휴먼프밍

팀 프로젝트

머신러닝을 이용한 게시글 분류

이화여대 휴먼기계바이오공학과

위재연 황서현

INDEX

CONTENTS

배경

왜 게시글 분류인가?

- 자연어 처리를 이용한 머신러닝 수요 증가
- 스스로 데이터 수집부터 가공까지 시도
- Github에 관련 예제 정보가 풍부

데이터 전처리 하기

크롤링이란?

데이터 수집

데이터 셔플

데이터 가공

데이터 전처리 하기

크롤링이란?

- 웹페이지의 내용을 그대로 가져와서 필요한 데이터를 추출하는 것
- 데이터를 대량 수집하는 기법

데이터 전처리 하기

STEP
02

데이터 수집

- 라벨링된 데이터 필요
- 네이버 뉴스 크롤링(크롤러 이용)


```
from korea_news_crawler.articlecrawler import ArticleCrawler

Crawler = ArticleCrawler()
Crawler.set_category("정치", "IT과학", "economy")
Crawler.set_date_range(2017, 1, 2018, 4)
Crawler.start()
```

데이터 전처리 하기

데이터 셔플

- 데이터 섞기
- 최적의 W값을 찾기 위해서!
  사회 -> 정치 -> 스포츠 카테고리 순서대로 하면 “사회”학습 시
  W를 그대로 정치와 스포츠에 대입시켜버림

데이터 전처리 하기

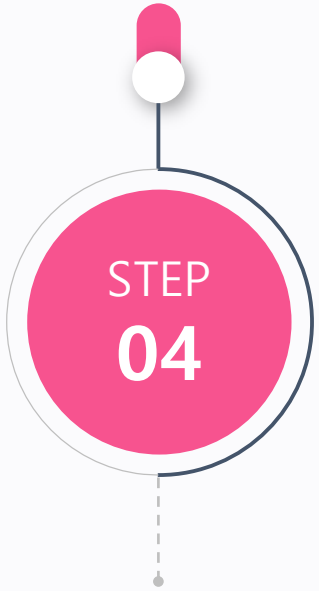
	A	B	C	D	E	F	G	H	I
1		IT과학	아이뉴스2	신한금융투자 핵심시스템 리눅스 전환					
2		생활문화	국민일보	1960년대 이후 한국교회사 교계 최초 집대성					
3		오피니언	경향신문	먹거리 공화국쌀이 죽어간다					
4		정치	뉴스1	통일부 정례 브리핑					
5		세계	뉴스1	대구 북구 침산동 교통사고 현장					
6		오피니언	파이낸셜뉴	데스크칼럼 서열문화와 갑질					
7		생활문화	스포츠경향	오늘의 운세 나침반2018년 4월 11일					
8		사회	아시아경제	경기도 부동산포털사이트 거래·생활정보도 제공한다					
9		사회	MBC	연대 정시 모집 확대...학종 불신 시작					
10		IT과학	ZDNet Kor	"북한 APT 해킹 작년부터 한국 넘어 일본 등으로 확대"					

```
import csv
import os

os.chdir("C:₩₩Users₩₩user₩₩Downloads")

category = ['IT과학','경제','정치','세계','오피니언','사회','생활문화']

file_unity = open('Article_unity.csv', 'w', encoding='euc-kr')
//열어서_저장해라(Article_unity라는 새로운 파일, 쓰기전용으로, 번역_인코딩은euc-kr유니코드한국어로)
wcsv = csv.writer(file_unity)
//WriteCSV에=써넣어라(file_unity를)
count = 0

for category_element in category:
        //위에 IT과학 경제 정치 ... 를 한 요소씩 반복
    file = open('Article_'+category_element+'.csv','r',encoding='euc-kr',newline="")
        //file에 = 열어서_저장해라(형식이 'Article_정치.csv'인 애를, 읽기전용으로,번역_인코딩은euc-kr유니코드한국어로, newline에는 빈 벡터
    line = csv.reader(file)
        //line에 = 읽어들여라(file을 즉:정치 기사리스트를 한 줄씩)
    try :
        for line_text in line:
                //line을 한 줄씩 반복(리스트를 반복)
            wcsv.writerow([line_text[1], line_text[2], line_text[3], line_text[4]])
                //아까 비어있던 file_unity를 넣은 WCSV에.한줄씩써넣어라(엑셀의[카테고리], 엑셀의[뉴스], 엑셀[본문] )
    except:
        pass
```

데이터 전처리 하기

```
import csv
import random
import os

os.chdir("C:₩₩Users₩₩Juwon₩₩PycharmProjects₩₩tensorflows₩₩parser₩₩Csv") // Csv가 있는 경로 설정

file = open('Article_unity.csv', 'r', encoding='euc-kr')
line = file.readlines()
random.shuffle(line)
rcsv = csv.reader(line)

file_write = open('Article_shuffled.csv', 'w', encoding='euc-kr', newline="")
wcsv = csv.writer(file_write)

for i in rcsv:
    try:
        wcsv.writerow([i[0].strip(), i[1], i[2], i[3]])
    ////아까 비어있던 file_unity를 넣은 WCSV에.한줄씩써넣어라(엑셀의 [날짜].₩n지우고, 엑셀의[카테고리], 엑셀의[뉴스], 엑셀[본문] )
    except:
        pass
```

데이터 전처리 하기



데이터 가공

- 형태소 분석 & Word2Vec
- 문장 -> 단어 -> 형태소로 자르기
- 형태소 (word) -> 벡터 (vector) 형태로 바꾸기

데이터 전처리 하기

```
from konlpy.tag import Twitter
from gensim.models import Word2Vec
import csv //comma separated values, txt형 파일확장자

twitter = Twitter()

file = open("Article_shuffled.csv", 'r', encoding='euc-kr')
    //file에=열어서 저장해라("Article_shuffled를")
line = csv.reader(file)
token = []
embeddingmodel = []

for i in line:
    sentence = twitter.pos(i[0], norm=True, stem=True)
        //sentence에 = twitter.내장함수인pos(part of speech=형태소)를(line안에 형태소별로 자르기)
    temp = []
    temp_embedding = []
    all_temp = []
    for k in range(len(sentence)):
        temp_embedding.append(sentence[k][0])
            ///temp_embedding에는 (형태소들 저장)
        temp.append(sentence[k][0] + '/' + sentence[k][1])
            /// temp에는 (형태소[0] / 해당품사[1] 저장)
        //for 문을 돌면 all_temp에는 모든 문장의 형태소[0]/품사[1]가 저장 , embeddingmodel에는 모든 문장의 품사[1]가 저장
    all_temp.append(temp)
        //all_temp에는 temp저장
    embeddingmodel.append(temp_embedding)
        //embeddingmodel에는 temp_embedding 저장
    category_number_dic = {'IT과학': 0, '경제': 1, '정치': 2,'세계':3,'오피니언':4, '사회': 5, '생활문화': 6}
    all_temp.append(category_number_dic.get(category))
        ///all_temp에 카테고리 매핑정보까지 저장
    token.append(all_temp)
print("토큰 처리 완료")
    //여기까지 하면 all_temp에 형태소[0], 품사[1],매핑 정보 저장되어있음
```

```
embeddingmodel = []
for i in range(len(token)): ////index를 돌고
    temp_embeddingmodel = []
    for k in range(len(token[i][0])): ////날짜를 돌고
        temp_embeddingmodel.append(token[i][0][k])
    embeddingmodel.append(temp_embeddingmodel)
        ////돌면서 temp_embeddingmodel에 넣고
// max_vocab size 10000000 개당 1 GB 메모리 차지
embedding = Word2Vec(embeddingmodel, size=300, window=5, min_count=10, iter=5, sg=1, max_vocab_size = 360000000)
embedding.save('post.embedding') //만들어진 사각형 숫자 정보를 -> vector로 바꾸는 모델
```

```
In [20]:

mecab.pos(c[:40])
```

```
[('대한민국', 'NNP'),
 ('헌법', 'NNG'),
 ('유구', 'XR'),
 ('한', 'XSA+ETM'),
 ('역사', 'NNG'),
 ('와', 'JC'),
 ('전통', 'NNG'),
 ('에', 'JKB'),
 ('빛나', 'VV'),
 ('는', 'ETM'),
 ('우리', 'NP'),
 ('대한', 'VV+ETM'),
 ('국민', 'NNG'),
 ('은', 'JX'),
 ('3', 'SN'),
 ('·', 'SC'),
 ('1', 'SN'),
 ('운동', 'NNG'),
 ('으로', 'JKB')]
```

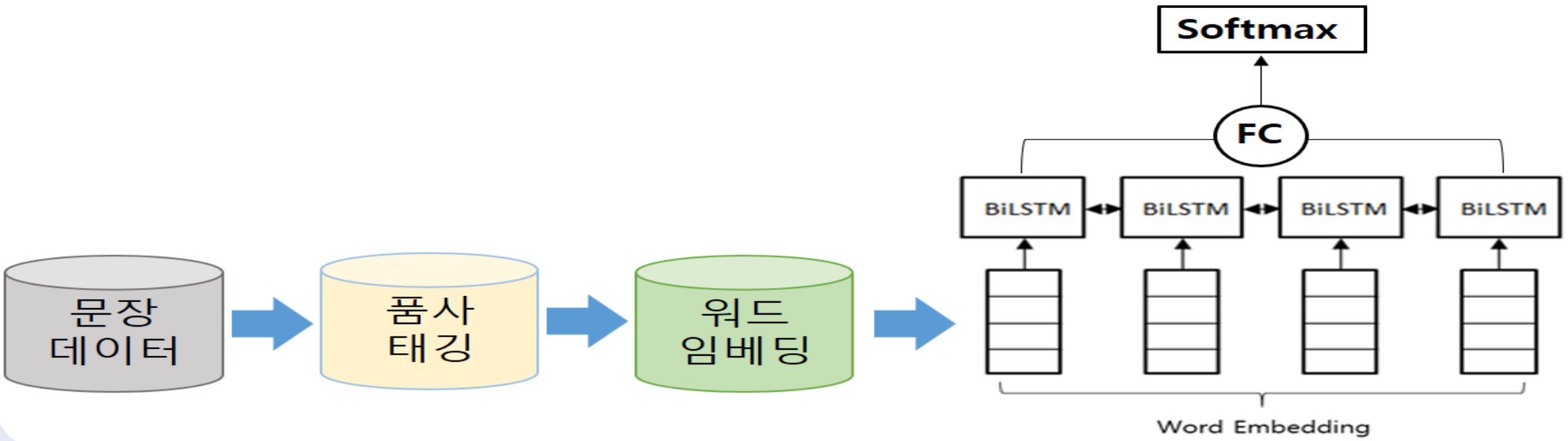
학습 *컴퓨터 학습시키기*

Bi_LSTM 모델

자연어처리 모델 중 하나

예시

학습된 결과물 예시



학습 *컴퓨터 학습시키기*

```
import time
import tensorflow as tf
import numpy as np
import Bi_LSTM as Bi_LSTM
import Word2Vec as Word2Vec
import csv
from konlpy.tag import Twitter
import os

twitter = Twitter()
W2V = Word2Vec.Word2Vec()

file = open("Article_shuffled.csv", 'r', encoding='euc-kr')
line = csv.reader(file)
token = []
embeddingmodel = []

for i in line: ///line으로 자르기
    content = i[3]  # csv에서 뉴스 제목 또는 뉴스 본문 column으로 변경
    sentence = twitter.pos(i[0], norm=True, stem=True) ///line안에 형태소별로 자르기
    temp = []
    temp_embedding = []
    all_temp = []
    for k in range(len(sentence)):
        temp_embedding.append(sentence[k][0]) ///temp_embedding에는 (형태소들 저장)
        temp.append(sentence[k][0] + '/' + sentence[k][1]) /// temp에는 (형태소[0] / 해당품사[1] 저장)
    all_temp.append(temp)
    embeddingmodel.append(temp_embedding)
    category = i[1]  # csv에서 category column으로 변경
    ///for문 다 돌면 all_temp에는 모든 문장의 형태소[0]/품사[1]가 저장 , embeddingmodel에는 모든 문장의 품사[1]가 저장
    category_number_dic = {'IT과학': 0, '경제': 1, '정치': 2,'세계':3,'오피니언':4, '사회': 5, '생활문화': 6}
    all_temp.append(category_number_dic.get(category)) ///all_temp에 카테고리 매핑정보까지 저장
    token.append(all_temp)
print("토큰 처리 완료")
    ////앞과정과 동일하게 토큰 만들기
```

학습 *컴퓨터 학습시키기*

```
tokens = np.array(token)
print("token 처리 완료")
print("train_data 최신 버전인지 확인")
train_X = tokens[:, 0] ///형태소들
train_Y = tokens[:, 1] ///품사들

train_Y_ = W2V.One_hot(train_Y)  ///품사들도 vec형태로
train_X_ = W2V.Convert2Vec("Data₩₩post.embedding",train_X) ///만들어둔embedingmodel 가져오기

Batch_size = 32
Total_size = len(train_X)
Vector_size = 300
seq_length = [len(x) for x in train_X]
Maxseq_length = max(seq_length)
learning_rate = 0.001
lstm_units = 128
num_class = 12
training_epochs = 5
keep_prob = 0.75
///머신러닝 기본 요소들 설정

X = tf.placeholder(tf.float32, shape = [None, Maxseq_length, Vector_size], name = 'X')
Y = tf.placeholder(tf.float32, shape = [None, num_class], name = 'Y')
seq_len = tf.placeholder(tf.int32, shape = [None])

BiLSTM = Bi_LSTM.Bi_LSTM(lstm_units, num_class, keep_prob)

with tf.variable_scope("loss", reuse = tf.AUTO_REUSE): //공유 변수(W와 B가 너무 많을 때 한번에 관리해줌)
    logits = BiLSTM.logits(X, BiLSTM.W, BiLSTM.b, seq_len) //logistic 회귀 모델을 이용한 BiLSTM
    loss, optimizer = BiLSTM.model_build(logits, Y, learning_rate) //학습 변수로(logit,Y,leanring rate넣기)

prediction = tf.nn.softmax(logits) //softmax : 결과값을 0과 1 사이로 만들어주는 함수
correct_pred = tf.equal(tf.argmax(prediction, 1), tf.argmax(Y, 1)) //correct_prediction : 예측값과 실제라벨링값이 일치하는가?
accuracy = tf.reduce_mean(tf.cast(correct_pred, tf.float32)) //일치하면1을 증가시켜서 accuracy 측정

init = tf.global_variables_initializer() //변수 초기화

total_batch = int(Total_size / Batch_size) //전체 batch_size 결정

print("Start training!")
```

학습 *컴퓨터 학습시키기*

```
modelName = "BiLSTM.model"
saver = tf.train.Saver()


with tf.Session() as sess:

    start_time = time.time()
    sess.run(init)
    train_writer = tf.summary.FileWriter('Bidirectional_LSTM', sess.graph)
    i = 0
    for epoch in range(training_epochs): ///epoch 횟수만큼

        avg_acc, avg_loss = 0. , 0.
        for step in range(total_batch):

            train_batch_X = train_X_[step*Batch_size : step*Batch_size+Batch_size] //+Batch_size만큼씩 늘려가면서 학습
            train_batch_Y = train_Y_[step*Batch_size : step*Batch_size+Batch_size] //+Batch_size만큼씩 늘려가면서 학습
            batch_seq_length = seq_length[step*Batch_size : step*Batch_size+Batch_size]

            train_batch_X = W2V.Zero_padding(train_batch_X, Batch_size, Maxseq_length, Vector_size) //zero_padding적용

            sess.run(optimizer, feed_dict={X: train_batch_X, Y: train_batch_Y, seq_len: batch_seq_length})
            //W 최적화 진행
            loss_ = sess.run(loss, feed_dict={X: train_batch_X, Y: train_batch_Y, seq_len: batch_seq_length})
            avg_loss += loss_ / total_batch
            //loss 함수값 계산

            acc = sess.run(accuracy , feed_dict={X: train_batch_X, Y: train_batch_Y, seq_len: batch_seq_length})
            avg_acc += acc / total_batch
            //accuracy 값 계산
            print("epoch : {:02d} step : {:04d} loss = {:.6f} accuracy= {:.6f}".format(epoch+1, step+1, loss_, acc))

        summary = sess.run(BiLSTM.graph_build(avg_loss, avg_acc))
        train_writer.add_summary(summary, i)
        i += 1

        ////////////시간 정보 알려주는 추가 설명/////////////
    duration = time.time() - start_time
    minute = int(duration / 60)
    second = int(duration) % 60
    print("%dminutes %dseconds" % (minute,second))
    save_path = saver.save(sess, os.getcwd())

    train_writer.close()
    print('save_path',save_path)
```

결과 *두근두근두근두근*

```
WARNING:tensorflow:From C:₩Users₩user₩Anaconda3₩lib₩site-packages₩tensorflow₩python
₩training₩saver.py:1266: checkpoint_exists (from tensorflow.python.training.checkpoi
nt_management) is deprecated and will be removed in a future version.
Instructions for updating:
Use standard file APIs to check for files with this prefix.
INFO:tensorflow:Restoring parameters from C:₩Users₩user₩Downloads₩Bi_LSTM.model
문장을 입력하세요 : [시론] 모두 카산드라가 되어야 한다

C:₩Users₩user₩Anaconda3₩lib₩site-packages₩konlpy₩tag₩_okt.py:16: UserWarning: "Twitt
er" has changed to "Okt" since KoNLPy v0.4.5.
  warn('"Twitter" has changed to "Okt" since KoNLPy v0.4.5.')
C:₩Users₩user₩Anaconda3₩lib₩site-packages₩jpype₩_core.py:210: UserWarning:
--------------------------------------------------------------------------------
Deprecated: convertStrings was not specified when starting the JVM. The default
behavior in JPype will be False starting in JPype 0.8. The recommended setting
for new code is convertStrings=False.  The legacy value of True was assumed for
this session. If you are a user of an application that reported this warning,
please file a ticket with the developer.
--------------------------------------------------------------------------------

  """)

IT과학 15.29 %
경제 32.22 %
정치 0.19 %
세계 0.03 %
오피니언 50.23 %
사회 0.33 %
생활문화 1.71 %

문장을 입력하세요 :
```

마무리 & 평가

A 위 재 연

따라하는 것만으로도 어려운 것 같다.

이론 공부보다는 프로젝트가 배우는 것이 더 많은 것 같다.

A 황 서 현

매번 모델링만 공부했었는데 데이터 처리부터가 머신러닝임을 알게 되었다.

남의 코드를 열심히 읽고 해석해야 하는지 깨달았다.

감사합니다