Chatbot에 활용되는 딥러닝 모델 분석

산업인공지능학과

2021254005 김준태

목차

* 챗봇
* 챗봇에 사용되는 딥러닝 이론
  + NLP
  + 의도 분석: CNN 모델
  + 개체명 인식: RNN – LSTM 모델
* 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 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
    - 토큰화 : 주어진 말뭉치(corpus)를 토큰으로 나누는 작업
      * 토큰화 단위 : 문장 단위 토큰화 / 단어 단위 토큰화 / 형태소 단위 토큰화
      * 형태소 단위 토큰화
        + 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

=> 한국어를 영어에서의 단어 단위 토큰화와 같은 효과를 얻으려면 형태소 단위로 토큰화해야 함

* + - * + 형태소: 의미상 더 이상 쪼개지지 않는 단어

자립 형태소: 그 자체로 의미를 갖고 있어 자립하여 사용할 수 있는 형태소로 명사, 대명사, 수사, 관형사, 부사, 감탄사 등이 있음

의존 형태소: 다른 형태소와 결합하여 사용되는 형태소로 접사, 어미, 조사, 어간 등이 있음

* + - * + KoNLPy(한국어 자연어 처리 라이브러리) 형태소 분석기 비교 ~ 표

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

* + 품사 태깅(Part-Of-Speech Tagging)
    - 해당 단어가 어떤 품사로 쓰였는지 구분하는 것이 단어의 의미를 파악하기 위한 주요 지표가 될 수 있음
  + 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
    - Padding: 자연어 처리 중 각 문장의 길이를 동일하게 맞춰주어 하나의 행렬로 만들면 병렬 연산을 통해 한꺼번에 처리가 가능
    - 임베딩 기법
      * One-Hot Encoding
        + 단어 집합의 크기를 벡터의 차원으로 하고 표현하고 싶은 단어의 인덱스에 1의 값을 부여하고 나머지에는 0을 부여하는 가장 기본적인 임베딩 기법

단어 집합 : 말뭉치에서 나오는 서로 다른 모든 단어들을 모아놓은 집합

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

말뭉치에 존재하는 단어의 수가 One-Hot 벡터의 차원을 결정

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

* + - * LSA(Latent Semantic Analysis, 잠재 의미 분석)
        + 빈도 수만으로 표현된 단어 행렬을 특이값 분해(SVD, Singular Value Decomposition)를 통해 차원을 축소시키고 단어들의 잠재적인 의미를 표현하는 새로운 행렬로 만듦
        + 행렬에 새로운 데이터를 추가하려면 처음 빈도 수 행렬을 만들어 특이값 분해 계산을 처음부터 다시 수행해야 한다는 단점이 있음
      * 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

* + - * Glove
        + 2014년 스탠포드 대학에서 개발한 단어 임베딩 모델
        + 카운트 기반과 예측 기반을 모두 사용하여 LSA와 Word2Vec의 단점을 보완하기 위해 개발됨
        + 임베딩된 중심단어와 주변 단어 벡터의 내적이 전체 코퍼스에서 동시
        + 등장 확률이 되도록 만드는 것

-유사도는 쓴 거 없지 않나?

* + 텍스트 유사도
    - 챗봇이 주로 개발되는 FAQ(Frequently Asked Questions) 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
  + 개체명 인식을 위한 BIO 표현
    - Begin-Inside-Outside의 약자로 개체명이 시작되는 부분에 B, 개체명 내부 부분에 I, 개체명이 아닌 부분에 O를 Tagging
* 의도 분석: CNN 모델
  + CNN(Convolutional Neural Network, 합성곱 신경망)
    - 비전 분야에서 주로 사용
    - 1D Convnet의 경우 텍스트 분류나 시계열 예측, 오디오 생성, 기계 번역 분야에서 좋은 성능을 보여준 사례가 존재함
    - 주요 용어 및 개념
      * Convolution(합성곱): 일정 영역 값들에 대해 가중치를 적용하여 하나의 값을 만드는 연산, 연산을 통해 입력 데이터의 특징을 추출
      * filter(= mask, window, kernel): 설명 추가
      * Stride: 필터를 다음 합성곱 연산을 위해 이동시키는 칸 수
        + Filter size와 stride가 출력 크기에 영향을 미침
      * Feature map: 합성곱 적용 결과로 만들어지는 행렬, 합성곱 filter에 표현된 특징을 대응되는 위치에 포함하고 있는 정도를 원소값으로 가짐
      * Padding: 입력 feature map의 가장자리에 적절한 행과 열을 추가하여 filter size와 stride에 의해 변하는 출력 크기를 조절
      * Pooling: 일정 크기의 블록을 통합하여 하나의 대표값으로 대체하는 연산
        + feature map의 크기를 줄여 과적합을 방지
        + 대표값(최대값을 활용한 Max Pooling, 평균값을 활용한 Average Pooling을 주로 사용)을 통해 일정 영역에서 나타나는 특징들을 결합하거나 위치 변화에 강건한 특징을 추출

Convolution으로 추출된 특징은 평행 이동 불변성(translation invariant)를 갖는다고 표현되는데 이는 사람이 물체를 시각적으로 보았을 때 그 물체가 평행 이동하여 다른 곳에 위치하더라도 같은 특징을 갖는 물체라고 인식하는 것과 같은 원리라고 할 수 있음

* + - 구조
      * 특징 추출을 위한 convolution layer
      * 추출된 특징을 활용하여 분류 및 회귀 결과값을 출력하는 Fully connected layer
* 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

* 실습용 챗봇 기능 : 음식 주문 예약 챗봇
* 챗봇 엔진 구현 과정
  + 챗봇 엔진 처리 과정: 발화 입력 - 전처리 – 발화 의도 분석 - 개체명 인식 - DB 내 답변 검색 - 답변 출력
  + 전처리 객체 생성 및 모듈화
    - 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
    - 전처리: konlpy 형태소 분석기 komoran 및 파이썬 내부 함수 활용
      * \t으로 헤더, 문장, label 구분
      * 헤더 부분인 ‘0000’ 제거
      * Komoran 활용 품사 태깅 후 불용어를 제거하여 키워드가 될 단어만 남김
        + 불용어로 사용할 품사 설정

불용어 처리한 품사

관계언 – 주격조사(JKS), 보격조사(JKC), 관형격조사(JKG), 목적격조사(JKO), 부사격조사(JKB), 호격조사(JKV), 인용격조사(JKQ), 보조사(JX), 접속조사(JC)

기호 – [마침표, 물음표, 느낌표](SF), [쉼표, 가운뎃점, 콜론, 빗금](SP), [따옴표, 괄호표, 줄표](SS), 줄임표(SE), 붙임표(SO)

어미 – 선어말 어미(EP), 종결 어미(EF), 연결 어미(EC), 명사형 전성 어미(ETN), 관형형 전성 어미(ETM)

접미사 – 명사 파생 접미사(XSN), 동사 파생 접미사(XSV), 형용사 파생 접미사(XSA)

* + - * 키워드로 남은 단어들에 인덱스를 붙여 단어 사전 구축
      * 구축된 단어 사전을 활용한 전처리 객체 생성
        + 문장을 형태소 단위로 Token화
        + 구축한 단어 사전에 포함된 경우 Token에 포함
      * 단어 사전화 테스트
        + 입력 문장 : “오늘 오후 5시 30분에 닭고기를 먹고 싶어 ㅎㅎㅎ”
        + 처리 결과 : 오늘~40, 오후~155, 5시~1, 30분~1, 닭고기~210, 먹~233, 싶~11, ㅎㅎㅎ~10928
  + 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
        + 총 row 개수 : 156,658  
          인사 : 121  
          욕설 : 1,590  
          주문 : 39,061   
          예약 : 53,086  
          기타 : 11,800
      * 전처리
        + 이전 단계에서 생성한 전처리 객체 호출

단어~인덱스 사전과 형태소 전처리 간 사용자 정의 단어 사전 활용

* + - * + 전처리 객체를 활용하여 단어 시퀀스 벡터를 생성하고 패딩을 통해 벡터 크기를 동일하게 맞춤
      * 7:2:1 비율로 trian set :validation set : test set 나눔
    - CNN 모델 구조
      * Model summary 내용 첨부
    - Hyper parameter ~ 체크 후 작성
      * Drop\_out 비율: 0.3
      * Epoch: 30
      * Optimizer: Adam
      * Learning rate: 0.0005
      * Loss function: sparse categorical crossentropy
    - 결과
      * Accuracy:
      * 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
      * O: 9568  
        B\_PS: 1487  
        B\_OG: 1144  
        I: 792  
        B\_LC: 698  
        B\_FOOD: 239  
        B\_DT: 229  
        B\_TI: 17
    - BI-LSTM 모델 구조
      * Embedding layer
      * Bidirectional LSTM
      * TimeDistributed Dense layer
        + Many-to-Many model
    - 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 그려 보기
  + 답변 검색
    - 분류된 의도명과 인식된 개체명을 기반으로 쿼리문을 생성하여 답변을 검색하는 모듈 생성
    - 답변 DB 구조 기술

결과 및 고찰

* 단어 사전 및 양질의 학습 데이터 구축 필요
  + 챗봇의 경우 사용 분야 및 목적을 명확하게 정의하고 사용 범위를 좁혀야 정확하고 일관성 있는 서비스 제공이 가능하다고 알려져 있음
  + 챗봇 적용 분야에서 주로 사용되는 용어 및 프로세스, 도메인 지식 등을 반영하여 명확한 기준으로 labeling된 단어 사전 및 학습 데이터가 신경망 모델 성능을 좌우함
* 전처리 및 임베딩 모듈 성능 검증 및 개선
  + 구축된 학습 데이터를 신경망이 학습할 수 있도록 전처리하고 벡터화, 수치화하는 임베딩 결과 역시 신경망 모델 성능을 좌우함
* 신경망 모델 구조 개선
  + 발화 의도 분류 및 개체명 인식에 사용되는 신경망의 성능을 높여야 정확한 답변 제공 가능
  + 자연어 관련 신경망 모델의 대부분 무거운 편이기 때문에 정확도뿐만 아니라 효율성을 가진 모델 구성이 필요
* 효율적인 DB 모델링
  + 적절한 답변을 검색하여 보여줄 수 있도록 효율적인 DB 모델링이 필요
  + 쿼리 성능이 답변 제공 속도를 좌우할 수 있을 거라 생각됨

참고 자료

* 처음 배우는 딥러닝 챗봇, 조경래, 한빛미디어
* 케라스 창시자에게 배우는 딥러닝, 프랑스와 숄레, 길벗
* 딥러닝을 이용한 자연어 처리 입문, 유원준, https://wikidocs.net/book/2155