|  |  |
| --- | --- |
| 교육 제목 | AI 프로젝트 기반 S/W 전문가 양성과정 |
| 교육 일시 | 21.12.6 |
| 교육 장소 | 학원 |
| **교육 내용** | |
| 오전 | 1. 지난주 REVIEW    1. output이 0/1이면 sigmoid, 아니면 softmax    2. SGDClassifier(loss = log, max\_iter = 5) sc.fit sc.score Sequential compule fit evaluate    3. optimizer = 스텝    4. save\_best\_only = True : 최고 저장    5. patience = 10 : earlystopping 얼마나 기다릴지    6. evaluate [0] 손실, [1] 정확도 2. CNN 컨볼루션 신경망    1. 가로세로 0으로 더 붙임 =  padding = same 원래 사이즈(패딩 한다), valid(안한다) kernel\_size = (3, 3) = mask filter = mask사이즈    2. poolsize = 2 : 1/2    3. strides = 옮기는 칸수    4. maxpulling = 최대값으로 옮김    5. dropout 과적합 방지, 일정 퍼센티지 노드 계산 안함    6. flatten() 평탄화 3. CNN 정리    1. 딥러닝 모델 실행 절차    2. 모델 설정 : Sequential()    3. 이미지는 특성 추출 : 합성곱, 풀링, 과적합(dropout), layer 증가 …    4. Conv2d(커널사이즈 …), : 가중치 마스크 적용    5. Conv2d(커널, …, padding = “same” or “valid”)    6. 맥스 풀링(특징을 최적화 해서 찾아냄)    7. Dropout(소수): 0.25, 25%는 계산 안 함 🡪 2차원 배열    8. flatten() : 1차원 배열로 변환    9. Dense() : layer 추가 1차원 배열, input­\_dim 필요 없음    10. 출력층 추가    11. 컴파일 loss = categorical\_crossentropy > OHE loss = sparse\_categorical\_crossent 🡪 int    12. fit() : modelcheckpoint, earlystopping으로 성능 고려    13. 프레딕트 |
| 오후 | 1. 실습 2. 자연어 처리    1. 토큰화 – 입력할 텍스트가 준비되면 이를 단어별, 문장별, 형태별로 나눔 text\_to\_word\_sequence() 단어로 쪼갠 결과 리스트로 반환    2. token = Tokenizer() # 인스턴스 열기 token.fit\_on\_texts(docs) tokenizer 하기 token.word\_counts 🡪 return ordereddict(순서까지 매기는 dict)    3. token.document\_count = 몇문장이냐 token.word\_counts = 각 단어의 수 token.word\_index = 빈도 내림차순으로 ordering 3. 단어의 OHE    1. 0번지는 인덱스임    2. 순서 토큰화 하고 각각 OHE 4. padding 배열의 길이 맞춤 5. 단어 embedding    1. 공간 낭비를 줄이기 위해 압축?????????    2. 단어간 유사도를 계산함    3. model.add(Embedding(16, 4) input 16, output 4 input\_length = 2 라고 하면 입력되는 단어수는 16개지만 2개씩만 넣겠다 6. 조합해서 실습 |