서 동 수 교수 지도

석사학위 청구논문

멀티 도메인 상의 확장성있는 대화상태 트래킹 연구

A Study For Scalable Dialog State Tracking in Multi-Domain

2020

성신여자대학교 대학원

컴퓨터학과

이 민 주

멀티 도메인 상의 확장성있는

대화상태 트래킹 연구

A Study For Scalable Dialog State Tracking in Multi-Domain

서 동 수 교수 지도

이 논문을 석사학위논문으로 제출함

2020년 5월

성신여자대학교 대학원

컴퓨터학과

이 민 주

인 준 서

이민주의 석사학위 논문으로 인준함

2020년 5월

심사위원장 ..... (서명 또는 인)

심 사 위 원 ..... (서명 또는 인)

심 사 위 원 ..... (서명 또는 인)

성신여자대학교 대학원

논 문 개 요

목적 지향 대화 시스템에서 대화 상태 트래킹은 사용자의 의도를 모니터링하고 대화 상태라는 정보를 추출하여 시스템의 액션을 결정하는데 관여하는 모듈이다. 목적 지향 대화 시스템을 이용한 서비스가 점점 다방면에서 제공되면서 대화 상태 트래킹도 멀티 도메인을 지원하는 것이 필수적인 것이 되었다.

기존의 모든 슬롯과 값을 담은 온톨로지에 의존하는 상태 트래킹은 여러 도메인으로 확장할 경우 학습 데이터를 구축하는데 많은 비용이 들며 복잡도가 증가하는 문제가 있다. 따라서 멀티 도메인 상태 트래킹을 구축하기 위해서는 온톨로지에 의존하지 않고 대화 데이터 자체를 학습하는 방법을 적용하는 것이 중요하다. 대화 데이터와 같은 텍스트 데이터는 머신러닝에서 시퀀스 모델링 문제로 여겨진다. 따라서 데이터에서 시퀀스 정보를 잘 추출하는 모델을 구축하는 것이 중요하다.

텍스트 데이터에 사용되는 주된 신경망은 순환신경망과 합성곱 신경망이지만 최근에는 셀프 어텐션 메커니즘을 이용한 트랜스포머와 BERT가 기계 독해와 기계 번역에서 좋은 성능을 보여주었다. 본 논문에서는 셀프 어텐션 메커니즘과 순환신경망을 이용한 모델을 구축하여 대화 데이터로부터 대화 상태를 트래킹하는 실험을 진행하고 기존의 순환신경망만을 이용한 모델과 정확도를 비교하여 셀프 어텐션의 효과를 알아본다. 그리고 대화 상태 트래킹에서 문제로 지적되는 Out-of-Vocabulary 문제를 해결하기 위해 사전 훈련된 단어 임베딩과 복사 메커니즘을 사용한다. 모델은 자연어 이해(Natural Language

Understanding) 모듈을 거치지 않고 대화에서 바로 슬롯을 트래킹하는 end-to-end 방식으로 개발된다. 실험에서는 기본 구조인 인코더-디코더 모델부터 제안 모델까지 여러 버전의 모델에 대한 테스트를 진행하고 실험을 통해 성능을 측정해본다.

목 차

표 목 차

그 림 목 차

Ⅰ. 서 론

1. 연구의 배경 및 목적

최근 기계번역, 음성인식 등의 다양한 인공지능 분야에서 신경망의 심층 학습으로 인해 큰 성능향상을 가져왔다. 특히 신경망 기반의 자동번역의 성공은 사용자의 요구에 대한 시스템의 응답을 문장으로 자동 생성하는 신경망 기반의 대화처리 기술 연구를 활발하게 하였다. 목적 지향 대화 시스템(Task-oriented Dialog System)은 단순한 응답이 목적인 Chit-Chat 대화 시스템과는 다르게 사용자가 이루고자 하는 목표가 있고 시스템은 사용자가 원하는 정보를 찾아 전달하는 것을 목적으로 한다. 목적 지향 대화 시스템은 [그림 1]과 같이 특정한 목적을 수행할 수 있는 시스템으로 다양한 도메인에서 사용되고 있다.

System Action Templates
hello what can I help you with today any preference on a type of cuisine
api_call <*price*> <*number of people*> <*cuisine*> <*location*>
great let me do the reservation
here it is <*info_address*>
here it is <*info_phone*>
how many people would be in your party
I'm on it is there anything i can help you with
ok let me look into some options for you
sure is there anything else to update
sure let me find an other option for you
what do you think of this option: <*restaurant*>
where should it be which price range are looking for
you're welcome

[그림 1] 식당 예약 관련 시스템 액션 템플릿 예

대화 상태 트래킹(Dialog State Tracking, DST)은 목적 지향 대화 시스템에서 사용자의 의도, 슬롯, 값을 추적하는 모듈이다. 대화가 진행됨에 따라 대화의 상태(State)를 나타내는 사용자의 의도, 사용자가 원하는 정보에 관련된 슬롯이 지속적으로 변경된다. 이 상태를 바탕으로 시스템은 다음 행동을 결정하게 되므로 대화 상태 트래킹은 전체적인 대화 시스템 성능에 많은 영향을 미치는 중요한 요소이다.

대화 상태 트래킹 시스템은 지속적으로 변하는 상태를 찾아내야 하며 또한 여러 도메인에 걸쳐 이뤄지는 대화와 새롭게 추가되는 도메인과 슬롯을 처리할 수 있도록 확장성 있는 시스템으로 만들어져야 한다. 그러나 사전에 모든 데이터가 구축된 온톨로지(Ontology)를 확보하는 것은 어려울 뿐만 아니라 데이터들이 매우 가변적이어서 수시로 업데이트된다. 따라서 슬롯과 값이 고정된 온톨로지를 이용해 분류(classification)를 하거나 슬롯-값 쌍의 후보 세트(Candidate set)를 만들고 점수를 매기는 방식은 제한적이다.[1] 본 논문에서는 이러한 문제에 대한 대안으로 자연어 이해 모듈을 거치지 않고 대화 데이터 자체를 입력으로 받는 종단형(end-to-end) 방식을 사용하여 대화 내에서 도메인과 슬롯, 값을 추적하고 고정된 온톨로지를 사용하지 않는 멀티 도메인 대화 상황 트래킹 시스템을 개발하고자 한다.

최근에 많은 연구들이 순환신경망과 합성곱 신경망을 어텐션 메커니즘과 함께 사용하거나 대체하는 모델들을 제안하고 있다[2, 3]. 순환신경망은 입력을 순차적으로 처리하는 구조의 특성상 문장의 멀리 떨어진 정보들은 소실되고 마지막 몇 단어만 기억하는 장기 의존성 문제(long-term dependencies)를 갖고 있다. 어텐션 메커니즘은 이러한 문제를 해결할 수 있음을 보여주었고 기계 번역과 기계 독해에서 좋은 성능을 나타냈다. 대화 시스템에서는 슬롯 채움(Slot-filling), 의도 탐지(Intent Detection) 등에서 적용되어 순환신경망의 성능을 높이기 위해 사용되었다.

[2]에서 제안된 셀프 어텐션 메커니즘은 어텐션을 다른 시퀀스가 아닌 현재 시퀀스에 대해 수행하는 방법으로 장·단기 기억을 보존해야하는 여러 시퀀스 문제에 대해 효과를 보였다. 입력 시퀀스 내의 토큰 간의 위치 관계를 이용해 시퀀스를 모델링하고 긴 시퀀스를 처리하는 능력이 우수함을 나타냈다.

본 논문에서는 셀프 어텐션 메커니즘을 기존 인코더-디코더 모델과 결합하여 긴 대화를 효과적으로 인코딩하고 대화 상태를 예측하는 모델을 제안한다. 그리고 셀프 어텐션의 입력 시퀀스에서 토큰들의 위치 정보를 제공하는 절대적·상대적 두 가지 위치 인코딩 방법을 적용해 모델의 성능을 비교해본다. 또한 멀티 도메인 상태 트래킹을 위해 온톨로지의 고정된 슬롯-값을 사용하지 않고 복사 메커니즘(Copy Mechanism)을 이용해 대화에서 처음 나타나는 단어에 대해 대처할 수 있는 유연한 시스템을 만드는 데 중점을 둔다.

2. 논문의 구성

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 통해 대화 상태 트래킹과 멀티 도메인에서의 활용 현황에 대해 알아보고 3장에서는 대화 처리 모델의 기본구조와 셀프 어텐션 메커니즘을 설명한다. 4장에서는 본 논문에서 집중하는 대화 상태 트래킹의 개념과 활용 현황에 대해 알아본다. 5장에서는 제안 모델과 실험에 필요한 학습 데이터에 대한 설명을 하고 모델의 특성에 맞춰 실험 설정을 한다. 6장에서는 5장에서의 실험 결과를 제시하여 모델의 구조에 따른 정확도를 비교하고 7장에서는 결론 및 향후 계획을 통해 발전 방향에 대해 논의를 하고 마무리 한다.

Ⅱ. 관련 연구

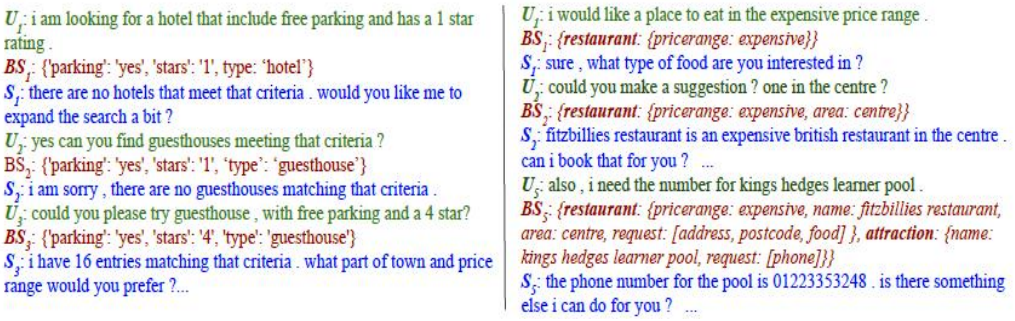
대화 상태 트래킹은 대화 시스템의 핵심기능이다. 대화 상태 트래킹의 목적은 대화 내에서 사용자의 의도를 모니터링하고 대화 상태라는 컴팩트한 대화 정보를 추출하여 정책 모듈로 전달하는데 있다. 초기의 대화 상태 트래킹 연구들은 자연어 이해 모듈과 대화 상태 트래킹 모듈을 연결한 구조를 사용해 트래킹 모듈이 대화가 끝날 때 까지 매 턴마다 대화 상태를 업데이트했다. 그리고 상태 트래킹을 분류 문제로 보고 모든 슬롯과 값을 포함한 온톨로지를 구축하여 대화 내의 값에 대해 슬롯을 분류하는 방법을 이용했다.

Ramadan[4]은 고정 온톨로지를 사용하여 단어와 사용자 발화의 단어들 간의 유사도를 이용해 여러 도메인에서 서로 다른 슬롯들에 대한 정보가 공유될 수 있도록 했다. Zhong[5]는 각각의 슬롯-값에 대해 이진 분류를 하는 모델인 GLAD[1]를 제안했다. GLAD는 전체 슬롯을 인코딩하는 전역 LSTM과 각 슬롯을 인코딩하는 지역 LSTM을 함께 사용한 인코더를 이용하고 사용자 발화와 시스템의 이전 액션을 입력으로 받아 슬롯과 사용자 발화에 대해 유사도를 계산한다. Nouri, Hosseini-Asl[6]은 GLAD가 인코더가 대화와 시스템의 이전 액션을 모두 학습해야 하는 어려움이 있음을 지적했다. GLAD 모델과 달리 하나의 전역 LSTM만 사용하고 슬롯 타입에 대한 임베딩을 인코더 앞단에 놓았다. 모든 슬롯에서 공유되는 인코더를 사용해 복잡성을 낮추고 성능이 조금 더 개선된 모델인 GCE(Globally-Conditioned Encoder)를 제안했다.

위 연구들의 전제는 모든 슬롯과 값을 포함한 온톨로지를 구축해야한다는 것이다. 싱글도메인 상태 트래킹에서는 이 방법이 좋은 성과를 보였으나 멀티

1) GLAD : Global-Locally Self Attentive encoder

도메인 대화에서는 슬롯과 값이 늘어남에 따라 복잡도가 증가하는 문제가 있었다. 또한 사전에 모든 정보를 포함하는 온톨로지를 구축하는 것은 어려울 뿐더러 학습 과정 중에 나타나지 않았던 단어들에 대해 처리하지 못한다는 문제(Out-of-Vocabulary, OOV)를 발생시킨다.



[그림 2] 멀티 도메인을 다루는 대화 예제
(S : System, U : User, BS : Belief States)

[5, 20, 23]에서는 자연어 이해 모듈과 상태 트래킹 모듈을 통합한 종단형 대화 상태 트래킹 모델을 제안하며 복잡도를 낮추고자 했지만 여전히 고정 온톨로지를 사용하여 처음 나타나는 단어들을 처리할 수 없는 문제가 있었다. 다중도메인 상태 추적이 싱글 도메인의 상태 추적보다 어려운 점은 다중도메인 대화는 [그림 2]의 멀티 도메인 대화에서 볼 수 있듯이 여러 도메인에 걸친 많은 정보들 때문에 싱글도메인 대화에 비해 훨씬 크고 다양한 상태들이 존재해 온톨로지를 구축하는 것이 어렵다는 점이다. 그리고 각 도메인은 도메인 간에 공통된 슬롯들을 공유할 수도 있고, 공유하지 않는 유일한 슬롯을 가질 수도 있어 복잡한 양상을 보인다. 따라서 멀티 도메인 상태 트래킹 연구에서는 기존의 온톨로지 기반의 대화 트래킹의 한계를 극복하여 처음 등장하는 단어를 어떻게 처리할 것인지에 대한 문제가 중요한 과제가 되었다.

대화 상태 트래킹 모델은 크게 고정단어(fixed-vocabulary)기반의 접근과, 개방형 단어기반의 접근으로 나눌 수 있다[7]. 고정단어 기반 방법은 [8, 9]와 같이 미리 확보한 온톨로지를 이용해 슬롯-값 후보에 대해 점수를 매기는 방식이고 개방형 단어기반 방법은 OOV 문제를 해결하기 위해 제안된 것으로 온톨로지에 의존하지 않고 입력 데이터로부터 후보를 생성하는 방법이다. Rastogi[10]는 온톨로지에 없는 단어를 처리하기 위해 대화 내에서 나타나는 단어들에 대해 슬롯과 쌍을 만들어 점수를 매기는 방법을 사용했고 P. Xu[1]는 대화 상태 트래킹 시스템에 입력 시퀀스에서 특정 단어를 복사해 출력으로 내보내는 포인터 네트워크를 도입해 처음 등장하는 단어에 대한 문제를 해결하고자 했다. Wu[31]는 인코더-디코더를 기반하여 입력 대화로부터 복사 메커니즘을 이용해 온톨로지의 단어분포와 현재 대화로부터 단어분포를 결합하여 대화 상태를 생성해내는 모델 TRADE[2]를 제안했다. Goel[7]는 계층적 인코더를 이용해 고정 온톨로지를 사용하는 고정형 단어 기반 방식과 대화 데이터에서 모든 슬롯 값들을 얻는 개방형 단어 기반 방식을 함께 사용해 각 슬롯마다 두 방법 중 더 적합한 방법을 선택해 값을 만들어내는 방법을 사용하여 모델 HyST[3]를 제안했다.

대화 상태 트래킹은 대부분의 연구들이 심층 신경망과 순환 신경망을 중심으로 이뤄지고 있다. 자연어처리에서 Transformer와 BERT 모델이 기계번역에서 좋은 성능을 보이면서 대화 상태 트래킹에도 이용되기 시작했다. Lee[11]는 고정 온톨로지를 이용해 도메인-슬롯 쌍을 쿼리 벡터로, 대화 컨텍스트를 키 벡터로 이용하는 BERT를 이용한 모델인 SUMBT[4]를 제안했다. 그리고 예측 슬롯과 정답 슬롯의 유클리디언 거리를 이용하여 점수를 줄여가는 방식으로 학습했다.

---

2) TRADE : TRAnsferable Dialogue state Generator
3) HyST : A Hybrid Approach for Flexible and Accurate Dialogue State Tracking
4) SUMBT : Slot-Utterance Matchingfor Universal and Scalable Belief Tracking

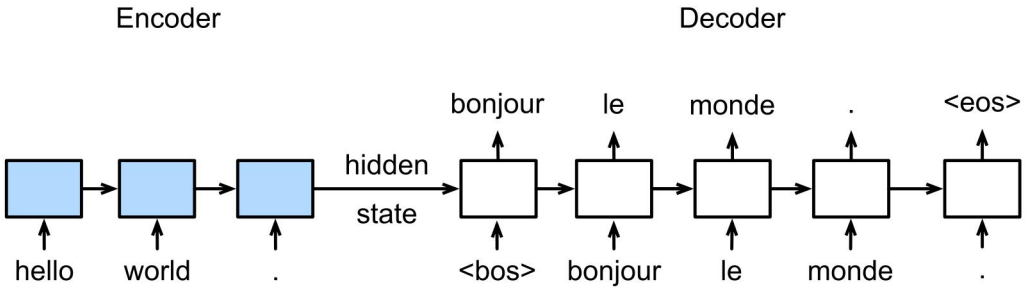
Ⅲ. 대화 처리 모델

1. 인코더-디코더 구조

순환 신경망은 특별히 연속데이터를 처리하기 위해 만들어진 모델로 자연어 처리 연구에서 널리 쓰이고 있다. 순환 신경망은 연속데이터의 각 스텝마다 이전 스텝의 연산과 결과를 바탕으로 같은 일을 반복적으로 한다. 일반적으로 순환 신경망은 각 유닛(unit)에 토큰을 하나씩 주입하면서 연속 데이터를 고정 사이즈의 벡터로 표현해 출력한다. LSTM(Long Short-Term Memory)은 순환신경망의 한 종류로 기존 순환신경망이 시점이 길어질수록 앞의 정보가 뒤로 충분히 전달되지 못하는 현상인 장기 의존성 문제를 해결하기 위해 제안된 모델이다[32]. LSTM은 은닉층의 메모리 셀에 입력 게이트, 망각 게이트, 출력 게이트를 추가하여 불필요한 기억을 지우고, 기억해야할 것들을 정한다. LSTM은 기존 순환신경망에 비교하여 긴 시퀀스의 입력을 처리하는데 탁월한 성능을 보인다. GRU(Gated Recurrent Unit)는 LSTM의 장기 의존성 문제에 대한 해결책을 유지하면서, 은닉 상태를 업데이트하는 계산을 줄여 그 구조를 간단화 시켰다[33]. GRU에서는 업데이트 게이트와 리셋 게이트 두 가지 게이트만 존재한다.

순환 신경망의 이와 같은 구조는 연어 모델링, 기계 번역, 음성 인식, 이미지 캡셔닝 등에 적합해 순환 신경망 기반의 모델들이 자연어 처리에서 지배적인 역할을 하고 있다[12].

인코더-디코더 구조는 순환신경망을 이용한 것으로 대표적으로 시퀀스-투-시퀀스(seq2seq)에 사용된다. seq2seq는 기계 번역, 텍스트 요약, 시계열 데이터 예측 등에서 좋은 활약을 보였다. [그림 3]은 seq2seq 모델을 시각화 한 것으로 입력인 'hello world .'를 처리하여 'bonjour le monde .'를 출력한다. 인코더의 순환신경망 네트워크는 입력 시퀀스를 문장을 표현하는 벡터인 은닉 상태(hidden state)를 만들고 디코더는 이 벡터를 이용해 출력 시퀀스를 디코딩한다.



[그림 3] 인코더-디코더 구조

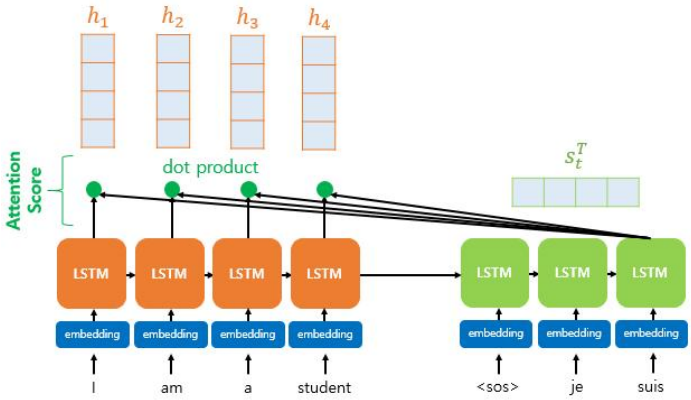
출처 : https://d2l.ai/chapter_recurrent-modern/seq2seq

2. Self-Attention Mechanism

2-1. 어텐션 메커니즘(Attention Mechanism)

어텐션 메커니즘[13]은 기계번역에서 긴 문장을 기억하기 위해 고안된 방법이다. 기존의 시퀀스-투-시퀀스(Sequence-to-Sequence, seq2seq)모델은 전체 문장에 대한 정보를 하나의 고정된 길이의 컨텍스트 벡터로 압축하고 디코더는 이 벡터를 이용해 출력을 만들어낸다. 여기에는 두 가지 문제가 있는데 인코더가 문장에 대한 모든 정보를 하나의 벡터로 압축하기 때문에 정보를 담는데 한계가 있고 병목현상이 일어난다. 또한 학습이 진행됨에 따라 RNN의 전형적인 문제인 기울기 소실로 인해 문장에 대한 정보를 조금씩 잃어버리게 된다.

어텐션 메커니즘은 이를 위한 대안으로 입력 문장이 길어지면 출력 문장의 정확도가 떨어지는 것을 보정해주기 위해 만들어졌다. 어텐션 메커니즘에서는 인코더의 각 은닉 상태에 가중치를 적용해 입력 문장의 은닉 상태들을 동일한 비율로 참고하지 않고, 디코더의 해당 스텝에서 예측해야할 단어와 연관이 있는 입력 단어부분을 집중해서 본다.



[그림 4] seq2seq 구조에서의 어텐션 메커니즘

출처 :https://wikidocs.net/22893

인코더의 각 시점에 따른 은닉 상태를 $h_1, h_2, ... h_n$ 라고 할 때 디코더의 현재 시점 t에서의 은닉 상태를 s_t, t번째 단어를 예측하기 위한 어텐션 값을 a_t라고 정의한다. 그리고 현재 디코더의 시점 t에서 단어를 예측할 때, 인코더의 각각의 은닉 상태와 디코더의 현 시점 s_t가 얼마나 유사한지 판단하는 어텐션 스코어와 s_t와 인코더의 모든 은닉 상태의 어텐션 스코어의 모음 값을 e_t라고 정의하면 다음과 같다.

$$s^T h_i = score(s_t, h_i) \ \ (i \in n)$$

$$e_t = [s^T h_i, ..., s^T h_N]$$

e_t에 소프트맥스 함수를 적용하여, 모든 값을 합하면 1이 되는 확률 분포인 어텐션 분포 α_t를 만든다.

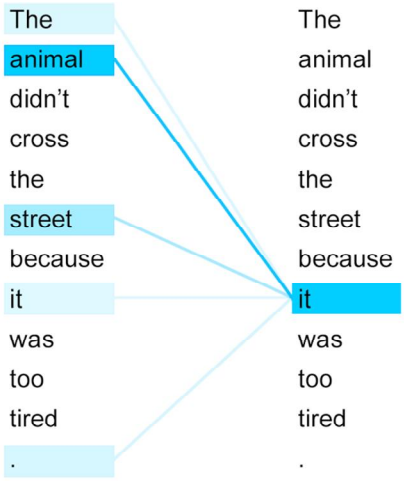
$$\alpha_t = softmax(e^t)$$

$$a_t = \sum_{t=1}^{N} \alpha_i^t h_i$$

어텐션의 최종 결과값을 얻기 위해 각 인코더의 은닉 상태와 어텐션 가중치를 곱하고 더하는 가중합(weighted sum)을 하여 어텐션 함수의 출력 값인 a_t를 구한다. 어텐션 값 a_t는 인코더의 문맥을 포함하고 있어 컨텍스트 벡터(Context Vector)라고 불린다. 디코더에서는 디코딩 시 t시점의 은닉 상태를 어텐션 값 α_t와 결합해 하나의 벡터로 만드는 작업을 수행한다.

2-2. 셀프 어텐션 메커니즘(Self-Attention Mechanism)

인트라-어텐션(Intra-Attention)[1]이라고 알려져있는 셀프 어텐션 메커니즘은 한 문장 내에서 단어들 간의 관계를 표현하는 어텐션 메커니즘이다. 셀프 어텐션은 문장 내의 어떤 단어에 대해 표현할 때 동일한 문장 내에서 서로 다른 위치에 어텐션을 적용해 나타내는 방법으로 기계 독해, 이미지 캡션 생성 등의 분야에서 유용함이 밝혀졌다. Jianpeng Cheng, Li Dong[1]에서는 셀프 어텐션이 기계 독해에 적용되었는데 문장 내에서 현재 단어와 이전 단어들 간의 상관관계를 학습할 수 있음을 보여주었다. 기존 시퀀스-투-시퀀스에서의 어텐션에서는 입력문장과 목표문장이 서로 다른 문장들이고 그 단어들간의 연관성을 찾아냈기 때문에 동일 문장 내에서의 단어들 간의 정보는 얻을 수 없었다. [그림 5]는 문장 'The animal didn't cross the street because it was too tired.'에 대한 셀프 어텐션 예시를 나타낸 것이



[그림 5] 셀프 어텐션 시각화

출처 :https://ai.googleblog.com/

다. 모델은 단어 'it'을 표현하기 위해 문장 내의 'The', 'Street', 'animal' 단어들을 참고한다. 그리고 학습이 진행될수록 'it'이 'The'나 'Street'가 아닌 'animal'과 관련이 높다는 것을 찾아내어 높은 가중치를 부여한다. 셀프 어텐션은 문장 내의 다른 위치들을 보면서 해당 문장을 더 풍부하게 인코딩할 수 있도록 한다.

Vaswani의 "Attention is all you need" [2]에서는 셀프 어텐션을 완전한 모델로써 적용했는데 셀프 어텐션과 피드포워드 레이어만을 이용한 모델인 트랜스포머(Transformer)를 발표했다. 이 논문에서는 셀프 어텐션의 입력이 되는 임베딩에 위치 인코딩(Positional Encoding)을 적용하고 셀프 어텐션을 여러 번 적용하는 멀티헤드 어텐션과 이를 피드 포워드로 연결한 트랜스포머 블록을 만들었다. 그리고 이 블록으로 인코더와 디코더를 만들어 구축한 모델이 트랜스포머이다.

셀프 어텐션은 각 스텝에서 모든 단어에 대해 직접적인 관계를 모델링한다. 즉 위의 예시에서 'it'의 의미를 식별하기 위해 'animal'이라는 단어에 높

은 가중치를 부여해 즉각적으로 참고하는 방법을 배운다. 셀프 어텐션은 매 시퀀스의 토큰마다 적용되면서 입력 시퀀스 $X=(x_{1,}x_{2,}...,x_n)$를 $Y=(y_{1,}y_{2,}...,y_n)$ 표현으로 바꾼다. 여기서 y_i는 x_i에 대한 정보와 x_i가 X내의 다른 위치의 토큰과 얼마나 관련이 있는지에 대한 정보를 통합하여 담고 있다. 다음은 길이가 n인 문장 X에서 단어 x_i의 셀프 어텐션 계산 과정이다. 첫 번째로 시퀀스의 모든 입력 단어벡터 X에 대해 세 가지 가중치 행렬을 만들고 각 단어벡터와 곱해 세 가지 벡터를 만든다. 이 벡터들을 쿼리(Query) 벡터, 키(Key) 벡터, 값(Value) 벡터라 한다.

$$q_i,\ k_j,\ v_j = x_i W^Q,\ x_j W^K,\ x_j W^V\ (i,j \in n)$$

$$e_{ij} = softmax(\frac{q_i k_j^T}{\sqrt{d_k}})$$

$$Attention(q_{i,}k_{i,}v_i) = e_{ij}\ \cdot\ v_i$$

기존의 인코더-디코더에서 어텐션 메커니즘은 입력문장과 목표문장의 서로 다르고 입력문장은 인코더의 입력으로, 목표문장은 디코더의 입력으로 들어갔다. 그리고 인코더의 은닉 상태와 디코더의 은닉 상태를 키 벡터와 쿼리 벡터로 하여 어텐션을 계산했다. 셀프 어텐션에서는 동일 문장에 대해서 쿼리 벡터와 키 벡터를 만들고 어텐션을 계산한다. 값 벡터는 어텐션의 결과 계산된 쿼리와 키의 유사도가 적용된 키 벡터를 나타낸다.

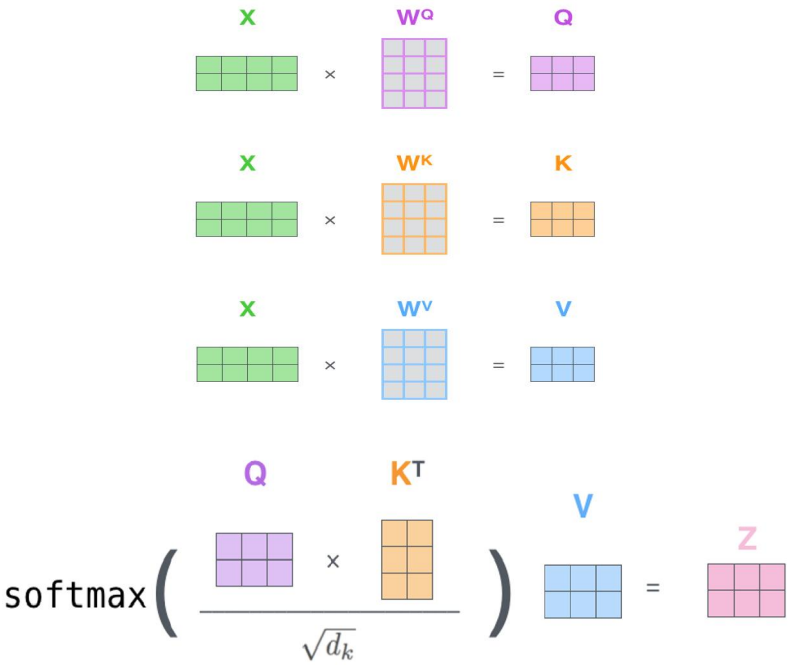
두 번째로 현재 스텝의 단어벡터를 쿼리 벡터, 유사도를 비교할 단어벡터를 키 벡터라고 하고 쿼리 벡터에 대해 시퀀스 내의 자신을 포함한 모든 키 벡터에 대해 유사도를 계산한다. 두 단어벡터의 유사도 e_{ij}는 두 벡터를 내적하고 벡터 차원의 제곱근으로 나눠줌으로써 구할 수 있다. 그리고 여기에

값 벡터 v_j를 곱하여 가중치가 적용된 어텐션 값을 구한다. 이 과정을 행렬 차원에서 진행하면 다음과 같다. 전체 단어벡터 행렬 X에 대해 세 가중치 행렬을 이용해 Q, K, V 행렬을 만든다. 그리고 모든 문장 내의 모든 키 단어벡터에 대해 유사도를 계산한 뒤 이에 대한 소프트맥스를 계산한다. 계산된 어텐션 스코어에 값 행렬을 곱하여 어텐션 값을 얻는다.

$$Q, K, V = XW^Q, XW^K, XW^V$$

$$Attention(Q, K, V) = softmax(\frac{QK}{\sqrt{d_k}})V$$

위 과정을 Scaled Dot-Product라고 한다. 쿼리와 키의 유사도를 나타내기 위해 내적(Dot-Product)를 하고 벡터 차원의 제곱근으로 나눠(Scale) 내적이 너무 커지는 것을 방지한다. 그리고 마지막으로 가중치가 적용된 모든 값 벡터들을 더하면 시퀀스 내의 모든 단어들에 가중치를 적용한 쿼리 단어의 표현을 출력한다. 위 과정을 행렬차원에서 보면 [그림 6]으로 나타낼 수 있다. 행렬 X의 각 행은 단어벡터를 의미한다.



[그림 6] Scaled Dot-Product Attention

출처 : http://jalammar.github.io/illustrated-transformer/

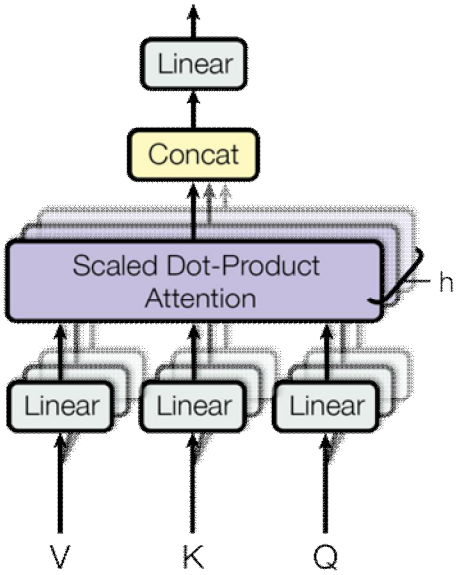
코사인 유사도에서 두 벡터의 내적이 클수록 유사도가 높아지듯이 어떤 쿼리 단어와 키 단어가 특정 일을 수행하는데 중요한 관계에 있다면 모델은 두 단어벡터의 내적을 키우는 방식으로 학습한다. 따라서 내적이 클수록 벡터 공간상에서 가까이에 있을 가능성이 높아지게 된다.

$$softmax(\frac{QK^T}{\sqrt{d_k}})V = \begin{matrix} \\ we \\ can \\ calculate \end{matrix} \begin{matrix} we\ can\ calculate \\ \begin{pmatrix} 0.2 & 0.3 & 0.5 \\ .. & .. & .. \\ .. & .. & .. \end{pmatrix} \end{matrix} \begin{pmatrix} V\,we \\ V\,can \\ V\,calculate \end{pmatrix} =$$

$$\begin{matrix} we \\ can \\ calculate \end{matrix} \begin{pmatrix} 0.2\,V\,we + 0.3\,V\,can + 0.5\,V\,calculate \\ .. \\ .. \end{pmatrix}$$

위 예시는 Scaled Dot-Product Attention을 이용해 쿼리, 키, 값 사이의 관계를 담은 새로운 행렬을 나타낸 것이다. 행렬의 행은 쿼리 단어이고 열이 키 단어일 때 쿼리와 키가 서로 같은 문장인 셀프 어텐션을 나타내고 있으며 계산 결과는 같은 문장 내의 모든 단어 쌍 사이에서 의미적, 문법적 관계를 담고 있다.

2-3. 멀티 헤드 어텐션(Multi-Head Attention)



[그림 7] 멀티 헤드 어텐션

출처 : http://jalammar.github.io/illustrated-transformer/

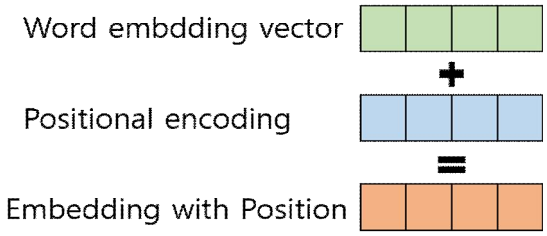
멀티 헤드 어텐션은 여러번의 셀프 어텐션을 통해 입력 데이터의 다양한 측면을 포착한다. 한 번의 셀프 어텐션의 출력 크기가 d_{model}일 때 동일한 입력 시퀀스의 임베딩에 대해 헤드의 개수만큼 셀프 어텐션을 계산하여

d_{model}/h 차원으로 나타낸 후 그 결과를 이어붙인다. 각 헤드의 어텐션 계산은 서로 다른 쿼리, 키, 값 가중치를 갖기 때문에 헤드의 수 만큼 다양한 정보가 저장될 수 있다. [그림 7]과 같이 이 과정이 세트 수 만큼 병렬로 진행되어 멀티 헤드 어텐션이라고 한다. 모두 합쳐진 어텐션 결과들은 피드포워드 레이어를 통해 형상이 맞춰진다. 이 과정을 통해 모델이 입력 시퀀스를 여러 개의 표현으로 나타내고 어느 한 표현에 치우치지 않고 서로 다른 포지션에 어텐션을 할 수 있도록 한다. [2]에서 제안된 트랜스포머는 8개의 헤드를 사용하며 싱글 헤드보다 멀티 헤드일 때 기계 번역에서 좋은 결과를 보였다.

2-4. 문장 내 위치정보

RNN이 자연어처리에서 유용한 이유는 입력에 대한 순차적인 정보를 신경망의 은닉 상태에 저장하기 때문이다. 그래서 같은 단어라 할지라도 그 위치에 따라 은닉 상태의 정보가 달라 입력 시퀀스에서 각 단어가 고유한 출력 표현을 갖도록 보장한다. 하지만 딥러닝 모델은 학습에 많은 데이터를 필요로 하고 모델이 학습하기에 텍스트 데이터가 적절히 가공되거나 라벨링되어 있지 않은 경우가 많아 데이터에 대해 추가적인 정보를 제공해 줄 필요가 있었다. 이후 여러 연구들에서 문장 내 토큰들의 위치를 인코딩한 정보를 함께 이용하는 방법들이 제안되었다. [14]에서는 위치정보를 포함한 어텐션 레이어를 순환 신경망 위에 쌓음으로써 Slot-filling 성능을 개선시킨 바가 있다.

셀프 어텐션만을 사용하는 트랜스포머는 순환신경망을 사용하지 않기 때문에 위치 정보를 추가적으로 통합하는 과정이 더욱 중요하다[15]. 트랜스포머는 단어 임베딩에 절대적 위치 인코딩(Absolute Positional Encoding)을 함으로써 시퀀스의 절대적인 위치에 대한 정보를 나타낸다. 절대적 위치 인코딩은 모델에게 단어에 대한 순서 정보를 주기 위해 삼각함수를 이용해 위치 별로 특정한 패턴을 따르는 위치 인코딩 벡터들을 만드는 것으로 단어 임베딩에 더하여 위치 정보를 통합한 임베딩을 구성한다.



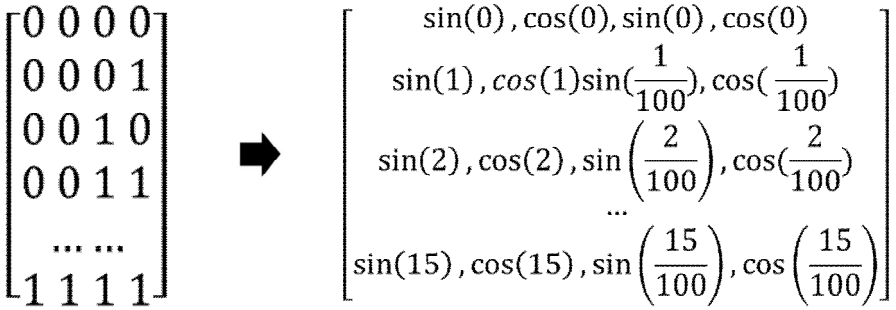
[그림 8] 트랜스포머의 위치 인코딩

각 단어의 임베딩 벡터는 인코더의 입력으로 사용되기 전에 [그림 8]과 같이 위치 인코딩 값들이 더해진다. 트랜스포머는 위치 정보를 가진 값을 만들기 위해 사인 함수와 코사인 함수의 값을 사용하고 이를 임베딩 벡터에 더해줌으로써 단어의 순서 정보를 나타낸다.

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$$

pos는 입력 문장에서 단어의 포지션을 나타내고 i는 임베딩 벡터 내의 인덱스로 $[0, d_{model}/2]$이며 d_{model}는 입력 단어의 차원을 말한다. 인덱스가 짝수인 경우 사인함수, 홀수인 경우 코사인 함수를 사용한다. 위치 인코딩을 계산하기 위해 삼각함수를 사용하는 이유는 다음과 같다. 예를 들어, 길이가 4인 문장을 바이너리로 나타낸다면 [그림 9]처럼 각 자리마다 비트를 바꾸면서 표현할 수 있다. 이것을 실수를 사용해 공간을 절약하면서 매 단어마다 고유한 인코딩을 할 수 있는 방법이 삼각함수를 이용하는 방법이다.

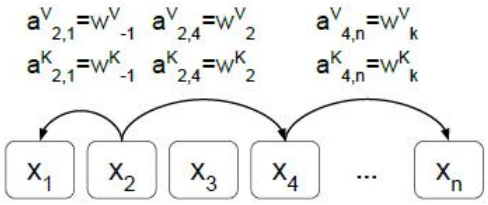


[그림 9] 위치 인코딩 예시

위에서 언급한대로 트랜스포머에서는 순환신경망을 사용하는 것이 아니기 때문에 시퀀스의 순서가 자연스럽게 인코딩되지 않는다. 삼각함수를 이용하면 어떤 길이에 대해서도 $PE_{(pos+k)} = a \cdot PE_{(pos)} + b$ 형태로 나타낼 수 있기 때문에 긴 길이의 문장이 나왔을 때도 적용할 수 있다. 또한 삼각함수의 특성에 따라 서로 다른 위치 사이의 관계도 표현할 수 있다.

Peter Shaw는 [15]에서 입력 시퀀스의 모든 토큰 쌍들의 관계를 고려한 상대적 위치 인코딩(Relative Positional Encoding)을 제안하고 이를 셀프 어텐션에 적용했다. [15]에서는 상대적 위치 인코딩이 두 단어 간의 거리가 너

무 먼 경우 유용하지 않아 최대 상대적 위치(maximum relative position)는 상수 k로 지정하였다. 따라서 두 단어 간의 상대적 거리 관계는 $2k+1$개의 라벨사이에서 정의된다. 입력 문장에서 i, j 번째 단어벡터 x_i, x_j 에 대해 상대적 위치 인코딩을 $a_{ij}^V, a_{ij}^K \in R^{d_k}$ ($d_k = d_{model}/heads$) 라고 정의했다.



[그림 10] 토큰 간 relative position 예시

[그림 10]은 2번째 위치한 단어 X_2와 다른 단어 간의 거리를 나타낸 것으로 토큰 간의 거리인 상대적 위치를 간선으로 나타내고 있다. 모델은 두 토큰 간의 상대적 거리인 w_k^K, w_k^V를 학습을 통해 배우게 된다. 위 내용을 식으로 나타내면 다음과 같다.

$$clip(x,k) = max(-k,\ min(k,x))$$

$$a_{ij}^V = w_{clip(j-i,k)}^K \ , \quad a_{ij}^K = w_{clip(j-i,k)}^K$$

식 (3), (4)에서는 각 단어토큰 간의 상대적 거리를 이용해 셀프 어텐션을 계산하기 위해서 기존 셀프 어텐션 식에 두 항을 추가하였다.

$$\alpha_{ij} = softmax(\frac{q_i(k_j + a_{ij}^K)^T}{\sqrt{d_k}}) \quad (1)$$

$$y_i = \sum_{j=1}^{n} \alpha_{ij}(v_j + a_{ij}^V) \quad (2)$$

식 (1)에서는 쿼리 벡터인 i번째 단어벡터와 키 벡터인 j번째 단어벡터에 대해 Scaled dot product를 하는 과정으로 키 벡터인 j번째 단어벡터에 에 두 단어 간의 상대적 거리 인코딩 a_{ij}^K를 더해 위치정보를 통합한다. 식 (2)는 i번째 단어벡터를 표현하기 위해 값 벡터 부분에 두 단어의 상대적 위치 인코딩을 더해준다. 저자는 상대적 위치 임베딩을 기존 키 벡터와 값 벡터에 더해줌으로써 상대적 위치 정보를 주입할 수 있다고 한다. [15]에서는 절대적 위치 인코딩을 없애고 상대적 위치 인코딩을 셀프 어텐션에 적용해 기계번역 실험을 한 결과 기존 트랜스포머보다 조금 더 높은 BLEU점수를 얻었다.

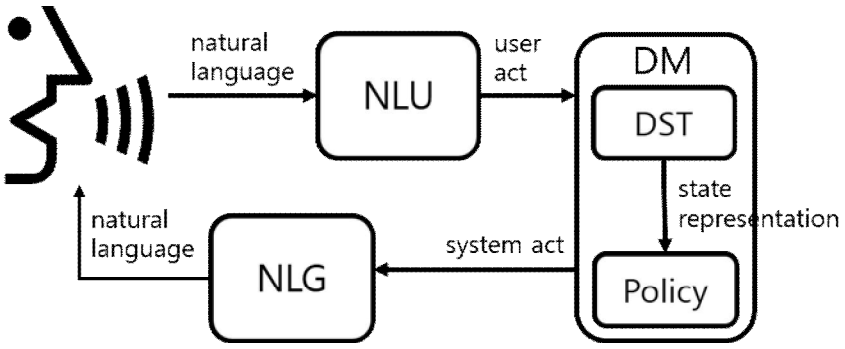
트랜스포머는 순환 신경망 없이 셀프 어텐션만을 이용한다. Lei[16]는 경량화 된 순환신경망인 SRU(Simple Recurrent Unit)을 제안하고 이를 트랜스포머의 어텐션과 함께 사용했다. 어텐션과 연결된 피드포워드 레이어를 SRU로 대체하여 순환신경망이 입력 시퀀스의 순차적 정보와 상대적인 위치 정보를 얻는 효과를 이용했고 기존 트랜스포머보다 높은 BLEU점수를 얻었다. Neishi[17]는 단어 임베딩에 절대적·상대적 위치 인코딩을 적용한 트랜스포머 모델들로 기계독해에 대한 실험을 통해 성능을 비교했다. 실험결과 상대적 위치 인코딩을 적용한 모델이 조금 더 높은 BLEU 점수를 받

으며 절대적 위치 인코딩을 적용한 모델보다 긴 문장을 잘 처리하고 번역의 정확도도 높음을 보여주었다.

Ⅳ. 대화 상태 트래킹(Dialog State Tracking, DST)

1. 목적 지향 대화 시스템

목적 지향 대화 시스템은 사용자가 자연어 형태의 텍스트나 음성을 통해 원하는 작업을 수행할 수 있도록 도와주는 시스템이다. 전통적인 목적 지향 대화 시스템은 [그림 11] 같이 자연어 이해(Natural Language Understanding, NLU), 대화 관리(Dialog Management, DM), 자연어 생성(Nagural Language Generation, NLG)모듈이 파이프라인 형태로 연결된 구조를 갖고 있다.



[그림 11] 목적 지향 대화 시스템

시스템은 사용자 발화로부터 원하는 것이 무엇인지 의도를 파악하기 위해 의도를 사용자가 채워야 할 폼(form)으로 간주하여 대화를 진행한다. 각각의 의도는 사용자의 요청을 처리하기 위해 슬롯 혹은 필드의 셋으로 구성된다. 대화 이해 모듈에서는 도메인 파악(Domain Identification), 의도 분류(Intent Classification)와 슬롯 태깅(Slot Tagging)을 통해 사용자의 의도를 파악하고 대화 내에서 요청한 것이 무엇인지 의도와 관련된 슬롯들을 채운다. 도메인 파악과 의도 분류는 분류 문제로, 슬롯 태깅은 시퀀스 라벨링(sequence labeling) 문제로 접근한다.

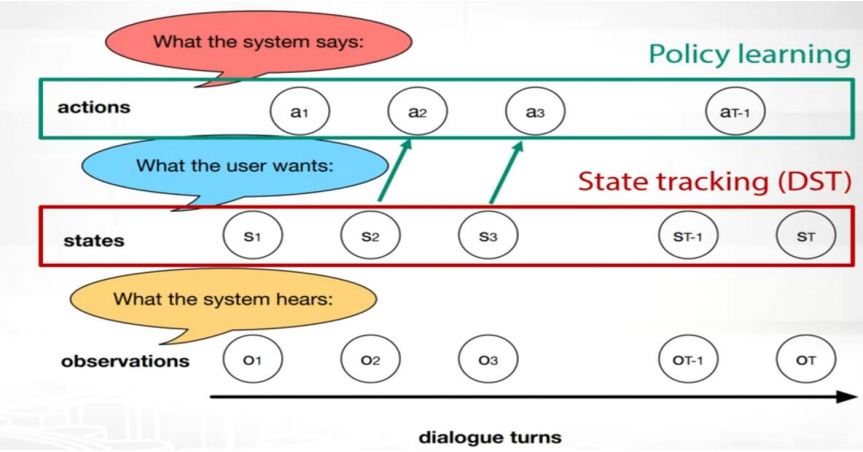
Utterance	Show	flights	from	Seattle	to	San	Diego	tomorrow
Slots	O	O	O	B-fromLoc	O	B-toLoc	B-toLoc	B-depart_date
Intent	Flight							

[표 1] 의도분류와 슬롯 태깅 예

대화 관리 모듈은 대화 이해 모듈의 결과와 대화 기록을 토대로 시스템 응답의 액션을 결정하는 모듈로 대화 상태 트래킹(Dialog State Tracking, DST)과 시스템의 액션을 결정하는 정책(Policy)로 구성된다.

대화 상태 트래킹은 지식베이스와 연계되어 추가적인 정보를 질의하고, 매 사용자의 발화마다 변화하는 대화 상태를 추적한다. 대화 상태(Dialog State)는 사용자가 제공한 대화를 압축적으로 표현하며 어떤 태스크를 완수하기 위해 유지되어야 할 정보이다. 보통 대화 상태 트래킹은 대화를 슬롯 채우기(slot-filling) 문제로 보고 대화 상태를 어떤 도메인에서 미리 정의된 슬롯들과 그 값으로 정의한다. 대화 상태 트래킹은 사용자의 발화를 입력으로 사용자의 목적에 해당하는 슬롯과 값을 예측하는 모듈이다. 대화의 턴(turn)마다 사용자로부터 얻어지는 정보들이 구체화되거나 변화하기 때문에 상태 트래킹 모듈은 그 상태를 추적하는 것을 담당한다. 대화 상태 트래킹은 대화 이해 모듈의 출력을 가지고 룰 기반(rule-based)의 트래킹을 하거나 신경망을 이용한 학습을

한다. 정책은 대화 상태를 입력으로 받아 시스템이 취해야 할 액션을 결정하는 것으로 대화 상태를 시스템 액션으로 매핑하는 과정이다. 정책은 일종의 룰로써 주어진 상태에서 어떤 액션을 취할지 결정하는 것이다. 정책을 학습하는 방법은 룰 기반의 방법과 머신러닝의 지도학습이나 강화학습을 이용하는 방법이 있다.



[그림 12] 대화 관리 흐름

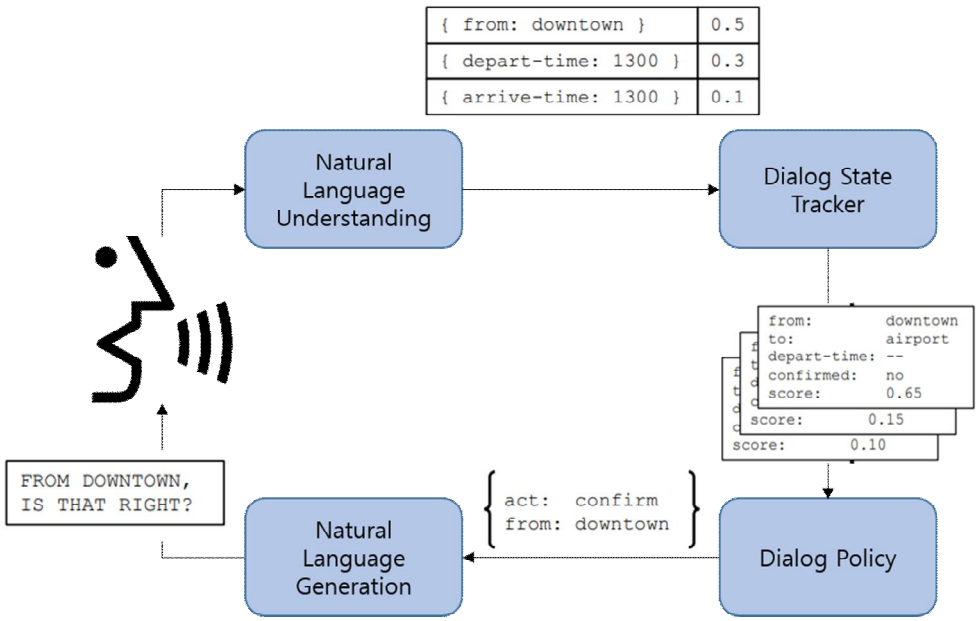
출처 :www.coursera.org/learn/language-processing

[그림 12]는 대화 관리 모듈을 나타낸 그림으로 사용자와 시스템의 대화가 턴 별로 진행되고, 시스템은 사용자의 발화를 관찰하며 대화의 상태를 업데이트한다. 대화 관리 모듈은 이 상태를 추적하는 역할을 하며, 사용자가 궁극적으로 달성하고자 하는 목적을 위해 대화가 진행되면서 점점 구체화되거나 변경되는 정보들을 추출한다. 그리고 시스템은 어떠한 대화 상태에 있을 때 어떤 응답을 해야 하는지에 대한 룰인 정책을 학습해야한다. 마지막으로 자연어 생성 모듈은 대화 관리 모듈의 상태와 시스템 응답을 자연어 형태로 매핑하여 사용자에게 전달한다. 이처럼 한 모듈의 출력을 다음 모듈의 입력으로 받는

대화시스템을 파이프라인 대화 시스템이라고 한다. 그러나 파이프라인 방식의 시스템은 상위모듈에서 발생된 에러가 하위모듈로 전파될 수 있고 문제가 발생했을 때 어떤 지점에서 기인했는지 파악하기 어렵다는 한계가 있다. 그리고 모듈 간의 의존성으로 한 모듈이 변경되면 나머지 모듈들도 다시 학습되어야 하는 문제가 있다. 이 한계를 극복하기 위해서 종단형 목적 지향 대화 시스템(End-to-End Dialog System)이 연구되기 시작했다. 종단형 대화 시스템은 여러 모듈을 하나로 통합해 한 모듈의 출력을 다른 모듈의 입력으로 이용해 학습하지 않고 대화 데이터 자체를 학습하는 방법이다. 종단형 목적 지향 대화 시스템은 기존 파이프라인 시스템의 모듈들을 부분적으로 합친 모델부터 모든 모듈을 하나로 통합한 모델까지 다양하다.

2. 대화 상태 트래킹

대화 상태 트래킹 모듈은 자연어 이해 모듈의 출력을 입력으로 받아 대화의 목적이 달성될 때 까지 정보를 지속적으로 업데이트한다. 따라서 대화 상태는 어떤 시점에서 사용자가 시스템에게 원하는 것에 대한 표현이며 대화 히스토리에 근거해 다음 상태를 예측하게 하는 정보이다. 상태 트래킹 모듈은 대화 히스토리에 대한 관찰 분포인 자연어 이해 모듈의 출력과 시스템 액션을 입력으로 현재 상태에 대한 추정분포를 예측한다.



[그림 13] 파이프라인 다이얼로그 시스템

일반적인 파이프라인 시스템에서는 [그림 13]과 같이 자연어 이해 모듈이 사용자 발화를 처리하여 도메인, 의도, 슬롯에 대한 신뢰점수를 N-best 리스트 형태로 생성하여 대화 상태 트래킹 모듈로 전달한다. 자연어 이해 모듈에서 종종 발생하는 에러는 파이프라인 시스템 특성상 하위로 전파되어 대화 상태 트래킹을 어렵게 한다. 따라서 자연어 이해 모듈을 없애고 사용자 발화와 대화 히스토리로부터 대화 상태를 바로 추론하는 종단형 대화 상태 트래킹 모델들이 등장하기 시작했다[18, 19, 20, 21, 22]. [18]에서는 처음으로 어텐션을 적용하여 종단형 방식으로 의도 분류와 DST의 슬롯 태깅을 함께 학습하는 Joint Training 방법을 제안했다.

```
  recommend(restaurant name= Au Midi, neighborhood = midtown,
  cuisine = french
1 restaurant_name is in neighborhood and serves cuisine food.
2 There is a cuisine restaurant in neighborhood called restaurant_name.
```

[그림 14] delexicalized sentences

출처: Speech and Language Processing. Daniel Jurafsky & James H. Martin.

이 모델들은 직접 추출한 피쳐 대신 신경망을 이용해 데이터에 대한 표현을 학습한다. 그리고 [그림 14]와 같이 사용자의 발화에서 슬롯에 해당되는 값을 대표 명사로 치환(delexicalization)하여 인코더-디코더 구조를 통해 토큰을 생성할 때 슬롯에 알맞은 값을 끼우는 방법을 사용했다. 그러나 이 방법 또한 고정 온톨로지를 사용하기 때문에 온톨로지에 존재하지 않는 슬롯 값에 대해서 처리할 수 없다는 한계가 있다. [5, 23]에서는 이런 문제를 지적하며 미리 정해진 슬롯-값 사전을 이용하지 않고 독립적인 슬롯 인코더를 사용하여 슬롯의 값에 해당하는 단어들과 슬롯을 매칭해 슬롯-값 쌍에 대한 점수를 매기거나 포인터 네트워크를 사용하여 처음 등장하는 단어들을 처리하도록 했다. 이 모델들을 기반으로 [6, 24]에서는 위 모델들을 경량화하거나 대화 인코딩에 대한 부분을 개선하면서 더 확장성있는 모델을 만드는데 주력하였다.

Ⅴ. 실험

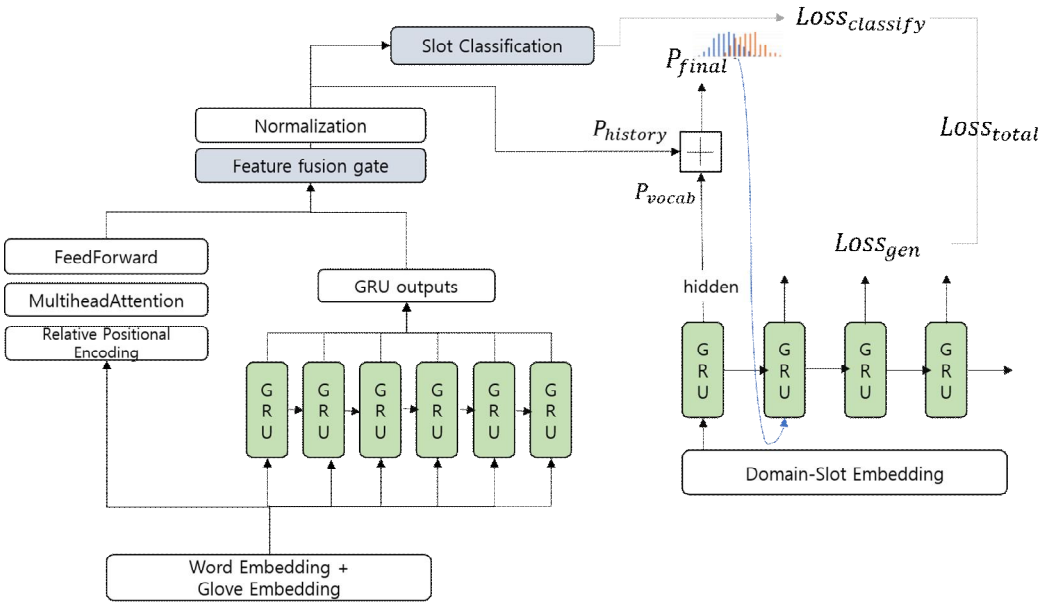
1. 제안 모델

앞 장에서는 관련 연구를 통해 목적 지향 대화 시스템에서의 대화 상태 트래킹과 멀티 도메인에서의 모델들을 살펴보았다. 이번 5장에서는 멀티 도메인의 대화 데이터셋을 입력 데이터로 활용하여 해당 대화에 포함된 도메인, 슬롯, 값을 추출해내는 모델을 제안하고 실험을 진행한다.

일반적으로 대화 상태 트래킹 연구에서는 사전에 온톨로지에 슬롯들과 각 슬롯이 취할 수 있는 값을 정의하여 이에 국한된 대화데이터를 사용하였다. 그러나 사전에 모든 값들이 정의된 온톨로지를 구축하는 것은 많은 비용이 들고 서비스를 만드는 회사에서는 제3자가 보유한 데이터베이스에 접근하는 것이 외부 API를 통하게 되므로 접근성을 확보하는 것이 쉽지 않다[23]. 따라서 대화 데이터 셋에서 온톨로지의 고정된 슬롯-값을 사용하지 않고 대화 자체만을 이용하여 대화 상태를 트래킹할 수 있도록 입력 텍스트인 대화 히스토리로부터 슬롯-값 후보들을 생성하는 개방형 단어 기반 방식의 트래킹 모델을 개발하고자 한다. 셀프 어텐션 메커니즘은 문장 내의 토큰들 간의 연관된 정도를 학습하고 이 정보를 바탕으로 문장 구조에 대한 피쳐를 추출하는 방법으로 기계번역, 독해 연구들에서 이를 이용한 모델들이 좋은 성과를 보였다. 본 연구의 모델에서도 대화 텍스트를 인코딩하여 피쳐를 표현하기 위해 셀프 어텐션을 여러 번 실행하는 멀티헤드 어텐션을 이용한다. RNN기반의 모델들은 그 자연적인 특성으로 인해 시퀀스 내의 토큰들의 위치적인 정보를 포함하고 있다. 하지만 셀프 어텐션은 그 자체로는 입력 시

퀀스에 대한 위치 정보를 익힐 수 없기 때문에 입력 문장의 토큰들에 대한 위치 임베딩을 제공해주어야 한다. 따라서 인코더에서는 단어 임베딩에 상대적 위치 인코딩을 적용하여 멀티헤드 어텐션을 수행하고 이 결과를 순환 신경망의 출력과 결합한다.

모델의 디코더에서는 슬롯에 대한 값을 생성하면서 현재 대화에서 적절한 단어를 복사해오는 복사 메커니즘을 사용해 온톨로지에 없는 단어들이 대화에 등장하는 경우에도 입력 문장으로부터 단어를 복사해오면서 OOV문제를 해결하고자 한다. 그리고 현재 대화 데이터에서 어떤 슬롯들이 포함되어 있는지 각 슬롯에 대한 분류를 하는 슬롯 분류기를 보조적인 학습방법으로 함께 이용한다. 슬롯 분류는 디코딩 과정에서 함께 사용되어 모델이 슬롯에 대한 값 생성하는 디코딩 프로세스를 도와준다. 슬롯 분류에서는 각 슬롯을 {‘none’, ‘dontcare’, ‘other’}로 분류한다.



[그림 15] 제안 모델

1-1. 인코더

인코더는 입력 시퀀스를 임베딩 층으로 주입해 벡터화하고 양방향 GRU를 통해 시퀀스를 인코딩한다. 인코더의 입력은 $X_t = [U_1, R_1, ..., U_t, R_t]$ 로 정의하며 사용자 발화와 시스템 응답을 담은 대화의 턴 t까지를 나타낸다. 임베딩은 300차원의 Glove 임베딩[32]과 100차원의 캐릭터 임베딩을 사용하여 400차원의 임베딩 emb를 사용한다. 임베딩 레이어는 입력 X_t의 단어들을 400차원의 단어 임베딩으로 투영한다. GRU는 임베딩 레이어의 결과를 받아 새로운 은닉 상태 h_t를 계산한다. GRU가 양방향이기 때문에 양방향 출력을 합쳐 다음 레이어로 전달한다.

$$emb = [GloVe, CharEmbedding]$$

$$h_t = \overrightarrow{GRU(emb(X_t))} \oplus \overleftarrow{GRU(emb(X_t))}$$

멀티헤드 어텐션은 상대적 위치 인코딩을 적용한 단어 임베딩을 입력으로 받는다. 그리고 시퀀스 내의 쿼리 벡터와 키 벡터간의 유사도가 계산된 어텐션의 결과는 정규화 레이어를 거쳐 피드포워드 레이어로 전달되고 어텐션의 결과와 피드포워드의 결과를 더하는 잔차 연결(Residual Connection)을 한다. 잔차 연결은 여러 레이어들이 있을 때 마지막 레이어의 결과에 입력을 다시 더해줌으로써 입력 데이터의 정보가 소실되는 것을 막는다. 여기서는 피드포워드의 결과에 멀티헤드 어텐션의 결과를 더함으로써 위치 정보를 보존하는 효과가 있다. 피드포워드 레이어의 결과는 다시 정규화 레이어를 통과한다. 그리고 기존 GRU의 출력과 피드포워드 레이어의 출력을 Feature fusion gate를 이용해 두 표현을 하나의 표현으로 합치는 과정을 거친다. Feature fusion

gate는 식 (1), (2)에서 학습 가능한 두 가중치 매트릭스를 도입해 f_t,g_t를 만들고 식 (3)을 통해 기존 표현인 GRU의 출력과 새롭게 합쳐진 표현사이에서 정보를 선택하여 합침으로써 최종적으로 인코딩된 문맥표현 h_t^{enc}를 출력한다.

$$s_t = MultiHead(emb(X_t), emb(X_t), emb(X_t))$$

$$f_t = tanh(W^f [h_{t,}\ s_t]) \tag{1}$$

$$g_t = sigmoid(W^g [u_{t,}{}' s_t]) \tag{2}$$

$$h_t^{enc} = g_t \odot f_t + (1 - g_t) \odot h_t \tag{3}$$

1-2. 디코더

디코더에서는 인코딩 된 대화 히스토리를 이용해 대화와 관련된 도메인과 슬롯을 찾아내고 이에 대한 값을 디코딩한다. 디코더 GRU의 입력으로는 도메인 단어 임베딩과 슬롯 단어 임베딩을 합쳐 (도메인, 슬롯) 쌍을 사용하고 각 쌍마다 알맞은 값을 예측해야한다.

디코더에서는 (도메인, 슬롯)에 대한 값을 디코딩하기 위해 복사 메커니즘을 이용한다. 복사 메커니즘은 현재 대화 내의 단어 분포로부터 단어를 생성해낼지 전체 단어 임베딩으로부터 단어를 생성해낼지 결정한다. 이 과정을 통해 대화 내의 단어가 학습 데이터에 존재하지 않더라도 슬롯에 대한 값을 디코딩할 수 있도록 한다. 대화 분포 $p_{history}$ 와 전체 단어 분포 p_{vocab}는 디코더의 매

스텝마다 생성된다.

$$p_{kt}^{vocab} = softmax(emb \bullet h_{kt}),\ p_{kt}^{history} = softmax(h_t^{enc} \bullet h_{kt}) \quad (1)$$

$$p_{kt}^{\text{gen}} = sigmoid(W \bullet [h_{kt}; c_{kt}; w_{kt}]),\ c_{kt} = p_{kt}^{history} \bullet h_t^{enc} \quad (2)$$

$$p_{kt}^{final} = p_{kt}^{\text{gen}} \bullet p_{kt}^{vocab} + (1 - p_{kt}^{\text{gen}}) \bullet p_{kt}^{history} \quad (3)$$

스텝 t에서 k번째 (도메인, 슬롯) 입력인 w_{kt}에 대한 값을 생성할 때 GRU에서 출력되는 은닉상태는 h_{kt}이다. 먼저 디코더는 h_{kt}와 전체 단어 임베딩 emb와 내적해 p_{kt}^{vocab}를 만들고, h_{kt}를 인코더의 출력 h_t^{enc}과 내적해 $p_{kt}^{history}$를 만든다. 두 분포를 하나의 분포로 만들기 위해서 p_{kt}^{gen}을 정의한다. p_{kt}^{gen}은 학습대상인 확률 값으로 두 분포를 어느 비율로 합칠 것인지 결정하는 값이다. p_{kt}^{gen}은 식 (2)를 이용해 계산된다. c_{jk}는 디코더의 은닉상태와 인코더의 출력을 내적한 컨텍스트 벡터이다. 학습된 p_{kt}^{gen}을 이용해 두 분포에 가중치를 적용하여 최종 단어출력에 대한 분포 p_{kt}^{final}을 계산한다.

디코더에서는 인코더의 출력과 현재 (도메인, 슬롯)이 현재 대화와 연관이 있는지에 대해 다중 클래스 분류를 한다. 분류 클래스는 모든 슬롯을 'none', 'dontcare', 'other'로 분류한다. 분류기는 각 대화 턴에서 어떤 도메인-슬롯 쌍이 언급되지 않으면 'none', 사용자가 이 슬롯에 어떤 값이든 수용할 수 있다면 'dontcare', 해당 슬롯에 어떤 값이 채워져야 한다면 'other'로 분류한다. 모델은 'none'으로 분류되는 관련 없는 슬롯에 대해 디코더가 출력한 값을

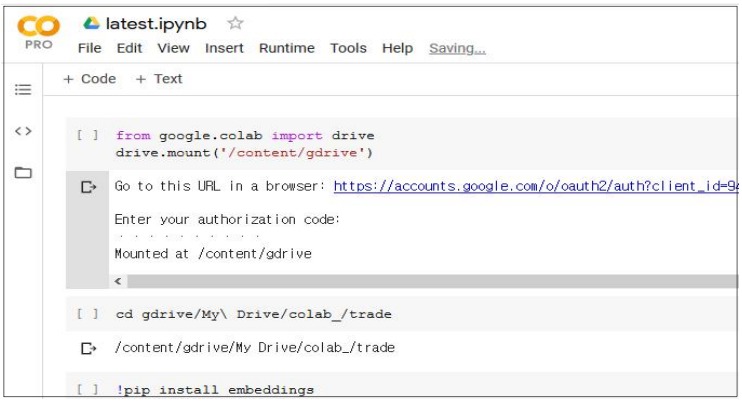
‘none’으로 치환하고 ‘other’로 분류된 슬롯에서만 복사 메커니즘에 의해 단어를 대화로부터 복사할지 새롭게 생성할지 결정한다. 모델은 슬롯에 대한 값을 생성함과 동시에 슬롯 분류를 함께 학습하고 최적화한다. 표준 크로스 엔트로피를 이용하며 슬롯 값 생성 Loss와 슬롯 분류 Loss를 합친 값을 전체 Loss로 정의한다.

$$Loss_{total} = Loss_{\text{gen}} + Loss_{classify}$$

2. 실험 개요

본 실험에서는 제안모델을 구축하여 멀티 도메인을 다룬 대화가 담긴 텍스트를 입력 데이터로 활용하여 대화 상태를 출력하는 실험을 진행한다. 실험은 제안모델에 대해 학습을 진행하여 결과를 확인하고 같은 데이터셋을 이용한 관련 연구들의 결과와 비교한다. 그리고 베이스라인 모델인 seq2seq 모델을 기준으로 제안모델에 적용된 기법들을 여러 개의 버전으로 만들어 구조별로 모델결과를 비교해본다.

실험에 쓰일 프로그래밍 언어는 Facebook AI에서 제공하는 오픈소스 딥러닝 라이브러리인 Pytorch이며, 학습은 브라우저를 통한 클라우드 기반의 Jupyter 노트북 개발환경을 제공하는 Google Colab에서 진행한다. Colab은 Google Drive와의 연동을 이용해 코드를 업로드하거나 Jupyter 노트북 내에서 바로 작성가능하다. 소프트웨어와 하드웨어에 대한 사항은 [표 2]에 있다.



[그림 16] Google Colab Jupyter

H/W	사양	S/W	버전
CPU	Intel(R) Xeon(R) CPU @ 2.30GHz	Pytorch	1.4.0
RAM	26 GB	CUDA	10.2
GPU	Tesla P100-PCIE-16GB	OS	Ubuntu 16.04

[표 2] 실험에 쓰인 하드웨어, 소프트웨어 사양

3. 실험 데이터 설정

Multiwoz 2.0[30]은 여러 도메인에서 다양한 주제를 가지고 사람-사람의 대화를 라벨링한 데이터셋으로 목적지향, 다중 도메인, 사람 간의 대화를 담은 데이터셋 중 가장 규모가 크다. Multiwoz는 평균 13.67턴의 1만개의 대화를 담고 있고 7개의 도메인과 37개의 슬롯타입을 포함한다. [표 3]은 실험에 사

용할 데이터, 도메인, 슬롯 갯수를 정리한 것이고 [표 4]는 도메인 별 슬롯을 정리한 것이다. MultiWOZ의 7개의 도메인 중 학습하기에 데이터가 적은 2개의 도메인(병원, 경찰서)을 제외하고 데이터 수가 충분한 5개의 도메인과 도메인에 속한 30개의 슬롯을 이용한다. 'area', 'type' 과 같은 몇 슬롯은 여러 도메인에서 나타나며 실험에서는 도메인 간에 슬롯을 공유하지 않는다. 따라서 디코더의 입력이 되는 도메인과 슬롯은 'domain-slot', 대화 상태 트래킹 결과로 출력되는 도메인과 슬롯, 값은 'domain-slot-value' 형식으로 나타낸다. 만약 사용자가 슬롯에 대한 값을 할당하지 않은 경우 'None'으로 나타낸다.

[표 4]는 데이터 셋의 예시로 각 대화 마다 여러 개의 턴으로 구성되어 있다. 학습에는 대화에 해당하는 'system_transcript', 'transcript'를 사용하고 대화 상태 예측에 대한 정답을 확인하기 위해 'belief_state', 'domain' 항목을 사용한다.

Dataset	Training dialogues	Validation dialogues	Test dialogues	Domain	Domain-Slot pair	Vocab Size
Multiwoz 2.0	8438	999	1000	5	30	18311

[표 3] 학습에 사용할 MultiWoz 2.0 데이터

Dataset	Domain	Slot
Multiwoz 2.0	Hotel	'pricerange', 'type', 'parking', 'book stay', 'book day', 'book people', 'area', 'stars', 'internet'
	Train	'destination', 'day', 'departure', 'arriveby', 'book people', 'leaveat'
	Restaurant	'food', 'pricerange', 'area', 'name', 'book time', 'book day', 'book people'
	Attraction	'area', 'name', 'type'
	Taxi	'leaveat','destination', 'departure','arriveby'

[표 4] Multiwoz 2.0의 도메인 별 슬롯

"dialogue_idx": "SNG01856.json"		
"domains": "hotel"		
dialogue	turn_index	0
	transcript	"I am looking for a place to stay that has cheap price range it should be in a type of hotel",
	domain	"hotel“
	system _transcript	“”
	belief_state	{"slots": [["hotel-pricerange", "cheap"]], "act": "inform"}, {"slots": [["hotel-type", "hotel"]], act": "inform" }
	turn_label	[["hotel-pricerange", "cheap"] , ["hotel-type", "hotel"]]
	turn_index	1
	transcript	"no , i just need to make sure it s cheap . oh , and i need parking"
	domain	"hotel"
	system _transcript	"okay , do you have a specific area you want to stay in ?"
	belief_state	{ "slots": [["hotel-pricerange","cheap"]], "act": "inform" }, { "slots": [["hotel-type", "hotel"]], "act": "inform" }
	turn_label	["hotel-parking", "yes"]

[표 5] Multiwoz 2.0 데이터 예제

4. 실험 설정

모델은 기본적으로 인코더-디코더 구조로 모델에 쓰일 단어 임베딩은 사전 훈련된 400차원의 Glove 단어 임베딩과 캐릭터 임베딩을 이용해 초기화했다. 인코더의 hidden size는 임베딩 차원과 같이 400으로 설정했으며 트랜스포머에서는 일반적으로 6개의 멀티헤드 어텐션과 피드포워드 레이어를 사용하지만 본 실험에서는 1개의 레이어만을 이용했다. 멀티헤드 어텐션의 head는 8개로 설정했고 어텐션 레이어와 이어진 피드포워드 레이어는 hidden size의 두배인 800으로 설정하였다. 학습률은 [0.001, 0.0001]이며 매 epoch마다 학습 스케줄러에 의해 조정되도록 하였다. Optimizer는 Adam을 이용하고 Batch는 16으로 설정했다. Dropout 비율은 멀티헤드 어텐션에서는 0.1, 나머지는 전부 0.3을 사용했으며 오버피팅을 방지하기 위해 초반 15 epoch까지 이 비율을 유지하고 이후 40 epoch까지 0.4로 조정하여 실험했다. [표 6]은 모델 성능평가 방법을 나타낸 것으로 멀티 도메인에서의 상태 트래킹의 성능을 평가하기 위해 Joint Goal Accuracy와 Average Slot Accuracy를 이용한다. Joint Goal Accuracy는 예측된 대화 상태와 정답을 비교하며 도메인, 슬롯, 값이 모두 일치할 때만 정답으로 간주한다. Average Slot Accuracy는 슬롯이나 값 중 하나만 정답과 일치하는 경우에도 정답으로 간주한다.

평가 방법	평가 기준
Joint Goal Accuracy	모든 슬롯과 값(joint goal)이 정확하게 예측된 대화 턴의 비율
Average Slot Accuracy	대화의 턴 별로 정확하게 예측된 슬롯이나 값의 비율

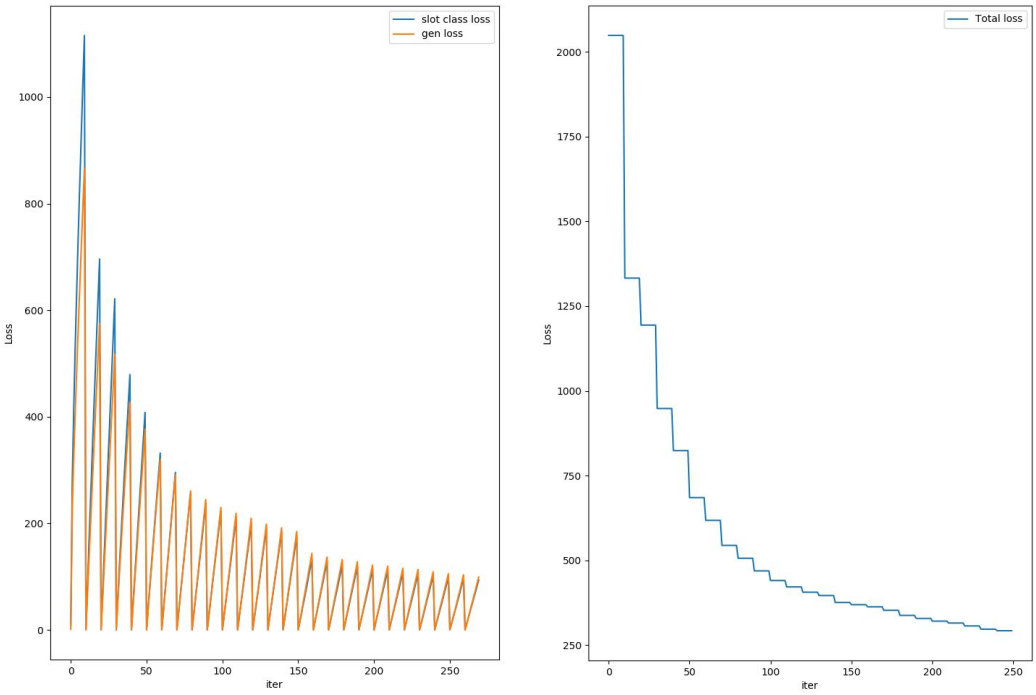
[표 6] 대화 상태 트래킹 평가 방법

Ⅵ. 실험 결과

1. 제안모델을 이용한 대화상태 트래킹 결과

Turn	Dialog	State
0	U : i am looking for a cheap restaurant in the center of the city .	restaurant-pricerange-cheap restaurant-area-centre
1	SYS : do you have any specific type of food you would like ? U : no i am not picky as long as the price -s are low .	restaurant-pricerange-cheap restaurant-area-centre
2	SYS : there is a cheap chinese restaurant called the dojo noodle bar located in the centre of town . would you like to book a table ? U : yes please for 8 people at 18:30 on thursday .	restaurant-pricerange-cheap restaurant-area-centre restaurant-book time-18:30 restaurant-book day-thursday restaurant-book people-8
3	SYS : i am sorry but dojo noodle bar is solidly booked at that time . i can try a different time or day for you . U : can you try to book it at 17:30 .	restaurant-pricerange-cheap restaurant-area-centre restaurant-book time-17:30 restaurant-book day-thursday restaurant-book people-8

[표 7] 테스트 데이터의 대화와 상태 트래킹 결과 예시



[그림 17] 모델 학습 그래프 (slot class loss/gen loss, total loss)

Models	Joint Accuracy	Slot Accuracy
GLAD[5]	0.355	-
GCE[6]	0.374	-
SUMBT[11]	0.424	0.910
HyST[7]	0.442	-
TRADE[31]	0.486	0.969
proposed model	0.505	0.971

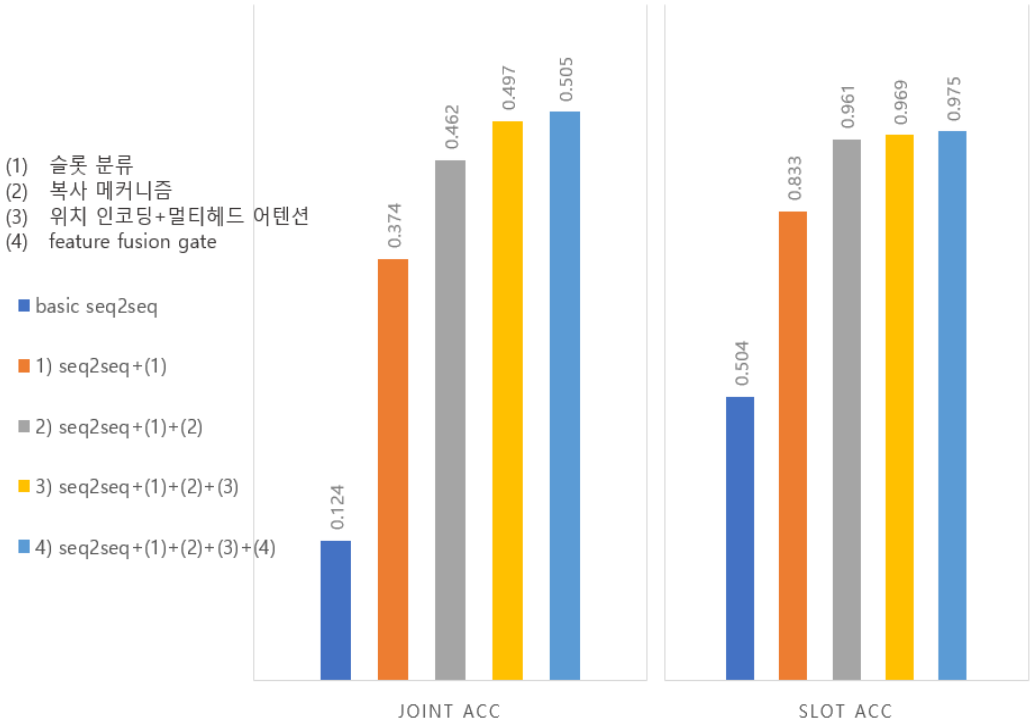
[표 8] 멀티 도메인에서의 상태 트래킹 결과

[표 7] 모델이 대화 데이터에서 예측한 대화 상태를 나타낸 것으로 대화 턴마다 '도메인-슬롯' 에 대한 값을 출력한다. [표 8]은 멀티 도메인 설정에 대해 실험하여 Joint Goal Accuracy와 Average Slot Accuracy를 계산한 결과이다. 비교 모델들은 모두 Multiwoz 2.0 데이터 셋에 대해 실험했다. 제안 모델은 비교 모델 중 높은 성능을 보인 TRADE 모델과 인코더-디코더로 기본 구조는 같지만 위치 인코딩과 멀티헤드 어텐션, Feature fusion gate를 사용했다는 점이 다르다. 실험 결과 제안 모델이 가장 높은 Joint accuracy를 보였다. [표 9]는 싱글 도메인에 대한 제안모델과 TRADE 모델의 실험 결과이다. 학습 데이터를 한 도메인으로만 한정했을 때의 실험 결과로 모든 도메인에서 제안 모델이 Joint accruacy를 기준으로 더 나은 성능을 보였다.

Single Domain				
	Proposed model		TRADE	
domain	Joint	Slot	Joint	Slot
hotel	0.584	0.933	0.555	0.926
restaurant	0.680	0.938	0.653	0.953
train	0.787	0.954	0.777	0.889
taxi	0.761	0.905	0.761	0.932
attraction	0.746	0.904	0.716	0.895

[표 9] 제안모델과 TRADE의 개별 도메인 학습에 대한 실험 결과

2. 모델의 구조별 실험 결과



[그림 18] 모델의 구조 별 상태 트래킹 결과

Encoding methods	Joint Accuracy	Slot Accuracy
Absolute Positional Encoding (1)	0.489	0.968
Relative Positional Encoding (2)	0.505	0.975

[표 10] 인코딩 방법 별 상태 트래킹 결과

[그림 18]은 기본 모델인 seq2seq 부터 제안모델을 구축하기 까지 구조를 하나씩 추가한 모델에 대해 각각 실험한 결과이다. 실험에서 seq2seq 모델에 슬롯 분류, 복사 메커니즘을 추가한 모델 (1), (2)는 GRU의 출력이 바로 디코더로 전달된다. 상대적 위치 인코딩과 멀티헤드 어텐션을 사용한 모델은 GRU의 출력과 멀티헤드 어텐션의 출력을 하나로 합쳐서 디코더로 전달한다. 실험 결과 기본 모델에 새롭게 추가한 구조들이 도메인과 슬롯, 값을 모두 찾는 Joint Goal Accuracy를 높이면서 좋은 성능을 냈고 Feature fusion gate 까지 추가한 모델이 정확도 0.505로 가장 높게 나타났다. 특히 슬롯 분류와 복사 메커니즘을 추가한 것이 정확도를 높이는데 많은 기여를 했다. [표 10]은 멀티헤드 어텐션의 입력을 인코딩하는 방법에 따른 결과이다. 단어 임베딩을 절대적·상대적 위치 인코딩을 하여 멀티헤드 어텐션의 입력으로 사용한 모델 중에서 상대적 인코딩을 적용한 모델이 Joint Accuracy와 Slot Accuracy 모두 더 좋은 결과를 보였다.

Zero-shot				
	Proposed model		TRADE	
domain	Joint	Slot	Joint	Slot
hotel	0.152	0.698	0.137	0.653
restaurant	0.125	0.546	0.115	0.534
train	0.229	0.487	0.223	0.493
taxi	0.635	0.778	0.605	0.739
attraction	0.214	0.591	0.198	0.555

[표 11] 제안모델과 TRADE의 Zero-shot 실험 결과

[표 11]은 제안모델과 TRADE의 Zero-shot 실험 결과를 나타낸 것이다. Zero-shot 은 라벨링 된 데이터만을 이용하여 데이터에 없는 클래스를 분류, 예측하는 실험으로 학습 데이터에서 없는 단어들이 등장했을 때 모델이 처리

하는 능력을 확인하고자 한다. 이 방법을 사용하는 이유는 실세계에는 매우 많은 도메인들이 존재하고 새로운 도메인에 대해 라벨링 된 데이터를 구축하는 것은 많은 비용이 들기 때문이다. 더욱이 딥러닝 모델들은 한 클래스를 학습하는 데 많은 데이터를 필요로 한다. 본 실험에서는 모델이 새롭게 나타나는 도메인을 분류하고 슬롯에 대한 값을 생성할 수 있는 지 평가하기 위해 Zero-shot 실험을 진행하였다. 실험은 전체 도메인 중 특정 도메인을 제외하여 학습한 뒤 그 도메인에 대해서만 테스트를 하는 방식으로 진행했다. 멀티 도메인은 특성상 슬롯들이 여러 도메인에서 중복되어 나타날 수 있다. Zero-shot 설정에서는 테스트 도메인의 슬롯이 학습 데이터에서 나타나지 않는 경우 더욱 어려운 평가가 될 수 있다. 하지만 학습 데이터에서 나타난 슬롯들이 테스트 도메인에서도 나타난다면 비교적 나은 정확도를 보여주기도 한다. 실험 결과 제안 모델이 비교 모델보다 최대 3% 더 높은 Joint Accuracy를 나타냈다.

Ⅶ. 결론 및 향후연구

본 연구에서는 셀프 어텐션을 포함한 인코더-디코더 구조를 이용해 멀티 도메인에서 대화 상태를 트래킹하는 모델을 구축해 학습하고, 슬롯과 그 값을 잘 찾아냈는지 정확도를 평가하였다. 본 연구를 통해 얻어진 주요 내용을 정리하면 다음과 같다.

- 대화 데이터를 인코딩 하기 위해 순환신경망과 셀프 어텐션 메커니즘을 함께 사용한 인코더를 이용한다.
- 셀프 어텐션에 사용되는 위치 인코딩을 두 가지 인코딩 방법(상대적, 절대적 위치 인코딩)을 사용하고 비교해본다.
- 디코더에서 슬롯 값을 생성할 때 복사 메커니즘을 이용해 학습 데이터에 없는 단어를 처리한다.
- 실험 결과, 상대적 위치 인코딩, 셀프 어텐션을 적용한 모델이 가장 높은 Joint Goal Accuracy인 0.505를 기록했다.

실험에서는 셀프 어텐션을 기존 순환신경망과 함께 사용하여 효과적으로 데이터를 인코딩할 수 있음을 확인했다. 이는 셀프 어텐션을 사용한 인코더를 통해 순환신경망 만을 이용했을 때 보다 데이터로부터 더 많은 정보를 인코딩할 수 있음을 나타낸다. 위치 인코딩은 인코더의 입력인 시퀀스 임베딩에 적용되는데 절대적 위치 인코딩보다 상대적 위치 인코딩을 한 모델들이 더 좋은 성능을 보였다. 그리고 멀티헤드 어텐션과 순환신경망의 출력을 하나의 분포로 합칠 때 단순히 이어붙여서 선형 레이어로 연결하는 것 보다 Feature fusion gate를 통해 합치는 방법이 더 높은 정확도를 보였다. 이 과정을 통해

여러 층의 멀티헤드 어텐션 블록을 사용하지 않고도 순환신경망을 함께 사용함으로써 충분히 인코더의 성능이 개선되었음을 알 수 있다. 모델의 여러 버전을 실행해본 결과 기존 GRU 출력과 합치지 않고 멀티헤드 어텐션을 단독으로 사용했을 때는 다른 모델 버전들에 비해 낮은 정확도가 나왔다. 따라서 인코더 뿐만 아니라 디코더에서도 멀티헤드 어텐션을 이용했을 때 비슷한 성능을 낼 수 있는지에 대해 연구를 진행해볼 필요가 있다. 디코더에서는 미리 정의된 슬롯-값을 사용하지 않고 복사 메커니즘을 이용해 온톨로지에 의존하지 않고 입력 데이터로부터 후보를 생성하는 개방형 단어기반 접근이 OOV문제를 경감시키고 확장성을 높이는데 기여함을 보였다.

일반적으로 기계독해에서는 텍스트를 문장 단위로 끊어서 학습하는 반면, 대화데이터는 턴 별로 입력 데이터를 사용하기 때문에 상대적으로 긴 텍스트를 사용한다. 멀티헤드 어텐션이 문장단위에서는 잘 작동할지라도 문단처럼 길이가 길어질수록 단어 간의 연관성을 잘 파악하지 못하는 경우가 있다. 따라서 대화처럼 긴 텍스트를 처리할 때 Pooling이나 Sliding Window 등을 멀티헤드 어텐션과 함께 이용해 지역적인 정보와 멀리 떨어진 정보들을 잘 캡쳐할 수 있는 방법에 대해 더 연구해 볼 필요가 있다. 그리고 대화 상태 트래킹 모델을 이용해 시스템의 응답을 생성하는 모델로의 확장하는 방법에 대해 연구할 예정이다.

참 고 문 헌

[1] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 551 - 561, Austin, Texas, November 2016b. Association for Computational Linguistics. URL https://aclweb.org/anthology/D16-1053.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30, pages 5998 - 6008. Curran Associates, Inc., 2017.

[3] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. Disan: Directional self-attention network for rnn/cnn-free language understanding. In *AAAI Conference on Artificial Intelligence.*

[4] Osman Ramadan, Pawel Budzianowski, and Milica Gasic. 2018. Large-scale multi-domain belief track-ing with knowledge sharing. InACL.

[5] Zhong, Victor et al. “Global-Locally Self-Attentive Dialogue State Tracker.” ArXiv abs/1805.09655 (2018)

[6] Nouri, E., & Hosseini-Asl, E. (2018). Toward Scalable Neural Dialogue State Tracking Model. ArXiv, abs/1812.00899.

[7] Goel, Rahul et al. “HyST: A Hybrid Approach for Flexible and Accurate Dialogue State Tracking.” ArXiv abs/1907.00883 (2019)

[8] Bing Liu and Ian Lane. An end-to-end trainable neural network model with belief tracking for taskoriented dialog. Proc of Interspeech, 2017.

[9] Tsung-Hsien Wen, David Vandyke, Nikola Mrkši, Milica Gaši, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (ACL): Volume 1, Long

Papers, pages 438 – 449, 2017.

[10] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning, pages 280 – 290, Berlin, Germany, August 2016. Association for Computational Linguistics.

[11] Hwaran Lee, Jinsik Lee, Tae-Yoon Kim. 2019. SUMBT: Slot-Utterance Matching for Universal and Scalable Belief Tracking. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5478 – 5483

[12] Young, Tom et al. “Recent Trends in Deep Learning Based Natural Language Processing [Review Article].” IEEE Computational Intelligence Magazine 13 (2018): 55-75.

[13] Bahdanau, Dzmitry et al. “Neural Machine Translation by Jointly Learning to Align and Translate.” CoRR abs/1409.0473 (2015): n. pag.

[14] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 35 – 45. Association for Computational Linguistics.*

[15] Shaw, P., Uszkoreit, J., & Vaswani, A. (2018). Self-attention with relative position representations. arXiv preprint arXiv:1803.02155.

[16] Lei, Tao et al. “Simple Recurrent Units for Highly Parallelizable Recurrence.” EMNLP (2018).

[17] Masato Neishi, Naoki Yoshinaga . On the Relation between Position Information and Sentence Length in Neural Machine Translation. In Proceedings of the 23rd Conference on Computational Natural Language Learning, pages 328 – 338

[18] Liu, B., & Lane, I. (2016). Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling. INTERSPEECH.

[19] Matthew Henderson, Blaise Thomson, and Steve Young. Deep neural network approach for the dialog state tracking challenge. In

Proceedings of the SIGDIAL 2013 Conference, pages 467{471, 2013.

[20] Mrksic, N., Séaghdha, D.Ó., Wen, T., Thomson, B., & Young, S.J. (2016). Neural Belief Tracker: Data-Driven Dialogue State Tracking. ArXiv, abs/1606.03777.

[21] Matthew Henderson, Blaise Thomson, and Steve Young. 2014b. Word-based dialog state tracking with recurrent neural networks. In Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), pages 292 – 299. Association for Computational Linguistics.

[22] Lukas Zilka and Filip Jurcicek. 2015. Incremental lstm-based dialog state tracker. In Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on, pages 757 – 762. IEEE.

[23] P. Xu and Q. Hu, “An end-to-end approach for handling unknown slot values in dialogue state tracking,” in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, 2018, pp. 1448 – 1457.

[24] Balaraman, Vevake and Bernardo Magnini. “Scalable Neural Dialogue State Tracking.” 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU) (2019): 830-837.

[25] Abhinav Rastogi, Dilek Hakkani-Tur, and Larry Heck. Scalable multi-domain dialogue state tracking. In Proceedings of IEEE ASRU, 2017.

[26] Rahul Goel, Shachi Paul, Tagyoung Chung, Jeremie Lecomte, Arindam Mandal, and Dilek Hakkani-Tur. 2018. Flexible and scalable state tracking framework for goal-oriented dialogue systems. arXiv preprint arXiv:1811.12891.

[27] Matthew Henderson, Blaise Thomson, and Steve J. Young. Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. 2014 IEEE Spoken Language Technology Workshop (SLT), pp. 360 – 365, 2014c.

[28] Le, H.T., Socher, R., & Hoi, S.C. (2020). Non-Autoregressive Dialog State Tracking. ArXiv, abs/2002.08024.

[29] Mihail Eric and Lakshmi Krishnan and Francois Charette and

Christopher D. Manning. 2017. Key-Value Retrieval Networks for Task-Oriented Dialogue. In Proceedings of the Special Interest Group on Discourse and Dialogue (SIGDIAL).

[30] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, I ˜nigo Casanueva, Stefan Ultes, Osman Ra-madan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In Proceedings of the2018 Conference on Empirical Methods in Natural Language Processing, pages 5016 - 5026

[31] Wu, Chien-Sheng et al. “Transferable Multi-Domain State Generator for Task-Oriented Dialogue Systems.” ArXiv abs/1905.08743 (2019): n. pag.

[32] Pennington, Jeffrey et al. “Glove: Global Vectors for Word Representation.” EMNLP (2014).

[33] Hochreiter, Sepp and Jürgen Schmidhuber. “Long Short-Term Memory.” Neural Computation 9 (1997): 1735-1780.

[34] Cho, Kyunghyun et al. “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.” ArXiv abs/1406.1078 (2014): n. pag.

ABSTRACT

A Study For Scalable Dialog State Tracking in Multi-Domain

Lee Min-Ju
Department of Computer Science
Graduate School of
Sungshin University

In a Goal-oriented dialogue systems, dialog state tracking is a module involved in determining the action of a system by monitoring the user's intentions and extracting the conversation information called dialog states. As services using a goal-oriented dialogue system have become increasingly available in many ways, it has also become essential for dialog states tracking to support multi-domain applications.

Ontology-dependent state tracking, which includes all existing slots and values, has a high cost and increased complexity in building learning data increases the complexity of the model when extended to multiple domains. Therefore, it is important to apply a method of learning the conversation data itself without relying ontologies to build multi-domain state tracking. Text data, such as dialogue data, is considered a sequence modeling problem in machine learning. Therefore, it is

important to build a model that extracts information well within the data for each step of the sequence.

The recurrent neural network and convolution neural network are considered as main techniques for processing text data. Recently the Transformer and BERT using self-attention mechanism have shown good performance in machine reading and machine translation. In this paper, we conduct an experiment of tracking the state of conversation from conversation data by building a model using self-attention mechanism and recurrent neural network, and compare the accuracy with the model using only the recurrent neural network to find out the effect of self-attention. And the model uses pre-trained word embedding and copy mechanisms to mitigate Out-of-Vocabulary problems pointed out as problems in dialog state tracking. The model is developed in an end-to-end manner that tracks slots directly in conversations without going through the Natural Language Understanding (NLU) module. Experiments conduct tests on different versions of the model, from a basic structure encoder-decoder model to the proposed model, and measure performance through the experiment. Finally, in conclusion we provide analysis of the experimental results and suggest follow-up researches.