도메인 상태를 이용한 다중 도메인 대화 상태 추적

전현민[O], 이근배[†]

포항공과대학교 컴퓨터공학과[O†], 포항공과대학교 인공지능대학원[†]

jhm9507@postech.ac.kr[O], gblee@postech.ac.kr[†]

Multi Domain Dialog State Tracking using Domain State

Hyunmin Jeon[O], Geunbae Lee[†]

Computer Science and Engineering, Pohang University of Science and Technology[O†]

Graduate School of Artificial Intelligence, Pohang University of Science and Technology[†]

요 약

다중 도메인 목적 지향 대화에서 기존 딥 러닝을 이용한 대화 상태 추적(Dialog state tracking)은 여러 턴 동안 누적된 사용자와 시스템 간 대화를 입력 받아 슬롯 밸류(Slot value)를 추출하는 모델들이 연구되었다. 하지만 이 모델들은 대화가 길어질수록 연산량이 증가한다. 이에 본 논문에서는 다중 도메인 대화에서 누적된 대화의 history 없이 슬롯 밸류를 추출하는 방법을 제안한다. 하지만, 단순하게 history를 제거하고 현재 턴의 발화만 입력 받는 방법은 문맥 정보의 손실로 이어진다. 따라서 본 논문에서는 도메인 상태(Domain state)를 도입하여 매 턴 마다 대화 상태와 함께 추적하는 모델을 제안한다. 도메인 상태를 같이 추적함으로써 현재 어떠한 도메인에 대하여 대화가 진행되고 있는지를 파악한다. 또한, 함축된 문맥 정보를 담고 있는 이전 턴의 대화 상태와 도메인 상태를 현재 턴의 발화와 같이 입력 받아 정보의 손실을 줄였다. 대표적인 데이터 셋인 MultiWOZ 2.0과 MultiWOZ 2.1에서 실험한 결과, 대화의 history를 사용하지 않고도 대화 상태 추적에 있어 좋은 성능을 보이는 것을 확인하였다. 또한, 시스템 응답과 과거 발화에 대한 의존성을 제거하여 end-to-end 대화 시스템으로의 확장이 좀 더 용이할 것으로 기대된다.

주제어: 목적 지향 대화 시스템, 대화 상태 추적, BERT

1. 서론

목적 지향 대화는 턴이 지남에 따라 사용자의 요구가 갱신되기 때문에 매 턴 슬롯 밸류를 추적할 필요가 있다. 따라서, 대화 상태 추적은 목적 지향 대화 시스템에 있어 핵심적인 역할을 하며 최근 딥 러닝을 이용한 모델들이 많이 연구되고 있다. 기존의 모델들은 여러 턴에 걸친 문맥 정보를 이용하기 위하여 현재 턴의 사용자 발화 뿐만 아니라 지난 턴 동안 누적된 대화의 history를 입력 받는다. 하지만, 이 방법은 대화가 진행될수록 history가 길어지기 때문에 모델의 입력이 커지고 연산량이 증가한다. 또한, 시스템 응답을 입력 받음으로써 시스템 응답에 대한 의존성을 만든다. 대화 상태 추적 시스템 자체는 응답을 생성하지 않기 때문에 기존의 모델들은 데이터 셋 내의 ground truth 시스템 응답을 입력 받는다. 실제 대화 시스템에서는 ground truth 시스템 응답이 존재하지 않기 때문에 모델이 생성한 응답을 사용해야 한다. 즉, 시스템 응답의 오류가 다음 턴의 대화 상태 추적까지 전파된다. 이는 시스템 응답까지 생성하는 end-to-end 대화 시스템으로의 확장성을 떨어뜨린다. 따라서 본 논문에서는 대화의 history 없이 현재 턴의 사용자 발화만 입력 받아 효율적으로 대화 상태 추적을 수행하며 end-to-end 시스템으로의 확장성을 향상시킨 모델을 제안한다.

† 교신저자(Correspondig author)

2. 관련 연구

최근 딥 러닝을 이용한 대화 상태 추적이 활발하게 연구되고 있다. 다중 도메인 목적 지향 대화는 매 턴마다 슬롯 밸류를 예측할 필요가 있다. 밸류의 예측은 정해진 리스트를 참조하지 않고 단어를 디코딩하여 밸류를 생성하는 방법[5, 8, 9, 10], 슬롯 마다 사전에 정해진 밸류의 후보를 직접 참조하여 밸류를 추출하는 방법[3, 4, 11] 그리고 생성과 추출을 모두 사용하는 방법[6, 7]들이 연구되었다. 이전의 연구들은 LSTM[12], GRU[13]와 같은 RNN 계열의 NLU를 사용해왔다. 최근에는 BERT[14], GPT-2[15]와 같이 대량의 데이터에서 사전 학습된 NLU를 사용하는 모델들이 연구되고 있다.

슬롯의 밸류를 예측하는 것뿐만 아니라 게이트를 사용해서 현재 턴에 슬롯이 언급 되었는지 아닌지를 분류하는 방법들도 연구되었다. [5~7]은 게이트를 통해 슬롯을 빈 값으로 놔둘지, “don’t care”로 채울지, 모델이 추출하거나 생성한 밸류로 채울지 결정한다. [8]은 이전 턴의 대화 상태를 같이 입력 받고, 현재 턴의 대화 상태를 이전 턴과 비교하여 변경할지, 그대로 둘지, “don’t care”로 채울지, 빈 값으로 삭제할지를 결정한다. [9]는 [5~7]의 방법에 추가로 시스템의 이전 응답에서 밸류를 가져올지를 결정한다. [11]은 [8]과 비슷하게 이전 턴과 비교하여 변경할지, 그대로 둘지 결정한다.

본 논문에서는 [4, 7, 8, 9, 11]처럼 사전 학습된 BERT를 NLU로 사용하며, [8]과 같이 게이트를 통해 4가지 경우를 결정하여 대화 상태 추적을 수행한다.

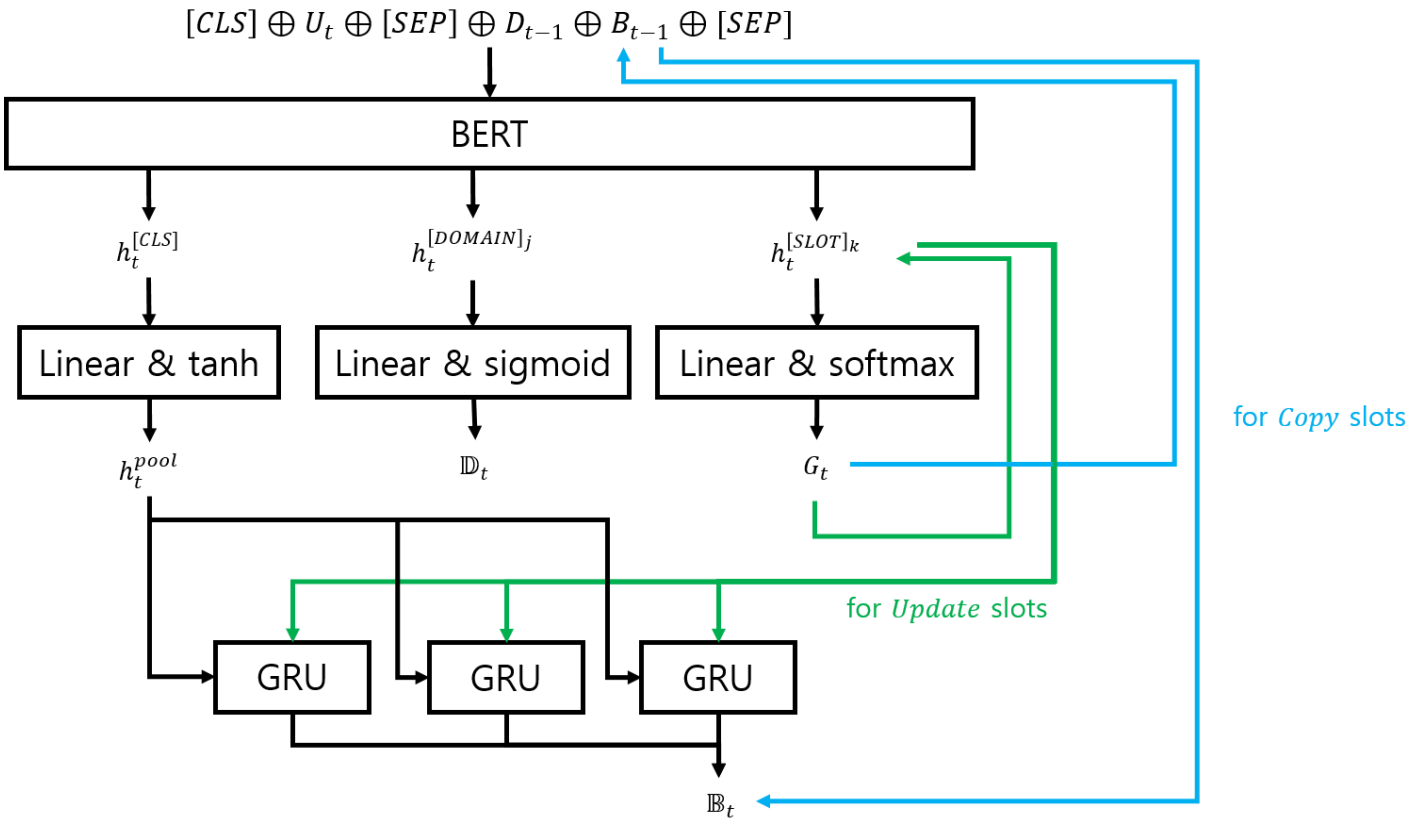

그림 1. 도메인 상태를 이용한 BERT 기반 다중 도메인 대화 상태 추적 모델의 구조

3. 도메인 상태를 이용한 BERT 기반 다중 도메인 대화 상태 추적

본 논문에서는 시스템 응답과 대화의 history 없이 현재 턴의 사용자 발화만을 사용하여 대화 상태 추적을 수행한다. 하지만, 다중 도메인 대화에서는 대화가 진행되면서 도메인이 변경되고 이전 대화의 내용들이 언급되는 경우가 있기 때문에 대화의 history를 사용하지 않으면 정보의 손실이 발생할 수 있다.

이러한 손실을 보완하기 위하여 도메인 상태를 도입하여 대화 상태와 함께 추적한다. 그림 1은 본 논문에서 제안한 대화 상태 추적 모델의 구조를 보여준다. 매 턴, BERT를 통해 J개의 $h_t^{[DOMAIN]_j}$와 K개의 $h_t^{[SLOT]_k}$를 계산하고 선형 계층을 통해 $\mathbb{D}_t$와 G_t를 생성한다. G_t가 *Update*로 분류된 슬롯들에 대해서만 GRU를 통해 밸류를 생성하고 Copy로 분류된 슬롯에 대해서는 이전 턴의 밸류를 복사한다. $1 \le j \le J$와 $1 \le k \le K$에 대하여 J와 K는 사전에 정의되고 고정된다.

대화 상태 턴 t의 대화 상태는 $\mathbb{B}_t = \{(S^k, V_t^k) \mid 1 \le k \le K\}$로 정의한다. 총 K개의 슬롯이 존재하며 각 슬롯 S^k는 턴 t에서 밸류 V_t^k를 갖는다. 슬롯은 도메인 이름과 슬롯의 이름을 결합하여 사용한다.

슬롯 게이트 턴 t의 슬롯 게이트는 $G_t = \{(S^k, g_t^k) \mid 1 \le k \le K,\ g \in \{Update, Copy, Delete, Dontcare\}\}$로 정의한다. K개의 슬롯에 대해 4 가지 범주로 분류한다.

도메인 상태 턴 t의 도메인 상태는 $\mathbb{D}_t = \{(d^j, a_t^j) \mid 1 \le j \le J,\ a \in \{0,1\}\}$로 정의한다. 총 J개의 도메인이 존재하며 각 도메인 d^j는 턴 t에서 활성화 되었는지 아닌지를 의미하는 이진 값 a_t^j를 갖는다.

3.1. 문맥 인코더

사전 학습된 BERT를 인코더로 사용하며 현재 턴의 사용자 발화 U_t, 이전 턴의 대화 상태 $\mathbb{B}_{t-1}$ 그리고 이전 턴의 도메인 상태 $\mathbb{D}_{t-1}$을 결합하여 입력한다. BERT에 입력하기 위하여 $\mathbb{B}_{t-1}$과 $\mathbb{D}_{t-1}$은 각각 $B_{t-1} = [SLOT] \oplus S^1 \oplus - \oplus V_{t-1}^1 \oplus \dots \oplus [SLOT] \oplus S^K \oplus - \oplus V_{t-1}^K$와 $D_{t-1} = [DOMAIN] \oplus d^1 \oplus a_{t-1}^1 \oplus \dots \oplus [DOMAIN] \oplus d^J \oplus a_{t-1}^J$로 변환한다. $[SLOT]$과 $[DOMAIN]$은 특수한 단어, $\oplus$는 concatenation이고 $-$는 슬롯과 밸류를 구분하기 위한 구분자의 역할을 한다.

턴 t에서 문맥 인코더의 입력은 $C_t = [CLS] \oplus U_t \oplus [SEP] \oplus D_{t-1} \oplus B_{t-1} \oplus [SEP]$로 정의한다. $[CLS]$와 $[SEP]$는 BERT의 입력으로 사용하기 위한 특수한 단어이다.

문맥 인코더의 C_t에 대한 출력은 $H_t = [h_t^1, \dots, h_t^{|C_t|}] \in \mathbb{R}^{|C_t| \times d}$로 표현된다. 임베딩의 크기와 은닉 계층의 크기는

d로 같다. H_t는 $[CLS]$, J개의 $[DOMAIN]$ 그리고 K개의 $[SLOT]$ 각각에 해당하는 출력 $h_t^{[CLS]}$, $h_t^{[DOMAIN]_j}$ 그리고 $h_t^{[SLOT]_k}$을 포함한다.

문맥 전체를 표현하기 위하여 출력 시퀀스 H_t를 학습 가능한 선형 계층 $W_{pool} \in \mathbb{R}^{d \times d}$과 $tanh$ 활성화 함수에 통과시켜 $h_t^{pool} = tanh\left(W_{pool} \cdot h_t^{[CLS]}\right)$를 계산한다.

3.2. 도메인 상태 생성기

매 턴 도메인 상태 $\mathbb{D}_t$를 생성하기 위하여 J개의 도메인들에 대해서 각 도메인이 현재 턴 t에 활성화 되었는지를 분류할 필요가 있다. J 개의 $[DOMAIN]$의 출력 $h_t^{[DOMAIN]_j}$을 학습 가능한 선형 계층 $W_{domain} \in \mathbb{R}^{1 \times d}$과 시그모이드 활성화 함수에 통과시켜 $p_t^{domain_j} = \sigma\left(W_{domain} \cdot h_t^{[DOMAIN]_j}\right)$을 계산한다. 그 후, 이진 분류를 통해 다음과 같이 $\mathbb{D}_t$를 생성한다.

$$\mathbb{D}_t = \{(d^j, a_t^j) \mid 1 \le j \le J\}$$

$$a_t^j = \begin{cases} 1, & p_t^{domain_j} \ge 0.5 \\ 0, & p_t^{domain_j} < 0.5 \end{cases}$$

3.3. 슬롯 게이트 분류기

매 턴 대화 상태 $\mathbb{B}_t$를 생성하기에 앞서 슬롯 게이트 G_t를 생성한다. K개의 슬롯에 대해서 $g_t^k \in \{Update, Copy, Delete, Dontcare\}$의 분류를 수행하기 위하여 $[SLOT]$의 출력 $h_t^{[SLOT]_k}$을 학습 가능한 선형 계층 $W_{gate} \in \mathbb{R}^{4 \times d}$과 소프트맥스 활성화 함수에 통과시켜 $p_t^{gate_k} = softmax(W_{gate} \cdot h_t^{[SLOT]_k})$를 계산한다. 그 후, 다중 분류를 통해 다음과 같이 G_t를 생성한다.

$$G_t = \{(S^k, g_t^k) \mid 1 \le k \le K\}$$

$$g_t^k = \begin{cases} Update, & argmax\left(p_t^{gate_k}\right) = 0 \\ Copy, & argmax\left(p_t^{gate_k}\right) = 1 \\ Delete, & argmax\left(p_t^{gate_k}\right) = 2 \\ Dontcare, & argmax\left(p_t^{gate_k}\right) = 3 \end{cases}$$

생성한 슬롯 게이트 g_t^k가 $Update$인 경우 V_t^k를 새로 생성, $Copy$인 경우 V_{t-1}^k를 복사하여 사용, $Delete$나 $Dontcare$인 경우 V_t^k에 각각 빈 값(None)과 "dontcare"를 할당한다.

3.4. 슬롯 밸류 생성기

매 턴 슬롯 게이트 g_t^k가 $Update$로 분류된 슬롯 S^k들에 대해 생성기를 통해 새로운 밸류 V_t^k를 생성한다. 슬롯의 밸류는 발화에 직접 언급되는 경우가 많기 때문에 발화의 단어를 직접 참조하는 복사 메커니즘을 사용한다.

생성기로는 GRU 디코더를 사용한다. GRU의 은닉 상태는 $r_t^{k,0} = h_t^{pool}$, 입력 임베딩은 $e_t^{k,1} = h_t^{[SLOT]_k}$로 초기화한다. $g_t^k = Update$인 k에 대해 다음과 같은 방법으로 $[EOS]$ 단어가 만들어질 때까지 밸류를 디코딩한다.

$$r_t^{k,i} = GRU\left(r_t^{k,i-1}, e_t^{k,i}\right) \in \mathbb{R}^d$$

$$p_{t,vocab}^{k,i} = softmax\left(E \cdot r_t^{k,i}\right) \in \mathbb{R}^{d_{vocab}}$$

$E \in \mathbb{R}^{d_{vocab} \times d}$는 임베딩 행렬이고 d_{vocab}은 사전의 크기이다. Pointer-Generator 복사 메커니즘[16]을 사용하여 생성 확률과 복사 확률의 가중 합을 통해 다음 단어의 확률 분포를 계산한다.

$$p_{t,copy}^{k,i} = softmax\left(H_t \cdot r_t^{k,i}\right) \in \mathbb{R}^{|C_t|}$$

$$s_t^{k,i} = p_{t,copy}^{k,i} \cdot H_t \in \mathbb{R}^d$$

$$z_t^{k,i} = \sigma\left(W_{weight} \cdot \left[r_t^{k,i} \oplus e_t^{k,i} \oplus s_t^{k,i}\right]\right) \in \mathbb{R}^1$$

$W_{weight} \in \mathbb{R}^{1 \times 3d}$는 학습 가능한 선형 계층, $s_t^{k,i}$는 어텐션(attention)을 통해 계산한 문맥 벡터 그리고 $z_t^{k,i}$는 생성 확률과 복사 확률을 합하기 위한 가중치이다.

$$p_{t,value}^{k,i} = z_t^{k,i} \times p_{t,vocab}^{k,i} + \left(1 - z_t^{k,i}\right) \times p_{t,copy}^{k,i} \in \mathbb{R}^{d_{vocab}}$$

3.5. 손실 함수

대화 상태 추적 모델의 학습 시 매 턴마다 도메인 상태 생성기, 슬롯 게이트 분류기 그리고 슬롯 밸류 생성기를 함께 최적화한다. $y_t^{domain_j} \in \{0,1\}$는 j번째 도메인이 t턴에 활성화 되었는지에 대한 라벨이며, 도메인 상태 생성기의 손실 함수 L_t^{domain}은 다음과 같이 계산한다.

$$L_t^{domain} = \frac{1}{J}\sum_{j=1}^{J}\Big\{-\left(y_t^{domain_j}\right) \cdot log\left(p_t^{domain_j}\right) - \left(1 - y_t^{domain_j}\right) \cdot log\left(1 - p_t^{domain_j}\right)\Big\}$$

$y_t^{gate_k}$는 t턴에 k번째 슬롯의 게이트에 대한 라벨 one-hot 벡터이며, 슬롯 게이트 생성기의 손실 함수 L_t^{gate}는 다음과 같이 계산한다.

$$L_t^{gate} = \frac{1}{K}\sum_{k=1}^{K} -\left(y_t^{gate_k}\right)^T \cdot log\left(p_t^{gate_k}\right)$$

$M = \{k \mid g_t^k = Update\}$일 때, $y_t^{value_{m,i}}$는 t턴에 m번째 슬롯의 밸류의 i번째 단어에 대한 라벨 one-hot 벡터이며

표 1. MultiWOZ 데이터 셋의 예시

사용자: Hi, I am currently planning to come to Cambridge, and I was looking for a relatively inexpensive place to eat in the centre. What would you suggest? **대화 상태:** $\{restaurant: \{price\ range = cheap, area = centre\}\}$
시스템: The Gardenia is an excellent and cheap place in the center of town, serving Mediterranean cuisine. Would you like a table there? **사용자:** Yes, please book a table for 1 people at 19:15 on Monday. **대화 상태:** $\{restaurant: \{price\ range = cheap, area = centre, time = 19{:}15, day = monday, people = 1\}\}$
시스템: Booking was successful. The table will be reserved for 15 minutes. Reference number is: IU63HAEN. Is there anything else I can help you with today? **사용자:** Yes. I need a place to stay in **the same part** of town. It must have free parking. **대화 상태:** $\{restaurant: \{price\ range = cheap, area = centre, time = 19{:}15, day = monday, people = 1\}, hotel: \{area = \boldsymbol{centre}, parking = yes\}\}$

슬롯 밸류 생성기의 손실 함수 L_t^{value}와 모델의 최종 손실 함수 L_t는 다음과 같이 계산한다

$$L_t^{value} = \frac{1}{|M| \times |V_t^m|} \sum_{m \in M} \sum_{i=1}^{|V_t^m|} -\left(y_t^{value_{m,i}}\right)^T \cdot \log\left(p_t^{value_{m,i}}\right)$$

$$L_t = L_t^{domain} + L_t^{gate} + L_t^{value}$$

4. 실험 및 평가

4.1. 데이터 구성

Multi domain Wizard-of-OZ(MultiWOZ)는 Cambridge의 여행객과 가이드의 가상의 대화를 다루는 다중 도메인 목적 지향 대화 시스템의 대표적인 데이터 셋이다. 본 논문에서는 MultiWOZ 2.0[1]과 오류를 개선한 MultiWOZ 2.1[2] 두가지 데이터 셋으로 실험을 진행하였다. 약 1만개의 대화로 이루어져 있으며 $restaurant$, $hotel$, $attraction$, $train$, $taxi$, $police$, $hospital$ 7개의 도메인이 존재한다. $police$와 $hospital$ 두 도메인은 학습 데이터에만 존재하고 검증과 평가 데이터에는 존재하지 않아 평가는 5개의 도메인에 대해서만 진행하였다. $police$와 $hospital$을 제외하고 총 30개의 (도메인, 슬롯) 쌍들이 존재한다. 표 1에서, 사용자가 $hotel$의 $area$에 대해 직접적으로 명시하지는 않았지만 "the same part"라며 $restaurant$과 같은 $area$를 간접적으로 요구하였기 때문에 대화 상태에서 $hotel$의 $area$도 $centre$가 된다. 이처럼 한 대화에 여러 도메인이 등장하며 사용자의 간접적인 요구를 추론하는 대화 상태 추적을 필요로 한다.

4.2. 실험 구성 및 평가 방법

본 논문에서는 사전 학습된 BERT-base-uncased[1] 모델을 문맥 인코더로 사용하였다. 도메인 상태 생성기와 슬롯 게이트 분류기로 한 계층의 선형 모델을 사용하였고 슬롯 밸류 생성기로 한 계층의 GRU를 사용하였다. 선형 모델과 GRU의 은닉 계층의 크기는 $d = 768$로 설정하였다. 학습 과정에서 BERT는 fine-tuning 하였고 문맥 인코더와 슬롯 밸류 생성기의 임베딩 계층은 공유하였다. 모델의 학습률(Learning rate)은 3e-5, 드랍아웃(Dropout)은 0.2로 설정하였고 Adam[17]을 사용해 파라미터의 최적화를 진행하였다. 검증 성능이 5 에폭(epoch)동안 갱신되지 않으면 학습을 종료하는 조기 종료(early stopping) 방법을 사용하였고 4 에폭 동안 검증 성능이 갱신되지 않을 때 마다 학습률을 절반으로 감소시키는 스케줄링 방법을 사용하였다.

표 2. 시스템 응답의 밸류를 추적하는 예시

사용자: I'm looking for a hotel in the south. **대화 상태:** $\{hotel: \{area = south\}\}$
시스템: I have a long list. Can you specify the kind of food you want? **사용자:** I am looking for a hotel with free wifi. It should be in the cheap price range. **대화 상태:** $\{hotel: \{area = south, price\ range = cheap, internet = yes\}\}$
시스템: I recommend Rosa's Bed and Breakfast, it is in the south, offers internet and is inexpensive. Would you like me to book it for you? **사용자:** Yes please. I would like it for 8 people for 5 nights starting from Monday. **대화 상태:** $\{hotel: \{area = south, price\ range = cheap, internet = yes, people = 8, stay = 5, day = monday, \boldsymbol{name = Rosa's\ Bed\ and\ Breakfast}\}\}$

상태 순서 혼합 이전 턴의 도메인 상태 D_{t-1}와 대화 상태 B_{t-1}의 도메인과 슬롯의 순서가 고정되면 학습 과정에서 도메인과 슬롯 자체의 의미뿐 아니라 위치가 영향을 줄 수 있다. 따라서, 도메인과 슬롯 그 자체의 의미를 보다

[1] https://github.com/huggingface/transformers

표 3. 데이터 셋의 실험 결과

	History	Ontology	MultiWOZ 2.0		MultiWOZ 2.1	
			Joint	Slot	Joint	Slot
GLAD[3]	✓	✓	35.57	95.44		
SUMBT[4]	×	✓	46.65	96.44		
TRADE[5]	✓	×	48.62	96.92	45.60	
DSTQA[6]	✓	✓	51.44	97.24	51.17	97.21
DST-Picklist[7]	✓	✓			53.30	
SOM-DST[8]	✓	×	52.32		53.68	
TripPy[9]	✓	×			55.29	
Simple-TOD[10]	✓	×			55.72	
CHAN[11]	✓	✓	52.68	97.69	58.55	98.14
Ours	×	×	48.82	96.89	54.51	97.49
Ours*	×	×	54.37	97.32	61.03	97.90

효율적으로 학습시키기 위하여 문맥을 인코더에 입력하기 전에 50%의 확률로 B_{t-1}와 D_{t-1} 내부의 순서를 무작위로 섞는 방법을 사용하였다.

시스템 응답의 *name* 슬롯 MultiWOZ 데이터 셋은 매 턴마다 누적된 대화 상태의 라벨을 가지며 *hotel*, *restaurant*, *attraction* 세 도메인은 *name*이라는 슬롯을 포함한다. 표 2에서 볼 수 있듯이, *name* 슬롯은 사용자의 발화뿐 아니라 시스템 응답의 밸류도 추적하도록 라벨링 되어 있다. 시스템이 사용자에게 *hotel*, *restaurant*, *attraction* 도메인의 특정 개체를 추천하면서 *name*을 알려주는 경우, 추론한 대화 상태로 DB에 접근하여 사용자의 요구 사항을 만족하는 개체의 정보를 가져오는 절차가 선행되어야 한다. 즉, 시스템 응답에 어떤 개체의 *name*이 등장한다면 그 *name*에 해당하는 값은 이미 시스템이 알고 있는 정보이기 때문에 다음 턴에 대화 상태 추적으로 해당 *name*의 밸류를 추론 하는 것은 자연스럽지 않은 절차이며 시스템 응답에 대한 불필요한 의존성을 만든다. 본 논문에서는 표 2의 예시와 같이 시스템 응답으로부터 나오는 *name* 슬롯의 밸류를 규칙 기반으로 채워 넣을 수 있다고 간주하여 라벨을 삭제하고 모델을 학습함으로써 시스템 응답에 대한 불필요한 의존성을 제거하였다. 단, 기존의 연구들과 같은 평가 기준으로 성능을 비교하기 위하여 현재 턴의 사용자 발화뿐 아니라 이전의 시스템 응답을 함께 입력 받는 모델의 실험도 진행하였다.

Zero-shot 도메인 본 논문에서 제안한 모델이 새로운 도메인에 대해서도 대화 상태 추적을 잘 수행하는지 검증하기 위하여 zero-shot 도메인에 대해서 실험을 진행하였다. 검증과 평가 데이터에 존재하는 5개의 도메인 중 하나를 새로운 도메인으로 간주하여 학습 데이터에서 해당 도메인이 포함된 대화를 모두 제외하고 해당 도메인에 대해 성능을 검증하는 방법으로 5개의 도메인에 대해 각각 실험을 진행하였다.

표 4. zero-shot 도메인의 실험 결과

		MultiWOZ 2.0		MultiWOZ 2.1	
		Joint	Slot	Joint	Slot
Hotel	TRADE	13.70	65.32		
	Ours	18.12	65.97	19.34	64.23
	Ours*	19.98	67.92	20.62	66.36
Train	TRADE	22.37	49.31		
	Ours	24.27	50.69	24.21	53.35
	Ours*	24.21	51.82	23.89	53.47
Attraction	TRADE	19.87	55.53		
	Ours	24.57	57.21	30.95	57.96
	Ours*	24.76	60.93	31.23	64.34
Restaurant	TRADE	11.52	53.43		
	Ours	17.28	57.49	20.15	58.84
	Ours*	17.69	59.54	23.31	64.57
Taxi	TRADE	60.58	73.92		
	Ours	60.83	73.40	60.83	74.78
	Ours*	60.94	73.60	61.06	74.87

4.3. 실험 결과

Slot 정확도와 Joint 정확도 두 개의 성능 평가 지표를 사용한다. Slot 정확도는 슬롯 30개의 밸류가 일치하는지를 독립적으로 평가하고 Joint 정확도는 슬롯 30개의 밸류가 모두 일치하는지를 평가한다.

표 3은 MultiWOZ 2.0과 MultiWOZ 2.1 데이터 셋에서 기존의 모델들과 본 논문에서 제안한 모델의 Joint 정확도와 Slot 정확도 수치를 보여준다. **History**는 누적된 이전 대화를 사용하는지, **Ontology**는 성능 평가 시 사전 정의된 밸류의 후보들을 직접 참조하는지 여부를 의미한다. 대화 상태 라벨의 오류를 개선한 MultiWOZ 2.1에서 훨씬 높은 Joint 정확도를 달성하였다.

Ours*는 end-to-end 대화 시스템으로의 확장성을 고려하여 시스템 발화에 대한 의존성을 제거하고 시스템의 응답에만 등장한 *name* 슬롯의 밸류를 규칙 기반으로 채워 넣을 수 있다고 가정한 모델이다. Ours는 공정한 평가를 위하여 이전 턴의 시스템 응답을 함께 입력 받아 기존 모델들과 성능 평가 방법을 일치시킨 모델이다.

표 4는 zero-shot 도메인에서 기존 모델과 본 논문에서 제안한 모델의 Joint 정확도와 Slot 정확도 수치를 보여준다. 슬롯이 비교적 간단한 taxi 도메인에서 가장 높은 Joint 정확도를 달성하였다.

5. 결론

본 논문에서는 다중 도메인 목적 지향 대화에서 도메인 상태를 활용하여 시스템 응답과 이전 대화의 history 없이 대화 상태 추적을 수행하는 모델을 제안하였다. 도

메인 상태를 도입함으로써 대화의 history가 아닌 간소화된 입력만으로 문맥 정보의 손실을 최소화하고 효율성을 높이면서 높은 Joint 정확도를 달성할 수 있었다. 또한, 시스템 응답에 대한 의존성을 제거함으로써 좀 더 실제에 가까운 대화 상태 추적을 수행할 수 있었다.

향후 연구로 대화 상태 추적 시스템을 end-to-end 대화 시스템으로 확장하고자 한다. 모델이 생성한 시스템 응답의 오류가 다음 턴의 대화 상태 추적에 전달되는 것을 방지함으로써 좀 더 안정적인 end-to-end 대화를 수행할 수 있을 것이다.

사사

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음 (IITP-2020-2020-0-01789)

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-01906, 인공지능대학원지원(포항공과대학교))

참고문헌

[1] Budzianowski, Pawel, et al. "MultiWOZ-A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling." Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. 2018.

[2] Eric, Mihail, et al. "MultiWOZ 2.1: A Consolidated Multi-Domain Dialogue Dataset with State Corrections and State Tracking Baselines. Proceedings of The 12th Language Resources and Evaluation Conference. 2020.

[3] Zhong, Victor, Caiming Xiong, and Richard Socher. "Global-locally self-attentive dialogue state tracker." arXiv preprint arXiv:1805.09655 (2018).

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

[5] Wu, Chien-Shen, et al. "Transferable multi-domain state generator for task-oriented dialogue systems." Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019.

[6] Zhou, Li, and Kevin Small. "Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering." arXiv preprint arXiv:1911.06192 (2019).

[7] Zhang, Jian-Guo, et al. "Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking." arXiv preprint arXiv:1910.03544 (2019).

[8] Kim, Sungdong, et al. "Efficient dialogue state tracking by selectively overwriting memory." Proceedings of the 58th Annual Meetings of the Association for Computational Linguistics (2020).

[9] Heck, Michael, et al. "TripPy: A Triple Copy Strategy for Value Independent Neural Dialog State Tracking." arXiv preprint arXiv:2005.02877 (2020).

[10] Hosseini-Asl, Ehsan, et al. "A simple language model for task-oriented dialogue." arXiv preprint arXiv:2005.00796 (2020).

[11] Shan, Yong, et al. "A Contextual Hierarchical Attention Network with Adaptive Objective for Dialogue State Tracking." Proceedings of the 58th Annual Meetings of the Association for Computational Linguistics (2020).

[12] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

[13] Cho, Kyunghyun, et al. "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches." Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation. 2014.

[14] Devlin, Jacob, et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). 2019.

[15] Radford, Alec, et al. "Language models are unsupervised multitask learners." OpenAI Blog 1.8 (2019): 9.

[16] See, Abigail, Peter J. Liu, and Christopher D. Manning. "Get To The Point: Summarization with Pointer-Generator Networks." Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2017.

[17] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).