를 이용했습니다. ALS는 Matrix Factorization의 optimization 기술로, user가 구매한 item들을 기반으로, 구매하지 않은 다른 item들의 구매 여부를 예측할 수 있습니다. 이 때 binary 형태의 '구매'가 아닌 '평점'(e.g., 영화 평점)과 같은 형태로도 수행할 수 있으며, CRR은 대회 규정 값들을 평점과 같은 numerical feature로 취급했습니다. 사용자가 설정한 규정 값들을 기반으로, '사용자1의 대회'를 user로 취급하여 '사용자1의 대회'의 latent factor를 학습하고, 각 규정들을 item들로 취급하여 규정의 latent factor와 내적하여 규정 값을 예측했습니다.

모든 규정들 간에 관계성이 있지는 않으며, 오히려 예측에 방해가 되는 규정들이 있습니다. 따라서 모델 학습 이전에 Pearson Correlation과 Regression 모델을 통해 두 규칙 간의 상관관계, 혹은 여러 규칙들이 하나의 규칙을 예측할 수 있는 정도 등을 분석하여 ALS 모델이 예측하는 데에 사용할 규칙들을 선별했습니다.

추가적으로, ALS의 학습 속도를 최적화했습니다. ALS는 대회를 생성할 때 마다 학습이 이루어져야 합니다. 사용자가 설정한 일부 규정(item)들을 이용하여 대회(user)의 latent factor를 학습해야 하고, 그 학습된 latent factor를 다른 규정들의 latent factor와 내적하여 값을 예측해야 합니다. 즉, 규정들의 latent factor는 배포 이전 시점에 이미 학습된 상태로 업데이

트되면 안되기 때문에, ALS 학습 과정에서 item들의 latent factor 업데이트에 사용되는 요소들을 모두 제거하여 CRR 모듈의 latency/throughput을 약 3배 단축/증가 시켰습니다.

[1-3] Undefined 개발 팀 경기 결과 자동 기록 시스템 개발 (1 개월)

iScrim의 세 번째 AI 모듈인 Match Result Recorder 입니다. MRR은 OCR 모듈을 포함한 시스템으로, 경기 및 매치 종료 후 참여 선수가 매치 종료 화면을 스크린샷을 iScrim 플랫폼에 제출하면 자동으로 스크린샷 내의 username, 순위 등을 기록해 줍니다. 자사 모델을 선택할 때 가장 주요하게 생각한 것은 multilingual의 가능 여부입니다. 한국 게임일지라도 username과 기타 속성들에 한글/영어/일본어 등의 여러 언어들이 혼합되어 존재하기 때문입니다.

iScrim의 세 가지 AI 모듈들 중 학습에 사용할 데이터셋이 가장 부족한 모듈이며, 자사 모델만으로 서비스에 이용되기는 가장 어렵다고 판단되는 모델입니다. 따라서 성능이 가장 우수하다고 판단되는 cloud 서비스인 Google Vision API를 이용하여 자사 모델을 보조하는 pipeline을 개발했습니다. Username, 순위 와 같은 필수적인 keyword들의 존재 여부를 검사하고, 순위 등수와 같은 숫자 값이 numerical data로 잘 예측되었는가를 판독했을 때 좋은 결과가 아니라고 판단되면 cloud를 이용하여 예측 결과를 도출하도록 pipelining을 했습니다.

[2-1] 한양대학교 대학원 ML 시스템 연구실 Graph Data Embedding (2 년 6 개월)

UNIST와 고려대학교의 연구실들과 함께 진행한 내용으로, 저희 연구실에서는 약물 반응성 예측에 활용하게 될 네트워크(단백질-세포-약물) 데이터에 대한 embedding representation 학습을, 다른 두 연구실에서는 약물 반응성 예측을 위한 분류 모델 학습과 데이터 분석을 담당했습니다. 아래 내용에 대한 논문은 Briefings in Bioinformatics(BIB) 저널에 2022년 12월, publish 됐습니다.

사용한 프로그래밍 언어는 Python이며, 기술로는 Node2Vec을 사용했습니다. Node2Vec은 Word2Vec 기술에 natural language 대신 graph의 node를 input으로 사용하는 기법이며, node의 representation vector를 학습하는 것이 목표입니다. 저희 프로젝트의 경우 단백질/세포/약물 세 가지 노드 타입이 존재합니다. 프로젝트의 최종 목표는 암세포와 항암 약물의 반응성을 예측하는 것으로, 단백질은 세포의 특성을 표현하기 위해 함께 사용됩니다.

저희 프로젝트에서 사용 중인 데이터셋은 1000개 정도의 세포와 300개 미만의 약물을 보유하고 있으며, 그에 반해 단백질의 개수는 약 2만 개 입니다. 즉, representation의 학습은 대체로 단백질에 의해 이루어지며, 상대적으로 수가 매우 적은 세포와 약물의 경우 학습에서의 영향력이 매우 적었습니다. 그리고 이로 인해 학습된 vector들은 세포와 약물의 관계를 잘 파악해 내지 못했습니다. 물론 단백질의 개수를 강제로 줄일 수 있겠으나, 이는 적은 데이터셋들이 갖는 단점들을 강화시킬 우려가 있었고, 최대한 많은 데이터들을 활용해 보고자 다른 방법을 연구했습니다.

고민하던 중 Word2Vec의 심화 기법 중 하나인 Negative Sampling(이하 NS)을 활용하여 이 문제를 해결해 보고자 했습니다. Negative Sampling은 Word2Vec의 computation cost를 감소시키기 위해 제안되는 기법입니다. 조금 더 구체적으로 설명 드리자면, NS는 Word2Vec의 loss를 계산할 때 사용되는 두 가지 계산인 [center word – neighbor word], [center word – all the other words]에서 후자의 항목을 [center word – selected negative words]로 approximate하여 계산량을 감소시키는 기법입니다. 물론 일반적인 NS를 사용할 경우, 후자의 항목인 'selected negative words' 역시 대체로 단백질만 선택되게 되며, 단백질이 학습의 주요 요소가 되는 현상은 여전히 발생했을 것입니다. 하지만 이를 변경하여, center word가 세포 혹은 약물일 경우, 선택되는 negative node가 resistant 한 약물 혹은 세포로 설정하도록 하여 embedding representation이 약물 반응성 예측에 특화될 수 있도록 했습니다. 그 결과 학습 데이터셋에 존재하는 95%의 약물에 대한 반응성 검사 성능이 상승했으며, 90%에 해당하는 테스트 데이터셋의 약물들의 성능이 향상됐습니다

[2-2] 한양대학교 대학원 ML 시스템 연구실 DNN Model Quantization - 1 (1 년 8 개월)

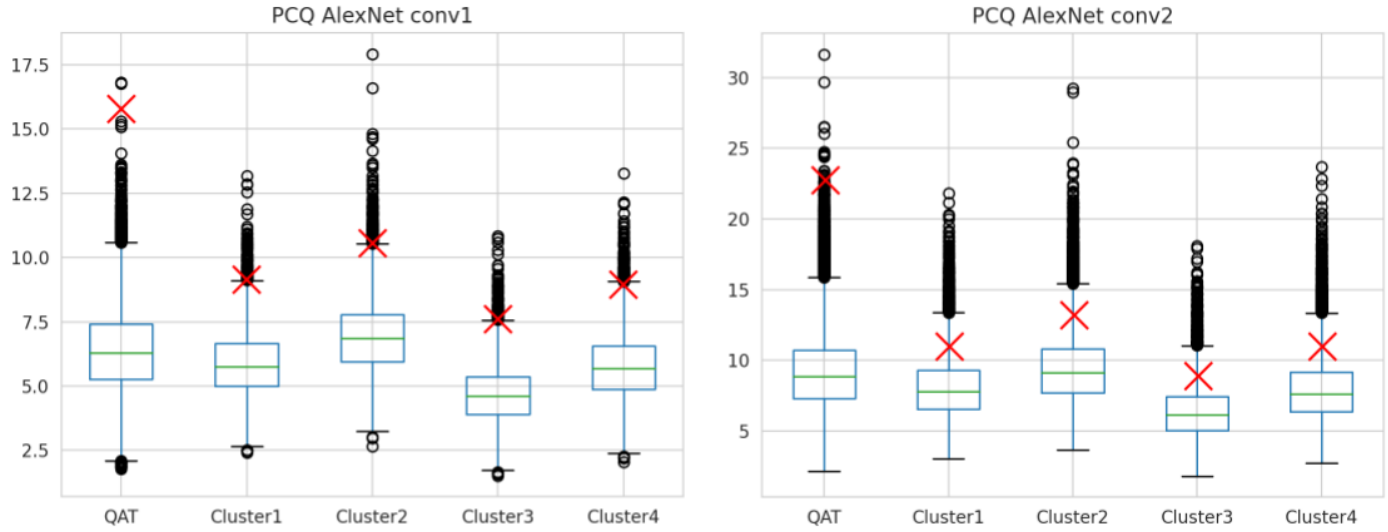
먼저 Quantization에 대해 간략히 설명 드리겠습니다. 일반적으로 딥러닝 모델들은 FP32(실수) 자료형을 이용하여 연산을 하도록 학습이 됩니다. Quantization은 이러한 실수 모델들이 더 작은 실수 자료형(FP16), 혹은 정수 자료형(INT8, INT4 등)으로 연산/저장 하도록 축소시킵니다. 이는 주로 실수 연산이 불가능한 하드웨어, 혹은 저장 공간이 제약되는 모바일에서 사용할 모델들을 축소시키는 데에 사용됩니다.

Quantization에 다양한 기법들이 있지만, 저는 가장 많이 사용되는 기술 중 하나인 Google TensorFlow의 Quantization Aware Training(이하 QAT)의 한계 극복에 대한 연구를 진행했습니다. QAT는 INT8 자료형으로의 quantization 만을 지원하며, INT8 보다 작은 자료형을 사용할 경우 모델의 성능이 크게 감소하는 단점이 있습니다. ImageNet dataset으로 학습한 ResNet50, DenseNet121 모델들을 QAT 기법으로 INT4 quantization을 진행했을 때, accuracy가 각각 26%, 22% 하락했습니다.

저는 위 한계를 극복하기 위해, QAT에서 quantization을 위해 학습하는 특수한 parameter들을 통계적인 관점에서 관찰했습니다. Quantization parameter들은 학습 데이터로부터 데이터셋의 분포를 파악하며 학습되지만, Google의 방법으로는 outlier들을 구분해 내지 못함을 파악했습니다. 따라서 이를 개선할 수 있는 기법 두 가지를 제안했고, 큰 성능 향상을 이끌어 낼 수 있었습니다. 위에서 언급한 두 모델들에 대해 저희 기법으로 INT4 quantization을 진행할 경우, QAT 대비 10.7%, 8.3%의 성능 향상이 있었습니다.

두 가지 contribution들이 각각 무엇인지 설명드리기 전에, 먼저 QAT 기법을 사용했을 때 accuracy drop이 발생하는 원인을 말씀드리겠습니다. Quantization을 위해서는 fine-tuning을 진행하며 quantization parameter를 학습 및 생성해야 합니다. Quantization parameter는 모든 layer 별로 생성되어야 하며, layer 별 output matrix가 갖는 실수 값들의 범위를 필요로 합니다. Output matrix 별로 실수 구간의 값들인 [실수-min, 실수-max] 를 원하는 타겟 정수 값으로, UINT8의 경우 [0, 255] 로 표현하게 됩니다. Min/max 값들은 input data에 따라 크게 달라지므로 fine-tuning을 하며 계속해서 min-value의 평균값과 max-value의 평균값을 갱신합니다. INT8로 양자화를 진행할 경우 256개라는 적지 않은 값들로 실수값들을 표현하게 되지만, INT4의 경우 16개로 매우 제한된 개수만을 사용할 수 있고, 이는 매우 큰 정보 손실을 야기합니다. 그런데 QAT 기법을 사용할 경우 각 matrix 별로 outlier들을 포함해서 min/max-value들을 평균하게 되어 실수 구간을 매우 크게 규정하게 됩니다. 이 방법은 정수 자료형이 충분히 클 경우 최대한 다양한 값들을 표현할 수 있다는 장점이 있지만, 자료형 크기가 작을 경우에는 대부분의 실수 값들이 동일하게 표현 되면서 성능 하락이 커지게 됩니다.

제안하는 기법의 contribution 두 가지 모두 quantization parameter 학습에서의 parameter quality를 향상시키는 것에 관련돼 있습니다. 첫 번째 contribution으로, 이전 단락에서의 QAT의 한계를 극복하기 위해 실수 구간(min/max-values) 측정할 때 최대한 outlier들을 제거하도록 개선시켰습니다. 두 번째 contribution으로, clustering을 통해 input data를 유형을 별로 나눈 후 cluster 별로 quantization parameter를 따로 학습했습니다. 이와 같은 기법을 생각한 이유는 input data에 따라 output의 min/max-range가 크게 달라지기 때문이며, 실수 구간의 크기가 작은 데이터들의 경우 더 작은 실수 구간으로 생성한 quantization parameter들을 사용하는 것이 accuracy 향상에 유리하기 때문입니다. Input data level에서 clustering을 수행한 이유는 여러 선형 변환들의 집합인 DNN(Lipschitz continuous function)의 output들은 input data에 dependency가 있으며, 따라서 input level 에서 intermediate output들의 대략적인 min/max 값을 예상할 수 있을 것으로 생각했습니다. Clustering 기법의 경우 K-means algorithm을 사용했으며, K-means model의 경우 1차적인 clustering을 진행하고(sub-cluster), 이후 target DNN에 보다 적합한 형태인 final-cluster로 merge 하는 방식을 제안했습니다. 이와 같이 sub2final 형태로 나누어 수행하는 이유는 input level에서 완벽하게 DNN's intermediate output을 예측할 수 없기 때문이며, intermediate output들을 관찰하여 유사 역할을 수행하는 cluster들을 merge하기 위함입니다.



위 그림에서 빨간색 X 표기가 fine-tuning을 진행하며 학습(평균)된 layer의 max-value이며, Box-plot으로 표현된 값들이 실제 각 input image 별 output matrix의 max-value들 입니다. Activation function으로 ReLU를 사용하므로 min-value는 항상 0 이며, 도식에서 생략했습니다. QAT 방법(각 도식에서의 맨 왼쪽)으로 max 값들을 평균할 경우 outlier들을 거의 모두 포함하는 형태로 학습되지만, 저희 논문의 contribution들을 적용한 결과(각 도식에서의 Cluster1~4), outlier들이 제외된 형태로 학습되는 것을 볼 수 있습니다. 이는 도식에 사용된 AlexNet 모델 외에 ResNet50 등에서도 동일한 현상을 보였고, 따라서 모든 경우에 대해 적용 가능하며 우수한 성능을 보임을 알 수 있습니다.

[2-3] 한양대학교 대학원 ML 시스템 연구실 DNN Model Quantization - 2 (3 개월)

[2-2]의 Quantization 연구를 진행하던 중 파생된 작은 연구였으며, 2022 ICEIC(International Conference on Electronics, Information, and Communication) 학회에 등재되어 발표했습니다. 논문 제목은 Quantization training with two-level bit width 입니다.

DNN Quantization 기술들 중 하나인 FaceBook의 QuantNoise와 Google의 QAT를 응용했습니다. QuantNoise는 QAT의 Fake Quantization(이하 FQ)을 개선시킨 기법이며, 저희 기술인 Fake Single Precision Training(이하 FST)은 이를 보다 개선한 기법입니다.

FQ은 fine-tuning을 진행할 때 forward propagation에서 사용되는 기법인데, 이는 weight matrix 혹은 output matrix를 quantize & dequantize 하여 고의로 quantization error를 발생시키는 기법입니다. 이로 인해 training loss에 quantization error가 포함되게 되며, 모델은 quantization error가 감소하는 방향으로 학습하게 되고, 결과적으로 quantization error로 인한 accuracy drop이 감소하게 됩니다.

QuantNoise는 FQ를 진행할 때 전체 matrix에 대해 quantize & dequantize 를 진행하게 되면 training loss 가 지나치게 biased 됨을 강조하며 전체 matrix가 아닌, matrix의 일부 값들에 대해서만 확률적으로 FQ를 적용하는 것을 제안했습니다. 하지만 저희가 실험하며 연구했을 때 QuantNoise 기법은 dataset과 DNN model에 따라 성능 차이가 매우 크게 발생하며, 논문에 참고된 dataset과 DNN model을 제외하면 QAT에 비해 저조한 성능을 보였습니다. 이는 학습 과정에서 충분히 quantization error를 유발하지 못한 것이 원인이었고, 저희는 보다 generalized 된 성능을 보이도록 조정했습니다.

INT4 Quantization의 경우 Fine-tuning을 진행하면서 발생하는 quantization error가 매우 커서 모델 자체의 성능이 많이 하락합니다. QuantNoise는 이를 방지하기 위해 일부 weight 값들에 대해서만 INT4 자료형을 이용하여 FQ (quantize & dequantize)을 진행하는데, FQ가 적용되지 않은 나머지 weight 값들이 충분히 학습되지 않아 문제가 됐습니다. 따라서 FST는 일부 weight 값들에 대해서는 INT4 자료형을, 나머지에 대해서는 INT4 보다 큰 자료형을 이용하여 FQ을 진행하도록 하여 QuantNoise와 QAT의 단점을 모두 극복할 수 있도록 개선했습니다.

[2-4] 한양대학교 대학원 ML 시스템 연구실 Artificial Intelligence Assistant (2 개월)

Stanford의 OVAL Lab에서 제작한 Almond라는 인공지능비서 개발에 잠시 참여했습니다. 저희에게 한글 버전의 Almond를 제작해 달라는 요청이 있었으며, 약 2개월 간 Almond의 시스템에 대해 분석해 보았습니다. 두 연구진의 이해 관계가 맞지 않아 중단하게 되었지만, 그 과정에서 인공지능비서의 작동 process를 배울 수 있었으며, 사용되는 인공지능 모델 중 하나인 Seq2SQL 에 대해 공부해 볼 수 있었습니다.

[3-1] Kakao 추천 팀 (자동차 동영상 & 만화) 추천 알고리즘 개발 (2 개월)

총 두 가지 서비스 및 도메인에 대한 추천 알고리즘을 개발했습니다. 첫 번째는 서비스는 자동차 동영상이며, 두 번째는 만화 서적입니다. 각 서비스의 개선을 위해 시도한 내용은 다음과 같습니다.

- 자동차 동영상 추천 알고리즘
  - (실험-1, 2) Thompson Sampling(Multi-armed Bandit) 하이퍼 파라미터 조정
  - (실험-3, 4) Ensemble의 Ranking Algorithm 개선
  - (실험-5) Item2Vec 모델 적용
- 만화 서적 추천 알고리즘
  - (실험-6) Word2Vec 학습 데이터 변경
  - (실험-7) Ensemble의 Ranking Algorithm 개선

위 내용들은 모두 추천 알고리즘의 개선을 위해 Offline Simulation Test, Online A/B Test를 진행하며 실험한 내용입니다. 실험 이유와 방법, 결과, 그리고 결과 분석 등의 자세한 내용들은 아래에 실험 순서 대로 기입했습니다.

& 진행했던 시간 순서로 나열한 것이며, 당시 실험을 진행했던 이유/결과/분석 내용들을 작성했습니다.

& 아래 실험 내용에서 "**ALS**" 라고 지칭하는 모델은 Collaborative Filtering에 이용되는 Latent Factor 모델을 **Alternating Least Squares optimization을 통해 학습한 모델**을 말합니다. 이 모델은 다음 링크된 논문을 기반으로 개발됐습니다. Collaborative Filtering for Implicit Feedback Datasets

실험 1. Thompson Sampling, Hyper Parameters Adjustment

1-1. 실험 동기

인턴 시작 후의 첫 실험이었으며, 실험을 진행할 때에는 Online A/B Test를 진행하며 시간이 소요되므로, 효율적인 실험 scheduling을 위해 가장 빠르고 간단하게 시작할 수 있는 하이퍼 파라미터 조정 실험을 진행했습니다.

다음미디어의 자동차 동영상의 사용자 시청 로그를 사용하여 생성되는 matrix(Users X Items) 데이터가 있었습니다. 해당 matrix는 Latent Factor 모델을 학습하기 위해 사용되는데, matrix의 sparsity(= # NonZero / # User * # Item)가 다른 서비스들에 비해 매우 높았습니다. 즉, matrix에서 0이 아닌 값들이 다른 서비스들에 많았으며, 이는 공통된 동영상들을 많이 시청한다는 것을 의미하므로, 새로운 동영상을 더 많이 보여주는 것이 도움이 될 것이라 생각했습니다(이는 당시의 생각이며, 현재 다시 돌아 보자면, 'sparsity가 높은 것이 도메인의 특성인 가' 에 대한 고민이 부족했다고 생각합니다).

1-2. 개선 방법

새로운 동영상들을 많이 보여주기 위해, 새로운 동영상들이 MAB에서 더 많이 select될 수 있도록, 즉, 더 많은 노출 기회를 갖을 수 있도록 조치했습니다. Thompson Sampling의 h-params 중, empirical draw의 대상이 되는 threshold를 낮췄고, 비교적 새로운 arm들이 beta sampling의 기회를 더 많이 갖도록 했습니다.

1-3. 결과 (실험 기간: 4일)

결과적으로, 매우 모험적인 Explore가 야기됐기 때문에 CTR이 하락했습니다. Empirical draw는 이미 threshold 이상 노출된, 비교적 검증된 arm들을 select하여 추가적인 beta sampling 기회를 주기 위함입니다. 하지만 threshold를 낮춤으로써 비교적 덜 검증된 arm들 까지 empirical draw에서의 beta sampling 기회를 갖게 됐습니다. 여기서 중요한 점은, 당시 H-param에 따르면 새로운 arm들의 expected reward가 수렴된 arm들에 비해 매우 높았다는 것입니다. 따라서 충분한 횟수의 노출이 발생하지 않은 arm들은 beta sampling 시 매우 높은 확률로 수렴된 arm들보다 예상 보상 확률이 높고, 검증된 arm들을 노출시키기 위한 empirical draw 영역의 수렴된 arm들을 모두 몰아냈습니다. 그렇게 덜 검증된 arm들이 계속해서 select 됐고, 낮추기 이전의 threshold와 낮춘 후의 threshold 사이의 노출 횟수를 가진 arm들은 대부분 click이 0이었으며, 따라서 전체 CTR은 낮아졌습니다.

이는 배포 기간을 더 길게 부여했더라도 유사한 결과를 불러왔을 것이라 생각했고, 4일 간의 배포 후에 실험을 중단했습니다. 물론 실험 설계 자체는 exploit을 기준으로 설계되었는데, 저희가 유도한 것은 수렴된 arm들이 exploit 되는 시점에서 새로운 arm들이 유입될 때, 더욱 더 많은 새로운 arm들이 노출 기회를 갖도록 하는 것이었습니다. 하지만, 트래픽이 낮아 4일이 지난 시점까지도 explore만이 발생했으며, 더 오랜 기간 배포한다 해도 계속해서 변하는 트렌드와 같은 편차에 의해 해당 bandit의 가치는 계속해서 감소될 것입니다. 즉, 만약 해당 bandit의 arm들에 많은 arm들이 수렴되는 시점이 온다 해도, 이는 매우 오랜 시간이 흐른 뒤였을 것이며, 오랜 시간이 흐른 뒤에도 해당 bandit의 가치가, 그리고 arm들의 가치가 이전과 같을 수는 없기 때문에 이번 실험이 목표하는 바를 이룰 수 없다고 판단했습니다.

Experiment 2. Thompson Sampling, Hyper Parameters Adjustment

2-1. 실험 동기

실험 2에 영향을 미친 실험-1에서 배운 사실은 다음 두 가지입니다.

- 4일이라는 시간 조차 부족했다.
  다음미디어 자동차 동영상 서비스의 경우 트래픽이 매우 낮으며, 그만큼 추천 결과의 노출 횟수가 매우 적다.
- Empirical draw 실험의 경우 그 초점이 Exploit에 맞춰져 있었다.
  MAB의 arm들 중 다수의 arm들이 이미 수렴해 있는 상황에서, 새로운 arm들이 bandit으로 유입될 경우, 빠르게 empirical draw의 대상으로 만드는 것이었다. 하지만 목표하는 Exploit 단계 까지 기다린 후 결과를 비교하는 것은 그 결과가 유의미한 것인지 알 수 없을 뿐 아니라(실험-1 결과 참고), 적어도 인턴인 우리에게는 시간이 소모적이다.

이러한 이유로 Explore에 관여하는 실험이 효과적일 것으로 생각했으며, 보다 덜 모험적인 Explore를 진행하여 트래픽이 낮은 단점을 보완하기로 했습니다. 트래픽이 낮기 때문에 현재 H-param들로는 Exploit이 거의 발생되지 않기 때문입니다.

2-2. 개선 방법

자동차 동영상 서비스의 평균적인 CTR은 3% 정도를 기록했으며, 이로 인해 초기 arm들은 수렴한 arm들에 비해 매우 큰 expected reward를 갖고 있었고, 따라서 bandit들은 대체로 새로운 arm들을 select했습니다. 이로 인해 모험적인 Explore가 계속되었고, 작은 트래픽으로 인해 수렴이 더욱 늦어진다는 점에서 Explore를 보다 감소시켜야 한다고 판단했습니다. 게다가 콘텐츠들은 여러 편차들 중 하나인 time bias에도 영향을 받으므로 오랜 시간 explore를 지속해야 할 이유가 더욱 없었습니다. 따라서 모든 아이템들을 계속해서 Explore하는 것 보다, 아직 새로운 arm들이 많을 지라도 지속적으로 비교적 검증된 arm들을 select 하는 것이 CTR을 향상시킬 것이라 예상했습니다.

2-3. 결과 (실험 기간: 6일)

배포한 후 CTR이 지속적으로 상승된 형태를 보였습니다. 또한, 데이터를 확인했을 때, 대조군과의 비교를 통해 훨씬 적은 비율로 새로운 arm들을 select 하고 있음을 확인했습니다.

Experiment 3&4. Ensemble Method, Rank Fusion to Weighted-sum

3&4-1. 실험 동기

실험 3과 4는 굉장히 empirical한 내용으로, weighted-sum에 이용 할 weight를 탐색하는 실험입니다. 자동차 동영상이라는 도메인에 익숙한 팀원이 없어서 추천이 잘 되고 있는 것인가에 대한 정성적 판단이 어려웠고, 실제로 Latent Factor 모델을 통해 사용자 소비 패턴을 관찰해 보고자 했지만 추천 결과의 질을 평가할 수 없었습니다. 따라서 좋은 성능을 보이는 모델을 찾기 위해 Ensemble의 fusion 알고리즘인 Rank Fusion에서 Weighted-sum으로 변경하는 실험을 진행했습니다.

3&4-2. 개선 방법

기존에는 앙상블 기법으로 Rank Fusion을 사용했으며, Rank Fusion은 모델 별 추천 결과들의 순위에 따라 점수를 부여하는 방법입니다. Rank Fusion의 경우 모델 별로 모두 동일한 가치를 갖고 있기 때문에, Word2Vector의 1순위 아이템과 ALS의 1순위 아이템이 점수가 같았습니다. 따라서 Weighted-sum을 사용하여 ALS에 힘을 실어주기로 했고, 이를 위해 Weighted-sum의 적절한 weight set을 찾아야만 했습니다. Weight set 선정 방법은 다음과 같습니다.

1. 당시 배포된 지 20일 가량이 넘었던, 잘 작동하고 있는 대조군에서 bandit 별로 상위 CTR을 보유하고 있는 arm들을 선정하여 정답 데이터로 삼음
2. Weighted-sum으로 앙상블 했을 때, 정답 데이터들이 가장 많이 포함되는 weight set을 선정

이 때 정답 데이터와 ensemble 결과를 비교할 때에는 Precision과 nDCG를 사용했는데, 두 가지 논문을 참고했습니다.

- Context-Aware Recommender System: A Review of Recent Developmental Process and Future Research Direction
- Performance of recommender algorithms on top-N recommendation tasks

실험 3은 ALS에 weight이 가중된 파라미터를 사용했으며, 실험 4의 경우 Word2Vector에 힘이 실렸습니다다.

3&4-3. 결과 (실험 기간: 실험-3 4.5일, 실험-4: 5일)

실험 3과 4 모두 대조군에 비해 낮은 CTR을 기록했습니다. 이에 대해 분석한 내용은 다음과 같습니다.

- ALS가 overfitting/underfitting 되었을 수 있다.
  이를 파악하기 위해, ALS 학습에 사용되는 파라미터를 확인해 보았으나, alpha/regularization에 해당하는 파라미터의 경우 다른 서비스들의 파라미터와 비교했을 때 유사한 값을 사용하고 있었다. 하지만 latent factor의 dimension이 matrix에서의 item 개수에 비해 지나치게 높다는 것이 확인됐다. 파라미터를 확인한 후 ALS의 추천 결과 또한 정성적으로 확인해 봤으나, top 100 아이템들이 지나치게 포함되지 않았다는 점(0.3 정도의 비율)에서 overfitting이 되었다고 판단할 수는 없었으며, 되지 않았다고 판단할 수도 없었다.
- 추천 결과를 생성할 때, 사용되는 아이템들의 리스트가 부적절하다.
  Bandit 별로 source 아이템에 대해 추천될 아이템들을 선정하는데, 이 때 추천될 아이템들의 후보군들이 매우 제한되어 있다. 이는 자동차 동영상 연관 추천의 다른 서비스들과의 큰 차이점이다. 해당 추천 아이템 풀의 제한을 없애고 추천 결과를 생성, 아이템 풀의 아이템들이 어느정도 순위에 위치해 있는지 확인해 봤을 때 평균적으로 30위가 넘었다. 즉, ALS와 Word2Vector이 생각하는 진짜 유사한 아이템들은 추천되지 못하고 있었다.
- Weight set 선정 과정에 잘못된 점이 있다.
  모델 별 추천 결과의 score를 확인하지 못했다. Rank Fusion과는 다르게, Weighted-sum에서는 모델들이 채점한 아이템 별 score를 비교하고, 대소 관계에 따라 순위를 결정한다. 이는 모델들 별 score 1점이 다른 모델들의 1점과 동일한 가치를 지니게 됨을 의미하는데, 모델 별로 score 분포를 확인해 봤을 때 매우 다른 양상을 띄었다. 만약 score들의 분포를 normalization을 통해 맞춰주거나, weight set 선정 시뮬레이션 당시 weight의 범위를 제한하지 않았더라면, 또 다른 weight set이 선정됐을 수도 있다.

Experiment 5. Item2Vector Model

5-1. 실험 동기

사용중인 아이템 풀이 부적절함을 깨달았지만, 아이템 풀을 계속해서 사용할 수 밖에 없는 상황이기 때문에 이를 최대한 활용하고자 했습니다. ALS의 경우 user가 시청한 동영상들에 대한 목록을 사용해서 학습을 진행하지만, stream에 대한 정보는 존재하지 않습니다. 즉, 어떤 동영상을 봤을 때, 그 직전/직후에 시청한 동영상이 무엇인지는 학습에서 고려하지 않습니다. 따라서 이러한 stream 성질을 강화할 수 있는 Item2Vector를 사용할 경우 보다 사용자의 의견을 많이 반영할 수 있을 것이라 생각했습니다.

5-2. 개선 방법

Item2Vector의 모델은 이미 라이브러리 존재했기 때문에 이를 활용했으며, 하이퍼 파라미터에서 일반적인 Item2Vector와 큰 차이가 있습니다. 일반적인 Item2Vector와 논문에서는 window size를 stream 길이 전체(유저 한명이 연달아 시청한 목록)를 사용합니다. 실제로 학습후 embedding vector들을 시각화 해 봤을 때, window size가 커짐에 따라 그 cluster들이 분명하게 나뉘는 효과가 있었습니다. 하지만 이렇게 학습할 경우 우리가 원하는 직전/직후에 대한 정보를 강조하여 학습할 수 없으리라 생각했고, 평균 stream 길이와 중앙값을 고려하여 window size를 선택했습니다.

Window size 외에, Word2Vector에서 사용하는 frequent sampling을 사용하여 과도하게 자주 등장하는 아이템들을 일부 제외 시킴으로서 top100 아이템들이 과도하게 학습되는 것을 방지했습니다.

5-3. 결과 (실험 기간: 5일)

추천 결과를 확인해 보면, 대략 40% 정도의 아이템들이 ALS와 다른 아이템으로 추천되었으며, CTR은 소폭이지만 지속적으로 상승했고, 다음미디어 자동차 동영상 연관 추천에 있어서 ALS 보다 Item2Vector가 더 적절함을 확인했습니다.

Experiment 6. Word2Vector Improvement

6-1. 실험 동기

픽코마 작품홈에서는 Word2Vector, ALS, VGG19 모델을 이용하여 text/als/image 측면에서 유사도를 계산합니다. 그런데 Word2Vector의 경우 일본어 wikipedia를 데이터셋으로 하여 학습을 했는데, 해당 데이터셋에서도 형태소 분석기를 통해 명사/대명사만을 사용하여 학습했습니다. 하지만 일본어를 잘 사용하는 팀원이 해당 형태소들만을 사용하는 것에 의문을 가졌고, 일본어 감성어 품사 빈도를 조사한 결과 형용사와 동사의 비중이 38%로 꽤 큰 비중을 차지한다는 연구 결과(Japanese Emotion Corpus Analysis and its UseFor Automatic Emotion Word Identification)를 확인했습니다.

6-2. 개선 방법

형태소 분석 후 Word2Vector 학습에 명사/대명사 뿐만 아니라, 동사와 형용사를 포함시켜 학습시켰습니다. 그리고, 추천 결과 생성을 위해 유사도를 계산할 때에도 데이터의 동사/형용사를 포함하여 계산했습니다.

6-3. 결과 (실험 기간: 2일)

명사/대명사/형용사/동사를 포함하여 Word2Vector 모델을 생성하고 활용하는 실험을 진행했지만, CTR과 CVR 모두 하락했습니다. 이에 대한 분석은 다음과 같습니다.

- 감성어 품사의 빈도 중 형용사/동사가 38%이지만, 명사의 경우 47%이다. 심지어 な형용사의 경우 명사로 분류되며, 추천에 필요한 감성어는 명사로도 충분히 충당되고 있을 수 있다.
- 형태소 분석 시 문법저긍로 존경, 수동을 뜻하는 (ら)れる가 동사로 분류된다. 즉, 동사를 포함할 경우 실질적 의미가 없는 형태소들이 포함되어 임베딩이 정확하게 이루어지지 않았을 수 있다.
- 현재 형태소 분석기는 분류 성능이 떨어지므로, 형태소 분석기를 개선시킨 후에 Word2Vector을 조정하는 것이 맞을 수 있다.

Experiment 7. Ensemble Method - Weighted-sum to Weighted Rank Fusion

7-1. 실험 동기

픽코마 작품홈에서는 철저하게 ALS 위주의 추천을 진행합니다. 앙상블 결과 리스트의 모든 아이템들은 ALS를 사용한 추천 결과에 존재하는 아이템들이며, Word2Vector와 VGG19 모델들은 ALS 리스트의 아이템들에 가중치를 부여하여 그 리스트를 재정렬하는 효과로 사용됩니다. 이러한 추천 형태는 empirical하게 실험해 봤을 때 CTR과 CVR이 가장 높게 나타나기 때문에 사용되고 있습니다.

그런데, 이전 실험 기록을 통해 ALS만 사용하여 추천하는 것이 가장 높은 CVR을 보이고 있음을 확인했으며, Word2Vector와 VGG19는 CF 알고리즘인 ALS의 cold start problem을 완화하기 위해 사용되고 있었습니다. 하지만 현재처럼 Weighted-sum을 사용할 경우 Word2Vector와 VGG19가 ALS 추천 결과의 순서를 크게 좌우하게 됨을 알 수 있었습니다. 예를 들자면, (20위의 ALS 추천 아이템)이, (80위의 ALS 아이템 + 80위의 Word2Vector 아이템) 조합에 의해 밀려나는 현상을 쉽게 확인할 수 있었습니다. ALS 위주의 추천을 위해 사용하는 Weighted-sum이지만, 부분적인 중복에 의해 순위 조정이 너무 크게 이루어지고 있다고 판단했습니다.

7-2. 개선 방법

이러한 문제를 해결하고자 Weighted Rank Fusion을 사용해 보았습니다. Rank Fusion은 모델 별 순위에 따라 점수를 부여하여 앙상블을 진행하는데, 그냥 Rank Fusion을 사용할 경우 모든 모델들이 같은 가치를 지니게 됩니다. 즉, ALS 2순위 아이템이 Word2Vector 혹은 VGG19에서의 1순위 아이템보다 낮은 순위를 갖습니다. 따라서 ALS 위주의 추천을 유지하기 위해 Weighted Rank Fusion을 사용했습니다. Weighted Rank Fusion은 말그대로 순위 + 가중치를 사용하는 것이며, Rank Fusion의 계산을 모두 포함합니다. Rank Fusion에 대한 설명과 식은 아래를 참고했습니다.

- Reciprocal Rank Fusion outperforms Condorcet and individual Rank Learning Methods

논문에서 *RRFscore*를 이용하여 랭크 별 점수를 계산합니다. 해당 식에서의 k 값이 작아질 수록 높은 순위와 낮은 순위의 점수 차이가 벌어지며, 높아질 수록 그 차이가 작아집니다. 따라서 k 값이 너무 커질 경우, weight까지 계산 했음에도 불구하고 ALS의 저순위 아이템이 Word2Vector/VGG19의 고순위 아이템보다 점수가 낮아지는 현상이 발생하므로, ALS 위주의 추천이 일부 적용되지 않는 현상이 있습니다. 따라서 ALS의 순위를 너무 파괴하지 않으며, ALS에 존재하는 아이템들만을 추천하도록 하는 하이퍼 파라미터 k 값을 찾아 배포했습니다.

7-3. 결과 (실험 기간: 3일)

하지만 실험 결과 CTR과 CVR 모두 하락했습니다. 하락의 원인을 분석하기 위해 고안한 내용은 두 가지가 있는데, 첫 번째는 예전 실험 결과(ALS만 사용하는 것이 가장 좋은 성능을 보임)가 현재도 유효한지 확인이 필요하다는 것입니다. 만약 현재도 같은 양상을 보인다면, 그 다음으로 Fusion의 세 가지 효과에 대해 고민해 보아야 할 것 같습니다.

- Fusion Via a Linear Combination of Scores
  - Skimming Effect
    각 모델 별로 input에 대한 해석이 다르므로, 다양한 모델들을 혼합하는 것이 성능 향상에 유리
  - Dark Horse Effect
    모델들 중 특별히 잘 예측하는 모델이 있고, 그 모델에 weight을 부여
  - Chorus Effect
    각 아이템들이 모델 별 리스트에 얼마나 많이 등장하는 가를 최종 점수 계산에 고려

위 세 가지는 Ensemble 기법을 사용함에 있어 고려해야 하는 내용인데, 저희 실험은 Chorus Effect를 약화시킴과 동시에 Dark Horse Effect를 강화시켰습니다. 하지만 실험 결과 성능이 낮아졌고, 만약 여전히 ALS 단독 사용이 가장 높은 성능을 보인다면, 위 세 가지 요소들에 대한 분석이 필요할 것입니다.

[4-1] 한빛소프트 AR 팀 인공지능파트 Chatbot 개발 (2 개월)

한빛소프트에서 진행한 첫 번째 프로젝트는 원어민 영어 강사를 대체하기 위한 chatbot application을 제작하는 것이었으며, 약 2 개월 간 진행했습니다. App의 파이프라인은 다음과 같습니다.

1. Speech Recognition (SR)
2. Sentence Domain Classification (DC)
3. Sentence Generation (SG)
4. Speech Synthesis (SS)

음성 인식 모듈을 통해 사용자의 질문을 text로 변환하고, 이에 대한 응답 text를 생성한 뒤, 음성 합성을 통해 답변 음성을 생성합니다. 저는 모듈 2(DC)와 3(SG)을 개선시켰습니다.

먼저 2, 3 모듈을 학습하기 위한 데이터셋을 탐색했습니다. App의 대화 도메인들은 모두 일상 대화를 주제로 하므로, 목표하는 도메인들에 속하는 Dialogue Dataset을 탐색했습니다. 탐색 후 stop words를 선정하는 등의 전처리 과정을 거쳤으며, 기존에 존재하는 Dialogue Dataset의 대화 주제 Label들을 app의 도메인에 맞게 재분류 했습니다.

문장 도메인 분류 모듈(2, DC)의 Word Embedding을 개선하기 위해 가장 먼저 모델의 평가 metric을 Accuracy에서 F1-score로 변경했습니다. Word2Vec 학습에 회사 내/외부 데이터셋들을 모두 사용했으며, class 별 데이터 개수의 차이가 매우 컸습니다. 따라서 올바른 모델 평가를 위해 metric 부터 변경했습니다.

Word2Vec은 간단하지만 매우 좋은 성능을 발휘합니다. 하지만 개발하는 app은 영어 학습을 위한 교육 app이었으며, 따라서 사용자들은 대체로 발음과 어휘력이 좋지 않아 app이 음성 인식 모듈(1, SR)이 인식한 text의 상태가 올바르지 않은 경우가 많았고, 이로 인해 2번 모델(DC) 또한 문장을 이해하기 어려웠습니다. 이와 같은 이유로, Word2Vec에 비해 오탈자에 robust 하다고 평가되는 character 기반의 word embedding 학습 기술인 FastText를 이용하여 모듈을 개선했습니다.

3 번 모듈의 경우 LSTM을 활용한 기본적인 Seq2Seq 모델을 이용하여 학습했습니다. 모델의 학습 데이터 뿐만 아니라, 사용자의 음성 데이터 또한 길이가 긴 경우들이 많았고, 이로 인해 Seq2Seq 모델이 답변 문장의 마침표를 제 위치에서 찍지 못하며 마지막 단어를 max sentence length 까지 반복하여 생성해 내는 문제가 있었습니다. 이는 매우 전형적인 Seq2Seq 모델의 문제점으로, 이를 해결하기 위해 attention mechanism을 적용했습니다.

- 데모 스냅샷 및 내용 정리 페이지

[4-2] 한빛소프트 AR 팀 인공지능파트 Speech Synthesis 모듈 개발 (4 개월)

두 번째 프로젝트는 다중 화자 음성 합성 모듈 개발입니다. 여러 사람의 한국어 목소리를 낼 수 있는 음성 합성 모듈이 필요했고, 이를 위해 open source인 Multi-speaker Tacotron를 응용했습니다. 한국어 버전의 음성 합성 모듈을 개발하는 것은 회사 내에서 처음으로 채택된 연구 주제이며, 따라서 음성 합성의 기본적인 이론 부터 공부해야 했습니다. Sampling rate와 같은 기본적인 음성 데이터의 특성 부터 파악해 나갔으며, 파악 이후에는 데모 테스트에 사용할 데이터셋을 탐색했습니다. 데이터셋으로는 학습을 위한 문장 단위의 음성 파일이 필요하며, 해당 음성 문장에 대한 대본(text)이 필요합니다. 따라서 기존에 존재하는 자동화 모듈을 수정했습니다. 데이터셋 마다 다르게 나타나는 음성 파일의 특성에 맞게 하이퍼 파라미터 및 코드를 수정하여 긴 음성 파일을 문장 단위로 분리하고, 분리된 음성 파일들의 대본을 생성 및 제작했습니다.

- 데모 음성 파일 및 내용 정리 페이지: https://jarvis08.github.io/pjt_hbs_multi.html