04. RNN(Recurrent Neural Network, 순환 신경망)

#src: 9_자연어처리/ch04_RNN 참고

1) RNN; 순서를 가진 데이터를 입력하여 단위 간 연결이 시퀀스를 따라 방향성 그래프를 형성하는 모델로 내부 상태(메모리)를 이용하여 입력 시퀀스를 처리함 #'입력 데이터 요약 정보' 기억

- RNN은 가중치 갱신을 위해 과거시점까지 역전파하는 BPTT활용 #순차적 데이터 처리에 적합

(1) 문맥 학습 후 다음 단어 예측하는 예제

```
from keras_preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding, SimpleRNN
t = Tokenizer()    #문장(text) Tokenizer()로 잘라서 encoding
t.fit_on_texts([text])
encoded = t.texts_to_sequences([text])[0]
sequences = []
for line in text.split('₩n'): #₩n을 기준으로 문장 토큰화
    encoded = t.texts_to_sequences([line])[0]
    for i in range(0, len(encoded)-1):
        for j in range(i+2, len(encoded)+1):
            sequences.append(encoded[i:j])
maxlen = max([len(s) for s in sequences]) #sequences에 가장 많은 단어가 들어가 있는 수
#sequences를 훈련가능한 데이터로 만들기
sequences = pad_sequences(sequences=sequences, maxlen=maxlen, padding='pre') #앞에 0을 붙임
X = sequences[:, :-1]  #독립변수(X), 종속변수(Y) 분리
Y = sequences[:, -1]
#종속변수(Y)를 원핫 인코딩
vocab_size = len(t.word_index)+1  #토크나이저 정수 인코딩은 1부터, 원핫 인코딩은 0부터 시작 +1
Y = to_categorical(Y, num_classes=vocab_size)
model = Sequential()    #RNN 모델 생성
model.add(Embedding(vocab_size, 10, input_length=X.shape[1]))  #희소행렬로 변환 (10:벡터)
model.add(SimpleRNN(32))  #SimpleRNN부분에 LSTM(Long Short-Term Memory)사용 가능
model.add(Dense(vocab_size, activation="softmax"))
#모델 학습과정 설정
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=['accuracy'])
hist = model.fit(X, Y, epochs=200, verbose=2) #모델 학습시키기
```