챌린지 베이스라인 설명


ETRI
한국전자통신연구원
Electronics and Telecommunications Research Institute

목 차

1. 문서의 목적 1

2. DB 1

2.1. 패션 아이템의 메타데이터 DB 1

2.2. 학습 대화 DB 3

2.3. 연속 학습 평가 대화 DB 4

3. 파일 구조 5

4. 실행 6

4.1. 실행 환경 6

4.2. 아나콘다에서 실행 환경 셋팅 7

4.3. 실행 방법 7

4.4. 옵션 8

5. 베이스라인 설명 9

5.1. 개요 9

5.2. 데이터 입출력 11

5.3. 모델 11

5.4. 학습 및 평가 12

6. 라이선스 12

7. 연락처 12

1. 문서의 목적

본 문서는 “2023 ETRI 자율성장 인공지능 경진대회”를 위한 챌린지 베이스라인을 설명하는 문서이다.

2. DB

2.1. 패션 아이템의 메타데이터 DB

- 패션 아이템의 메타데이터(./data/mdata.wst.txt.2023.01.26)는 형태/색채/소재/감성 특징을 포함하고 그 약어는 (표 1)과 같음

특징 종류	약어
형태	F
소재	M
색채	C
감성	E

표 1. 패션 아이템의 메타데이터 특징 종류별 약어

- 패션 아이템의 종류별 개수와 약어는 (표 2)와 같음

종류	약어	개수
자켓	JK	529
코트	CT	425
점퍼	JP	541
니트	KN	559
스웨터	SW	353
셔츠	SH	228
블라우스	BL	312
가디건	CD	341
조끼	VT	190
원피스	OP	274
치마	SK	480
바지	PT	588
신발	SE	313
합계		5,133

표 2. 패션 아이템의 종류별 개수와 약어

- 항목별 패션 아이템의 종류는 (표 3)과 같음

항목	약어	패션 아이템의 종류
겉옷(outer)	O	자켓(JK), 점퍼(JP), 코트(CT), 가디건(CD), 조끼(VT)
상의(top)	T	니트(KN), 스웨터(SW), 셔츠(SH), 블라우스(BL)
하의(bottom)	B	치마(SK), 바지(PT), 원피스(OP)
신발(shoe)	S	신발(SE)

표 3. 항목별 패션 아이템의 종류

- 파일 포맷은 "패션 아이템의 이름 - 항목 - 패션 아이템의 종류 - 특징 종류 - 특징 기술" 형태로 구성됨
  - 예제는 (그림 1)과 같음
  - 첫 번째 줄에서, "BL-001"은 패션 아이템의 이름을 말하고, "T"는 웃옷을 말하고, "BL"은 블라우스를 말하고 "F"는 형태 특징을 말하고, "단추 여밈 의 전체 오픈 형"은 기술된 특징을 말함

```
 1 BL-001  T   BL  F   단추 여밈 의 전체 오픈형
 2 BL-001  T   BL  F   스탠드 칼라 와 브이넥 네크라인 의 결합 스타일
 3 BL-001  T   BL  F   손목 까지 내려오 는 일자형 소매
 4 BL-001  T   BL  F   여유로운 핏
 5 BL-001  T   BL  F   어깨 에서 허리 까지 세로 절개 에 풍성 한 러플 장식
 6 BL-001  T   BL  F   와이드 커프스
 7 BL-001  T   BL  M   면 100%
 8 BL-001  T   BL  M   구김 이 가 기 쉬운
 9 BL-001  T   BL  M   드라이 클리닝 권장
10 BL-001  T   BL  C   시원_해 보이 는 소라색 SKY BLUE
11 BL-001  T   BL  C   단색 의 깔끔_한 느낌
12 BL-001  T   BL  E   여성 스러운
13 BL-001  T   BL  E   페미닌 한
14 BL-001  T   BL  E   세련 된
15 BL-001  T   BL  E   사랑 스러운
16 BL-001  T   BL  E   깔끔_한
17 BL-001  T   BL  E   오피스 룩
18 BL-001  T   BL  E   로맨틱 한 데이트 룩
19 BL-001  T   BL  E   포멀 한 이미지
20 BL-001  T   BL  E   단정 한 오피스 걸 룩 이미지
21 BL-002  T   BL  F   넓 은 셔츠 칼라 네크라인
22 BL-002  T   BL  F   앞 중심 에 반 오픈 끈 여밈 있 는 스타일
23 BL-002  T   BL  F   드롭 숄더
24 BL-002  T   BL  F   퍼프 형 소매 로 여성스러움 을 가미 함
25 BL-002  T   BL  F   세로 주름 조직
26 BL-002  T   BL  F   풍성 한 여유 가 있 는 몸통
27 BL-002  T   BL  M   면
28 BL-002  T   BL  M   세로 방향 크릴 클 조직
29 BL-002  T   BL  M   비치 는
30 BL-002  T   BL  M   얇 은
31 BL-002  T   BL  M   드라이 클리닝
32 BL-002  T   BL  C   단색 의
33 BL-002  T   BL  C   깔끔_한 아이보리
34 BL-002  T   BL  E   여성 스러운
35 BL-002  T   BL  E   편안_한
36 BL-002  T   BL  E   복고풍 의
37 BL-002  T   BL  E   로맨틱 한
38 BL-002  T   BL  E   섹시 한
```

그림 1. 패션 아이템의 메타데이터 예제

2.2. 학습 대화 DB

- 학습 대화 DB(./data/task*.wst.txt, *는 task 번호)는 "발화번호 - <CO>|<US>|<AC> - 발화 - TAG"로 구성되어 있음
  - 연속 학습은 서로 다른 여러 개의 문제가 순차적으로 입력 될 때, 새로운 문제를 학습하여도 이전에 학습한 문제의 풀이 방법을 잊지 않고 여전히 잘 풀 수 있도록 하는 학습 방법임
  - 따라서, 연속 학습 수행이 가능한 상황을 마련하기 위하여 서로 독립적인 여러 개의 task를 가지도록 DB가 구성되어 있음
  - 'task*.wst.txt' 의 *는 task 번호를 의미하며, 'task1.wst.txt'는 첫 번째 task의 학습 대화 DB, 'task2.wst.txt'는 두 번째 task의 학습 대화 DB에 해당함
  - <AC>는 추천된 패션 코디를, <CO>코디 에이전트를, <US>는 사용자를 말함
  - TAG는 (표 4)와 같음

TAG	내용
INTRO	대화 도입부
EXP_RES_*	추천 의상 설명
USER_SUCCESS	사용자가 기술한 추천 의상 성공
USER_SUCCESS_PART	사용자가 기술한 일부 추천 의상 성공
USER_FAIL	사용자가 기술한 추천 의상 실패
FAIL	대화에서 의상 추천 실패
ASK_*	사용자가 원하는 의상유형이나 스타일, 색상 등에 대한 질문
CONFIRM_*	확인 질문
SUCCESS	대화에서 의상 추천 성공
CLOSING	대화 종료
WAIT	대기 요청
SUGGEST_*	제안 발화
NONE	의상 없음
HELP	사용자 지원

표 4. 학습 대화 DB의 TAG

- 학습 대화 예제는 (그림 12)와 같음

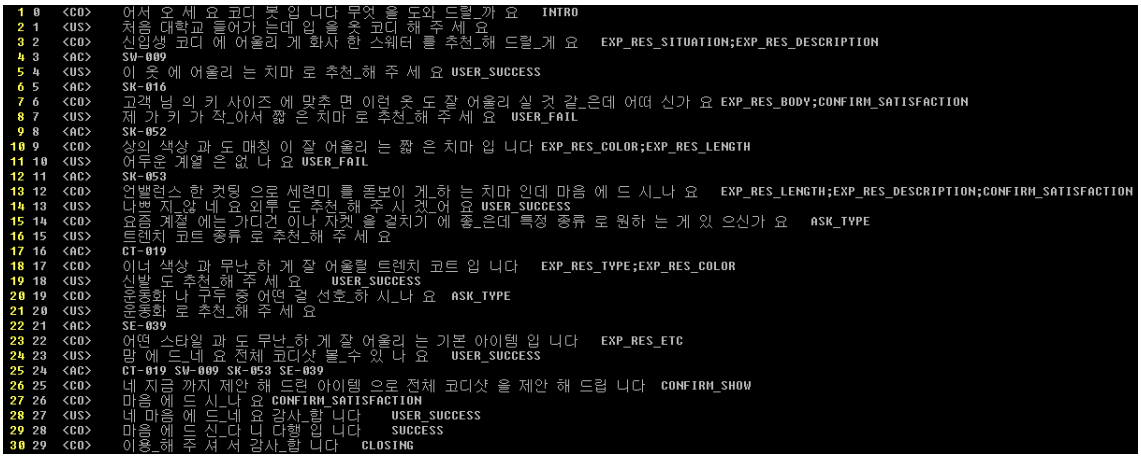


그림 2. 대화 예제

2.3. 연속학습 평가 대화 DB

- 평가 대화 DB(./data/cl_eval_task*.wst.dev, *은 task 번호)는 "; 대화번호", "US|CO 발화", "R1|R2|R3 패션 코디"로 구성되어 있음
  - 연속 학습 평가를 위하여, 서로 독립적인 task에 해당하는 평가 대화 DB가 구축되어 있음
  - 'cl_eval_task*.wst.dev' 의 *는 task 번호를 의미하며, 'cl_eval_task1.wst.dev'는 첫 번째 task의 평가 대화 DB, 'cl_eval_task2.wst.dev'는 두 번째 task의 평가 대화 DB에 해당함
  - US는 사용자를, CO는 코디 에이전트를 말함
  - R1는 1순위, R2는 2순위, R3는 3순위의 패션 코디를 말함
- 평가 대화 DB의 예제는 (그림 3)과 같음

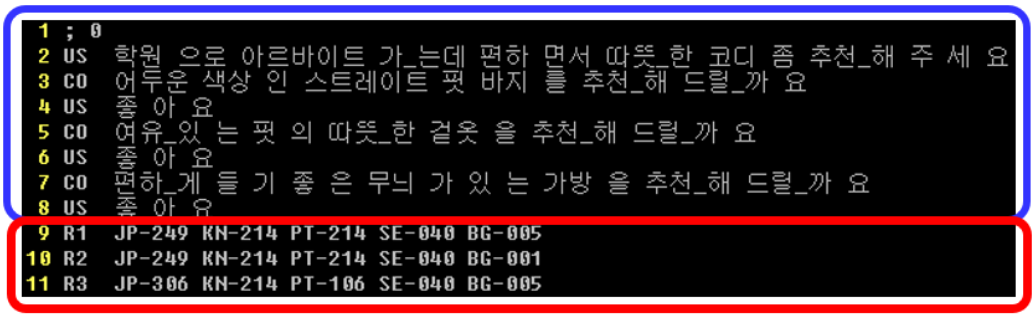
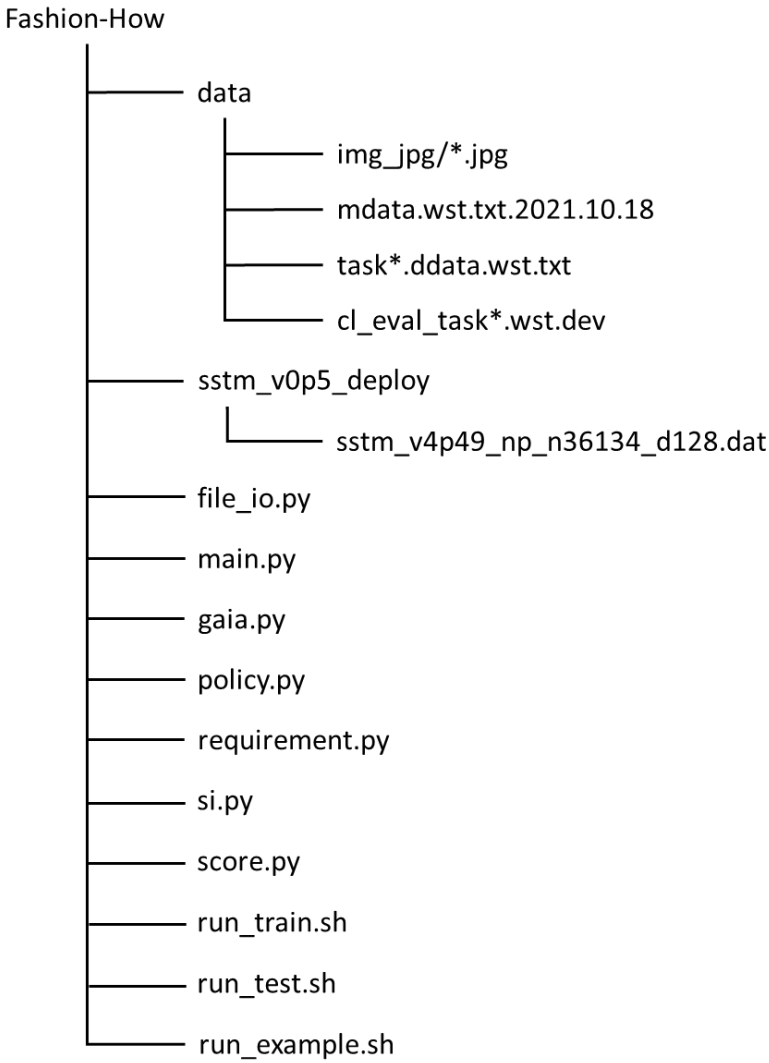


그림 3. 연속학습 평가 데이터의 예제

3. 파일 구조

- ./data/img_jpg/*.jpg
  - 패션 아이템의 이미지 DB
- ./data/mdata.wst.txt.2023.01.26
  - 패션 아이템의 메타데이터 DB
- ./data/task*.wst.txt (*은 task 번호)
  - 학습 대화 DB
- ./data/cl_eval_task*.wst.dev (*은 task 번호)
  - 연속학습 평가 대화 DB
- ./sstm_v0p5_deploy/sstm_v4p49_np_n36134_d128.dat
  - ETRI가 자체 개발한 서브워드 임베딩 DB
- ./main.py
  - 프로그램 실행 시 외부 인자를 받아서 설정하는 소스코드
- ./gaia.py
  - 설정된 외부 인자에 따라 훈련하거나 시험을 수행하는 소스코드
- ./file_io.py
  - 패션 아이템의 메타데이터 DB, 학습 및 평가 대화 DB, 이미지 특징 및 서브워드 DB를 읽어 오는 소스코드
- ./requirement.py
  - 사용자의 요구사항을 추정하는 소스코드
- ./policy.py
  - 사용자의 요구사항에 적합한 패션 코디 순위를 산정하는 소스코드
- ./si.py
  - 연속 학습을 적용하기 위한 소스코드
- ./run_train.sh
  - 기 학습한 task에 대한 망각을 줄이는 방향으로(연속 학습이 가능하도록) 학습하는 쉘스크립트
- ./run_test.sh
  - 연속학습을 평가하는 쉘스크립트
- ./run_example.sh

  변화하는 task에 따라 연속적으로 학습 및 평가를 진행하기 위한 예제 쉘스크립트



4. 실행

4.1. 실행 환경

CentOS 또는 Ubuntu

Python 3.8

CUDA 10.1

CUDNN 7.6.4

PyTorch 1.8

4.2. 아나콘다에서 실행 환경 셋팅

conda create --name Fashion-How python=3.8

conda activate Fashion-How

conda install -c anaconda cudatoolkit=10.1

conda install -c anaconda cudnn=7.6.4

pip install torch==1.8

pip install scikit-learn tqdm

4.3. 실행 방법

- 기 학습한 task에 대한 망각을 줄이는 방향으로 학습

./run_train.sh를 실행한다.

```
# $1: --in_file_trn_dialog
# $2: filename of $1
# $3: --in_file_tst_dialog
# $4: filename of $3
# $5: --model_path
# $6: path for saving trained model
# $7: --model_file
# $8: filename of $7 (loaded file after task#1)

CUDA_VISIBLE_DEVICES="0" python3 ./main.py --mode train \
                                    --in_file_fashion ./data/mdata.wst.txt.2023.01.26 \
                                    --subWordEmb_path ./sstm_v0p5_deploy/sstm_v4p49_np_n36134_d128.dat \
                                    --mem_size 16 \
                                    --key_size 300 \
                                    --hops 3 \
                                    --eval_node [6000,6000,200][2000] \
                                    --epochs 20 \
                                    --save_freq 5 \
                                    --batch_size 100 \
                                    --learning_rate 0.005 \
                                    --max_grad_norm 20.0 \
                                    --use_dropout True \
                                    --zero_prob 0.5 \
                                    --permutation_iteration 3 \
                                    --num_augmentation 5 \
                                    --corr_thres 0.7 \
                                    $1 $2 \
                                    $3 $4 \
                                    $5 $6 \
                                    $7 $8

```

- 연속 학습을 평가

./run_test.sh를 실행한다.

```
# $1: --in_file_tst_dialog
# $2: filename of $1
# $3: --model_path
# $4: path for loading trained model

CUDA_VISIBLE_DEVICES="0" python3 ./main.py --mode test \
                                  --in_file_fashion ./data/mdata.wst.txt.2023.01.26 \
                                  --subWordEmb_path ./sstm_v0p5_deploy/sstm_v4p49_np_n36134_d128.dat \
                                  --model_file gAIa-final.pt \
                                  --mem_size 16 \
                                  --key_size 300 \
                                  --hops 3 \
                                  --eval_node [6000,6000,200][2000] \
                                  --batch_size 100 \
                                  $1 $2 \
                                  $3 $4

```

- task 변화에 따라 학습과 평가를 반복

./run_example.sh를 실행한다.

```
# ### train task#1 ###
sh run_train.sh --in_file_trn_dialog ./data/task1.ddata.wst.txt --in_file_tst_dialog ./data/cl_eval_task1.wst.dev --model_path ./gAIa_CL_model
### test task#1 ###
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task1.wst.dev --model_path ./gAIa_CL_model
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task2.wst.dev --model_path ./gAIa_CL_model
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task3.wst.dev --model_path ./gAIa_CL_model

### train task#2 ###
sh run_train.sh --in_file_trn_dialog ./data/task2.ddata.wst.txt --in_file_tst_dialog ./data/cl_eval_task2.wst.dev --model_path ./gAIa_CL_model --model_file gAIa-final.pt
### test task#2 ###
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task1.wst.dev --model_path ./gAIa_CL_model
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task2.wst.dev --model_path ./gAIa_CL_model
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task3.wst.dev --model_path ./gAIa_CL_model

### train task#3 ###
sh run_train.sh --in_file_trn_dialog ./data/task3.ddata.wst.txt --in_file_tst_dialog ./data/cl_eval_task3.wst.dev --model_path ./gAIa_CL_model --model_file gAIa-final.pt
### test task#3 ###
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task1.wst.dev --model_path ./gAIa_CL_model
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task2.wst.dev --model_path ./gAIa_CL_model
sh run_test.sh --in_file_tst_dialog ./data/cl_eval_task3.wst.dev --model_path ./gAIa_CL_model

```

4.4. 옵션

- mode: "train"이면 연속 학습을 위한 학습 모듈이 실행되고, "test"이면 연속 학습 평가 모듈이 실행
- in_file_trn_dialog: 학습 대화 DB 파일명
- in_file_tst_dialog: 평가 대화 DB 파일명
- in_file_fashion: 패션 아이템의 메타데이터 DB 파일명
- model_path: 저장하거나 저장된 학습 모델의 경로명
- model_file: 저장하거나 저장된 학습 모델의 파일명
- eval_node: 패션 코디 결정부의 평가망에서 노드 개수들
- subWordEmb_path: 서브워드 임베딩 DB 파일명
- learning_rate: 학습률
- max_grad_norm: 최대 그래디언트 값

- use_dropout: 드롭아웃 기법 사용 유무
- zero_prob: 패션 코디 결정부의 평가망에서 드롭아웃 노드의 비율
- permutation_iteration: 동일한 학습 데이터 출력에 대해 입력의 순서를 바꾸는 개수
- num_augmentation: 학습 데이터를 증강하는 횟수
- corr_thres: 학습 데이터를 증강할 때 패션 아이템의 대체하는 임계값
- batch_size: 배치 크기
- epochs: 학습 횟수
- save_freq: 모델의 저장 주기
- hops: 요구사항 추정부의 기억망에서 사용하는 hop의 개수
- mem_size: 요구사항 추정부의 기억망에서 사용하는 메모리 크기
- key_size: 요구사항 추정부의 기억망에서 출력 크기
- evaluation_iteration: 입력의 순서를 바꾸어 수행할 평가 횟수
- use_batch_norm: 패션 코디 결정부의 평가망에서 배치 정규화 기법 사용 유무
- use_cl: 연속 학습 적용 유무

5. 베이스라인 설명

5.1. 개요

평가 데이터는 (그림 3)처럼, 적색 부분의 대화와 청색 부분의 패션 코디들로 구성된다. 본 알고리즘의 목적은 대화를 토대로 가장 적절한 패션 코디의 순위를 매기는 것이다.
기본 베이스라인의 구성도는 (그림 4)와 같다. 우선 요구사항 추정부는 기억망을 사용하여 과거 및 현재의 질의응답(대화)으로부터 시간/장소/상황에 적합한 사용자의 요구사항을 추정한다. 여기서 기억망으로 페이스북의 종단간 기억망[1]을 사용한다. 다음으로 다수 계층의 FCN(Fully-Connected Network)으로 구성된 평가망을 사용하여 패션 코디들의 순위를 산정한다. 각 패션 코디는 패션 아이템의 메타데이터로 임베딩을 수행하고, 사용자의 요구사항을 참조하여 3개 계층의 FCN으로 각 후보 패션 코디를 요약하며, 그 요약된 결과를 연결하여 2개의 FCN을 통해 최종 순위를 산정한다.

[1] Sainbayar Sukhbaatar et al., "End-To-End Memory Networks", NIPS2015

본 sub-task에서 다루고자 하는 연속 학습은, 뉴럴 네트워크가 하나의 문제를 학습 한 후 다른 문제를 학습하게 되는 경우 기존에 학습한 문제를 잘 풀지 못하는 파괴적 망각(catastrophic forgetting) 현상을 완화하기 위한 학습 방식이다. 본 베이스라인에서는 연속 학습의 다양한 방식 중, 하나의 task에 대한 학습이 끝나면 학습이 완료된 FCN의 각 node에 대하여 현재 학습한 task에 대한 중요도를 계산한 후, 이후 task를 학습 할 때에 해당 내용을 제한 조건으로 적용하는 정규화 방식을 적용하였다. 정규화 방식의 대표적인 기법 중 하나인 SI(Synaptic Intelligence)[2]를 적용하였으며, 학습 시 사용하는 손실 함수에 SI 손실 항을 추가하여 이전 task에서 중요도가 높았던 node들은 중요도가 낮은 node 대비 새로운 task를 학습할 때에 변화가 적도록 제한하는 방식으로 파괴적 망각 현상을 완화하였다.

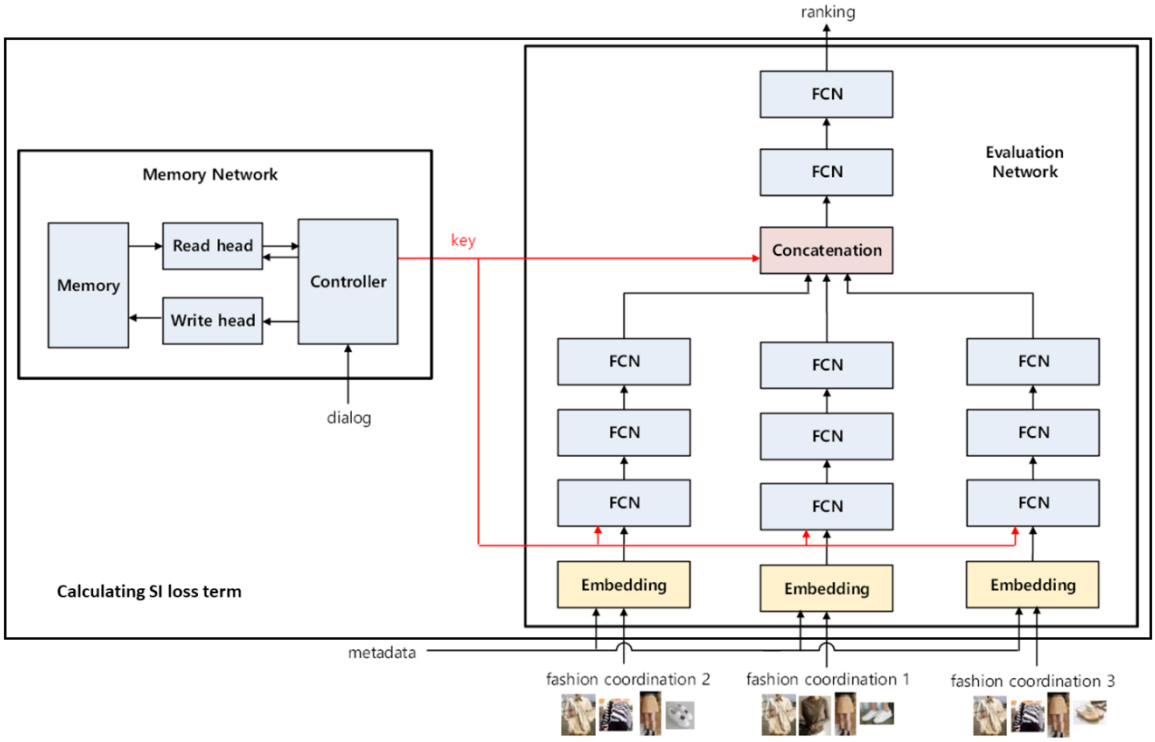


그림 4. 베이스라인 구성도

[2] Friedemann Zenke et al., "Continual learning through synaptic intelligence", ICML2017

5.2. 데이터 입출력

아래의 순서로 학습 및 평가를 위한 DB를 읽는다.

- <file_io.py>에 있는 "make_metadata" 함수를 사용하여 패션 아이템의 메타데이터 DB를 읽고 임베딩을 수행
  - <file_io.py>에 있는 "_load_fashion_item" 함수를 사용하여 패션 아이템의 메타데이터 DB를 읽음
  - <file_io.py>에 있는 "SubWordEmbReaderUtil" 클래스의 인스턴스를 생성하고, <file_io.py>에 있는 "_vectorize_dlg" 함수로 읽어 온 패션 아이템의 메타데이터에 대해 임베딩을 수행
    - "SubWordEmbReaderUtil" 클래스에서 사용하는 <./sstm_v0p5_deploy/sstm_v4p49_np_n36134_d128.dat>은 ETRI 자체 개발한 임베딩 DB임
  - <file_io.py>에 있는 "_categorize" 함수를 사용하여 패션 아이템을 항목별 분류
  - "sklearn" 파이썬 패키지에 있는 "cosine_similarity" 함수를 사용하여 패션 아이템들간의 코사인 유사도를 계산
- <file_io.py>에 있는 "make_io_data" 함수를 사용하여 학습 및 평가 대화 DB를 읽고, 학습 데이터는 평가에 적합한 형태로 변환하며 데이터를 증강
  - <file_io.py>에 있는 "_load_trn_dialog" 함수를 사용하여 학습 대화 DB를 읽음
  - <file_io.py>에 있는 "_load_eval_dialog" 함수를 사용하여 평가 대화 DB를 읽음
  - <file_io.py>에 있는 "_make_ranking_examples" 함수를 사용하여 최종 제안한 패션 코디로부터 대화의 역방향으로 중복 없이 3개의 패션 코디를 후보 패션 코디들로 결정. 또한 USER_SUCCESS가 발생한 대화의 최종 제안한 패션 코디에서 임의로 패션 아이템을 선택하고 그 아이템과 코사인 유사도가 적은 아이템으로 교체함으로써 후보 패션 코디를 생성
    - 후보 패션 코디의 순서는 임의로 배정
  - <file_io.py>에 있는 "_vectorize" 함수와 _memorize" 함수를 사용하여 대화에 대해 임베딩을 수행하고 기억망을 위한 메모리에 저장
  - <file_io.py>에 있는 "_indexing_coordi" 함수와 "_convert_coordi_to_metadata" 함수를 사용하여 패션 코디의 아이템들을 해당하는 메타데이터로 변환

5.3. 모델

대화를 토대로 패션 코디 순위를 산정하기 위한 모델을 생성한다.

- 사용자의 요구사항을 추정하기 위해 <requirement.py>에 있는 "RequirementNet" 클래스의 인스턴스(모델 네트워크)를 생성
  - "RequirementNet" 클래스 내부에서 기억망을 담당하는 "MemN2N" 클래스의 인스턴스를 생성
- 대화에 가장 적합한 패션 코디의 순위를 산정하기 위해 <policy.py>에 있는 "PolicyNet" 클래스의 인스턴스(모델 네트워크)를 생성
  - 요구사항을 참조하여 후보 패션 코디를 평가 요약하기 위한 복수 계층의 FC망 생성
  - 최종 순위를 산정하기 위한 복수 계층의 FC망 생성

5.4. 학습 및 평가

- <gaia.py>에 있는 "train" 함수를 사용하여 학습 데이터를 배치 단위로 읽어 교차 엔트로피로 손실 함수를 사용하여 확률적 경사 하강법으로 학습하고 주기적으로 모델을 저장한다. 첫 번째 task의 학습이 완료 된 이후부터는 교차 엔트로피 손실 항에 파괴적 망각 현상 완화를 위한 SI 손실 함수 항이 추가된 손실 함수를 사용하여 확률적 경사 하강법으로 학습한다.
- <gaia.py>에 있는 "test" 함수를 사용하여 저장된 모델을 불러오고 성능을 측정한다.
  - "gaia.py"에 있는 "_evaluate" 함수를 사용하여 평가 데이터를 입력으로 그래프에 따라 인퍼런스를 수행하고, 그 수행된 결과에 대해 "scipy" 파이썬 패키지의 stats.weightedtau로 평균 Weighted Kendal Tau Correlation을 계산

6. 라이선스

- 본 소프트웨어는 MIT 라이선스(https://opensource.org/licenses/MIT)를 따라야 함
- 타 오픈소스 SW 활용 시 해당 오픈소스 SW 라이선스에서 요구하는 라이선스 준수 의무를 이행해야 함

7. 연락처

담당자: 한란
E-mail: ran.han@etri.re.kr

끝.