Chatbot에 활용되는 머신러닝 모델 분석

산업인공지능학과 2021254005 김준태

목차

* 챗봇
* 챗봇에 사용되는 딥러닝 이론
  + NLP
  + CNN
  + RNN ~ LSTM
* 모델 구현 및 테스트
* 결과 및 고찰
* 참고 자료

챗봇

* 챗봇(chatbot) 정의: chatter와 robot의 합성어로 텍스트나 음성을 매개로 사람과 대화를 나눌 수 있는 프로그램
  + 기대 효과: 고객 상담 및 고객 서비스 업무에 도입하여 자동화 및 비용 절감
  + 발전 현황: 과거 다양한 시나리오에 대응할 수 있는 규칙 기반의 프로그램에서 NLP 딥러닝 기술을 도입하여 실제 언어를 이해하고 응답하는 수준의 프로그램으로 발전
* 챗봇 구조
* 챗봇 엔진 핵심 기능
  + 질문 의도 분류 : 화자의 질문 의도를 파악, 분류 모델 활용
  + 개체명 인식 : 화자의 질문에서 단어 토큰별 개체명을 인식
  + 핵심 키워드 추출: 형태소 분석을 통해 의미 해석에 핵심이 되는 단어 토큰 추출
  + 답변 검색 : 답변을 학습 DB로부터 검색
  + 소켓 서버(챗봇 엔진 서버 프로그램) : 클라이언트에서 요청한 질문을 처리

챗봇에 사용되는 딥러닝 이론

* NLP(Natural Language Processing, 자연어 처리)
  + 음성이나 텍스트를 컴퓨터가 인식하고 처리하도록 하는 것
  + Tokenization(토큰화)
    - Token : 자연어에서 일정한 의미가 있는 가장 작은 정보 단위
    - 토큰 단위를 어떻게 정의하느냐에 따라 자연어 처리 성능에 영향을 미침
      * 문장 단위
      * 단어 단위 // 장단점 비교
      * 형태소 단위
        + 형태소 : 의미상 더 이상 쪼개지지 않는 단어
        + KoNLPy 형태소 분석기 비교

Kkma

장점: 분석 품질이 좋음, 지원하는 품사 태그가 가장 많음

단점: 분석 속도가 느림, 사용자 사전으로 추가한 복합 명사에 대해 불완전하게 작동함

Komoran

장점: 자소가 분리된 문장이나 오탈자에 강함, 사용자 사전 관리 용이

단점: 적당한 분석 품질과 분석 속도

Okt

장점: 매우 빠른 분석 속도, 정규화 기능 지원

단점: 사용자 사전 관리 어려움, 용언 분석에 일관성이 부족함

* + 한국어 자연어 처리 라이브러리 KoNLPy
    - 한국어 전처리가 어려운 이유
      * 명사와 조사를 띄어 쓰지 않음
      * 용언에 따라 여러가지 어미가 붙음
      * 어근, 접두사/접미사, 품사 등 다양한 언어적 속성의 구조를 파악해야 함
      * 형태소 단위 토큰화에서 형태소의 뜻과 문맥을 고려한 품사 태깅이 필요
      * 챗봇 엔진은 분리된 형태소와 품사 정보를 활용해 문장의 의미를 분석
  + 단어 사전 구축
    - 기존 구어체나 문어체 이외 인터넷 구어체 역시 분석 가능하도록 새로운 단어나 문장의 단어 사전 구축이 필요
  + Embedding
    - 컴퓨터가 연산을 할 수 있도록 단어나 문장을 수치화하여 벡터 공간으로 표현하는 과정
    - 문장 단위 임베딩
      * 장점 : 전체 문장의 흐름을 파악해 벡터로 변환하기 때문에 문맥적 의미를 반영 가능
      * 단점 : 많은 문장 데이터가 필요하며 학습에 비용이 많이 들어감
    - 단어 단위 임베딩
      * 장점 : 학습 방법이 간단하고 비용이 많이 들어가지 않음
      * 단점 : 동음이의어에 대한 구분을 하지 않기 때문에 문맥적 의미를 구분하지 못함
      * 기법
        + One-Hot Encoding

단어 집합 사전 구축 필요 : 말뭉치에서 나오는 서로 다른 모든 단어 집합

말뭉치에 존재하는 단어의 수가 One-Hot 벡터의 차원을 결정

단어 순서에 의한 인덱스 값을 기반하여 인코딩한 값으로 단어 의미나 유사 단어와의 관계를 담고 있지 않음

단어 사전 크기에 따라 벡터 차원이 커지고 메모리 낭비와 계산의 복잡도가 커짐

* + - * + 분산 표현

희소 표현의 단점을 보완하기 위한 기법

희소 표현보다 적은 차원으로 단어 표현 가능

단어의 의미, 주변 단어 간의 관계 등 다양한 정보를 담을 수 있음

한 단어의 정보가 특정 차원에 표현되지 않고 여러 차원에 분산되어 표현

* + - * + Word2Vec

2013년 구글에서 발표한 가장 많이 사용되고 있는 임베딩 모델

CBOW(Continuous Bag-Of-Words)

주변 단어들을 이용해 타깃 단어를 예측하는 신경망 모델

타깃 단어의 손실만 계산하기 때문에 학습 속도가 빠름

Skip-gram

타깃 단어를 이용해 주변 단어들을 예측하는 신경망 모델

단어 분산 표현력이 우수에 임베딩 품질이 우수함

* + 텍스트 유사도
    - 챗봇이 주로 개발되는 FAQ 응대 업무의 특성 상 입력되는 문장과 시스템에서 해당 주제의 답변에 대한 질문이 얼마나 유사한지 계산할 수 있어야 적절한 답변의 출력이 가능
    - N-gram 유사도
      * 이웃한 n개 단어의 출현 횟수를 통계적으로 표현해 텍스트의 유사도를 계산하는 방법
      * n이 너무 클 경우 비교 문장의 토큰과 비교 시 카운트를 놓칠 확률이 커지고 n이 작을 경우 카운트 확률은 높아지지만 문맥을 파악하는 정확도는 떨어짐, 주로 1~5 사이 값을 사용
    - 코사인 유사도
      * 두 벡터 간 코사인 각도를 이용해 유사도를 측정하는 방법
      * 벡터 크기가 중요하지 않을 때 그 거리를 측정하기 위해 사용
* 의도 분석 ~ CNN 모델
  + CNN(Convolutional Neural Network, 합성곱 신경망)
    - 비전 분야에서 주로 사용
    - 합성곱을 통해 입력 데이터의 특징을 추출
    - Convolution
      * filter(mask, window, kernel 등 다양하게 불림)
        + Stride : 필터의 보폭
        + Filter size와 stride가 출력 크기에 영향을 미침
        + Padding : 입력 feature map의 가장자리에 적절한 행과 열을 추가하여 filter size와 stride에 의해 변하는 출력 크기를 조절
      * Pooling : feature map의 크기를 줄여 과적합을 방지하고 대표값을 통해 주요 특징을 추출
        + Max pooling ~ 최대값 / average pooling ~ 평균값
* 개체명 인식 ~ RNN ~ LSTM 모델
  + RNN(Recurrent Neural Network, 순환 신경망)
    - 은닉층 노드 출력값을 출력층과 그 다음 시점 은닉층 노드의 입력으로 전달하여 순환하는 특징을 가짐
    - Many-to-one, one-to-many, many-to-many와 같이 입출력 길이 조절 가능
  + LSTM(Long Short Term Memory) 모델
    - RNN의 입력 시퀀스 time step이 길어질수록 입력과 출력 간 연관 관계가 줄어들어 생기는 장기 의존성(Long-Term Dependency) 문제를 보완하기 위해 개발된 모델
    - 구조
      * Input gate : 현재 시점의 정보를 얼마나 기억할지 결정
        + sigmoid, tanh 활용
      * Forget gate : 이전까지 기억된 정보를 얼마나 삭제할지 결정
        + sigmoid 활용
      * Output gate : 현재 시점의 은닉 상태 결정
        + tanh 활용
  + Bi-Direction
    - 시퀀스나 시계열 데이터 처리에 특화된 RNN 계열 신경망에 데이터 입력의 역방향 처리를 더하여 양방향의 정보를 전달하고 학습시킴

모델 구현 및 테스트

* 챗봇 엔진 구현 과정
  + 챗봇 엔진 처리 과정 : 발화 입력 - 전처리 – 발화 의도 분석 - 개체명 인식 - DB 내 답변 검색 - 답변 출력
  + 적은 데이터셋과 부족한 모델 성능을 보완하기 위해 룰 베이스 알고리즘과 딥러닝 모델을 같이 사용
  + 전처리 : 형태소 분석기 komoran 활용
    - 불용어로 사용할 품사 설정 후 POS tagger 활용 제거
      * 품사 정보 언급
    - 단어 사전 구축
      * 사전 구축에 사용한 데이터 : corpus.txt
        + corpus.txt

label : 0~인사, 1~욕설, 2~주문, 3~예약, 4~기타

label별 데이터 개수 등 확인?

구조 : “0000 \t 문장 \t label” 형태로 line별로 데이터 작성되어 있음

\t으로 헤더, 문장, label 구분

헤더 부분인 0000 제거

문장 : 전처리기 활용 형태소 분석 및 품사 태깅

태깅된 품사에 근거하여 키워드 추출

추출한 키워드에 인덱스를 붙여 단어 사전 구축

* + 의도 분석: 1D convolution CNN을 사용한 분류 모델
    - 학습 데이터 분석 ~ total\_train\_data.csv
      * Column : query, intent
      * Intent label : 0~인사, 1~욕설, 2~주문, 3~예약, 4~기타
        + Label 별 데이터 개수 등 확인?
      * 전처리
        + komoran 형태소 분석기에 tsv 파일로 사용자 사전 추가
        + 전처리 객체를 활용하여 단어 시퀀스 벡터를 생성하고 패딩을 통해 벡터 크기를 동일하게 맞춤
    - CNN 모델 구조
      * Embedding layer
      * Convolution layer
      * Concatenate
      * Fully connected layer
    - Hyper parameter
      * Drop\_out 비율 : 0.3
      * Epoch : 30
      * Optimizer : Adam
      * Learning rate : 0.0005
      * Loss function : sparse categorical crossentropy
    - 결과
      * Accuracy : 63.89
      * Confusion matrix 결과 해석
  + 개체명 인식(NER, Named Entity Recognition): Bi-LSTM을 사용한 예측 모델
    - 학습 데이터 분석 ~ ner\_train.txt
      * 주요 개체명 label : B\_FOOD~음식, B\_DT~날짜 및 시간, B\_PS~사람, B\_OG~조직 및 회사, B\_LC~지역
    - BI-LSTM 모델 구조
      * Embedding layer
      * Bidirectional LSTM
      * TimeDistributed Dense layer
        + 이거 뭔지 확인
    - Hyper parameter
      * Dropout 비율 : 0.5
      * Recurrent\_Drop\_out 비율 : 0.25
      * Epoch : 10
      * Optimizer : Adam
      * Learning rate : 0.01
      * Loss function : categorical crossentropy
    - 결과
      * 정확도 : 98.4
      * Classification report 결과 해석
      * Confusion matrix 그려보기
  + 답변 검색 ~ 검색 알고리즘 관련 이론은 굉장히 방대함
    - Rule Base의 SQL 구문 형태로 실습

결과 및 고찰

* 각 모델의 성능을 높이기 위해서는 데이터 셋 구축이 무엇보다 중요함
  + 단어 및 문장에 대한 정확한 라벨링 필요
  + 각 시나리오 및 도메인 지식에 알맞게 수집된 데이터를 통한 학습
  + 학습을 위해 충분한 양의 데이터 확보 필요
* 효율적인 검색 알고리즘의 필요성
  + 짧은 시간 내 답변을 제공하려면 효율적인 알고리즘이 필요함
  + 사용된 딥러닝 모델이 무거울 경우 검색 속도에서 앱의 성능이 좌우될 수 있음
* 답변 DB 모델링 및 구성의 중요성

참고 자료

* 처음 배우는 딥러닝 챗봇, 조경래, 한빛미디어