포인터 생성 네트워크를 이용한 패러프레이즈 생성

박다솔[1O], 김영길[2#], 차정원[1*]

창원대학교[1], 한국전자통신연구원[2]

{dasol_p [O],jcha[*]}@changwon.ac.kr, kimyk[#]@etri.re.kr

Generation Paraphrase using Pointer Generation Network

Da-Sol Park[1O] , Young-kil Kim[2#], Jeong-Won Cha[1*]

Changwon National University[1], Electronic Telecommunications Research Institute[2]

요 약

다양한 발화를 모델링하는 요구는 자연어 처리 분야에서 꾸준히 있었으며 단어, 구 또는 문장과 동등한 의미 콘텐츠를 자동으로 식별하고 생성하는 것은 자연어 처리의 중요한 부분이다. 본 논문에서는 포인터 생성 네트워크(Pointer Generate Nework)를 이용하여 패러프레이즈 생성 모델을 제안한다. 제안한 모델의 성능을 측정하기 위해 사람이 직접 구축한 유사 문장 코퍼스를 이용하였으며, 토큰 단위의 BLEU-4 0.250, ROUGE_L 0.455, CIDEr 2.190의 성능을 보였다. 하지만 입력 문장과 동일한 문장을 출력하는 문제점이 존재하여 빔서치(beam search)를 적용하여 입력 문장과 비교하여 생성 문장을 선택하는 방식을 적용하였다. 입력 문장과 동일한 문장을 제외한 문장으로 평가를 진행했으며, 토큰 단위의 BLEU-4 0.234, ROUGE_L 0.459, CIDEr 2.041의 성능을 보였으나, 패러프레이즈 생성 데이터 양이 크게 증가하였다. 본 연구는 문장 간의 의미적으로 동일한 정보를 정확하게 추출할 수 있게 됨으로써 정보 추출, 온톨로지 생성에 도움이 될 것이다. 또한 이러한 기법이 챗봇에서 사용자의 의도 탐지 및 MRC와 같은 자연어 처리의 여러 분야에 유용한 자원으로 사용될 것이다.

주제어: 패러프레이즈, 문장 생성, 기계학습, 자연어처리

1. 서론

패러프레이즈는 텍스트간 의미적으로 거의 동일한 정보를 제공하기 위해 선택할 수 있는 다양한 표현 방법으로 이루어진 관계를 말한다. 일반적으로 텍스트 문장(T,text)과 가설 문장(H, Hypothesis)이 본질적으로 동일한 의미를 가질 때, “텍스트 문장(T)과 가설 문장(H)은 패러프레이즈이다.” 라고 한다. 사전적 정의는 더 명확하게 하기 위해 다른 단어 및 구를 사용하여 글을 쓰거나 말하는 것의 의미를 표현하는 것이다.

아래는 패러프레이즈의 3가지 예시를 보여준다. (1)은 단어가 다른 단어로 치환되는 경우이다. ‘돌아서서’ 가 ‘뒤돌아서’ 로 치환되었고, ‘산을 내려가다’ 가 ‘하산하다’ 로 치환되었다. (2)는 문장 구조의 치환되는 경우이다. ‘A는 B이다’ 의 문장 구조가 ‘B는 A이다’ 로 치환되었다. (3)은 (1)과 (2)과 동시에 일어나는 경우이다.

그러고는 돌아서서 산을 내려가기 시작했다.
→ 그러고는 뒤돌아서 하산하기 시작했다. (1)

경수는 가로수 아래에 서있다.
→ 가로수 아래에 서있는 경수이다. (2)

얼마 후 그녀는 내 방에서 나갔다.
→ 그녀는 얼마 지나지 않아 내 방 밖으로 나갔다.(3)

최근 패러프레이즈 생성은 다양한 자연어 처리 접근법으로 연구되었다. 규칙 기반 접근 방식[1,2]과 데이터 기반 방식[3], 언어 번역 작업으로 처리한 방식[4,5], 그중 2개 국어를 사용하는 말뭉치를 사용하여 수행[6,7]이 있다. 최근 가장 일반적인 접근 방식은 강화 학습 방식[8], 시퀀스-투-시퀀스(sequence-to-sequence, seq2seq) 모델[9], 비지도 접근법[10]이 있다.

자연어 처리 분야의 여러 태스크에서 특정 의미에 대한 서로 다른 표층적 언어 표현들이 성능 저하의 중요한 요인 중 하나이다[11,12]. 따라서 패러프레이즈에 대한 중요성이 강조되고 있으며 최근에 독립적인 분야로 꾸준히 연구되고 있다. 문장 쌍을 사용하고 Sequence-to-Sequence와 어텐션(attention)을 이용하여 패러프레이즈를 생성할 경우, 사전 내 OOV(out-of-vocabulary)으로 표시된 비이상적인 단어들로 인해 부정확한 패러프레이즈 결과를 불러올 가능성이 있다.

이러한 문제를 해결하기 위해 본 논문에서는 기존의 Sequence-to-Sequence 구조 위에 포인터 생성 네트워크(Pointer Generation Network)를 추가한 패러프레이즈 생성 모델을 제안한다. 패러프레이즈는 입력 문장의 주요 컨텐츠를 그대로 사용하되 어휘를 변경하거나 구조 변경이 발생해야 한다. 그래서 우리는 주요 컨텐츠를 보존하기 위해 복사 매커니즘(Copy Mechanism)을 포인터 네트워크로 구현하였다. 제안 모델의 접근법은 Pointer-

Generator Network는 입력 문장에서 단어를 가져올지 생성할지를 결정하고, 단어의 반복을 피하고자 이전 단어들의 분포를 고려하여 단어를 생성하고자 한다.

2. 제안 모델

그림 1은 본 논문의 제안 모델 구조이다. 제안 모델 구조는 인코더의 입력 단어들을 가져오고 대체할 단어는 학습된 사전에서 가져오는 구조이며 기존 Sequence-to-Sequence에서 OOV 문제는 입력 문장의 단어를 가져옴으로써 해결한다.

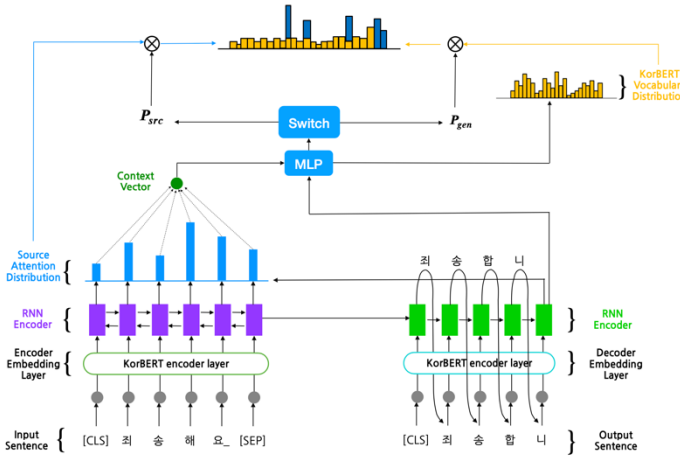

그림 1 제안 모델 구조

인코더 RNN은 워드피스(wordpeiece) 단위로 토큰화(tokenization)이 되어 있는 입력 문장을 읽는 모듈이고, 양방향의 순서를 고려한 양방향 LSTM(bi-directional LSTM)을 이용한다. 디코더 LSTM은 입력 문장의 패러프레이즈 문장의 워드피스 형태로 결과값을 반환한다.

소스 주의 분포 (Source Attention Distribution)는 입력 문장으로부터 다음 워드피스를 생성할 때, 입력 문장의 단어에 대한 확률인 주의 분포(attention distribution)은 인코더와 디코더의 은닉 상태(hidden state)를 입력 값으로 하여 아래와 같은 함수를 통해 표현된다. 이는 직관적으로 해당 네트워크가 다음 단어를 생성 시 원문 텍스트 중에 어떠한 단어를 집중해야 하는지에 대한 표현이다. 문맥 벡터 (context vector)는 집중 분포와 인코더의 은닉 상태를 이용한다.

$$e_i^t = v^T \tanh(W_h h_i + W_s S_t + b_{attn}) \quad (4)$$
$$a^t = softmax(e^t) \quad (5)$$
$$h_t^* = \sum_i a_i^t h_i \quad (6)$$

여기서 h_i는 인코더의 은닉 상태이며, S_t는 디코더의 은닉 상태이고, a^t는 주의 분포이다. v, W_h, W_s는 학습되는 파라미터이다. 식 (4), (5)를 통해서 주의 분포가 생성되고, 식 (6)을 통해서 문맥 벡터가 생성된다.

$$P_{vocab} = softmax(V'(V[s_t, h_t^*] + b) + b' \quad (7)$$

$$\hat{y} = P_{vocab}(w) \quad (8)$$
$$loss_t = -logP(w_t^*) \quad (9)$$
$$loss = \frac{1}{T}\sum_{t=0}^{T} loss_t \quad (10)$$

식 (7)에서 인코더 은닉 상태인 s_t와 문맥 벡터를 연결하여 2층의 선형층을 통해 P_{vocab}의 분포를 생성한다. V, V', b, b'는 학습되는 파라미터이다. P_{vocab}는 사전의 모든 단어에 대한 확률 분포이며, 식 (8)과 같이 단어 w를 예측한 최종 결과를 제공한다. 이 모델의 손실 함수는 음의 로그 우도(Negative log-likelihood)를 사용하고, 식 (9)은 t 시간의 정답 단어이고, 전체 시퀀스에 대한 손실함수는 식 (10)과 같다.

포인터 생성 네트워크는 어휘 분포 최종적으로 문맥 벡터와 디코더 은닉 상태의 출력벡터를 연결(concatenate)하여 2개의 MLP(Multi Layer Perceptron)을 통해 단어의 분포를 표현한다. 포인터 생성 네트워크는 기존의 입력 문장에서의 단어를 $1 - P_{gen}$의 확률로 인용하거나, 단어 사전에서 새로운 단어를 P_{gen}의 확률로 생성할 수 있다.

$$p_{gen} = \sigma(W_{h^*}^T h_t^* + W_s^T s_t + W_x^T x_t + b_{ptr}), p_{gen} \in [0,1] \quad (11)$$
$$p(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \quad (12)$$

여기서 s_t는 인코더 상태이고, x_t는 디코더 입력이고, h_t는 문맥 벡터이다. 식 (11)과 (12)을 통해서 해당 시점의 단어를 생성한다. 예를 들어, 생성 확률이 0에 가까우면 입력 문장으로부터 재현된 단어들을 통해 패러프레이즈 문장이 생성되고 이와 반대로 생성 확률이 1에 가까우면 단어 사전의 단어들을 통해 패러프레이즈 문장이 생성되도록 하는 것이다.

또, Sequence-to-Sequence에서 자주 발생하는 문제 중 하나인 생성된 단어의 반복을 피하고자 범위 매커니즘(Coverage mechanism)[13]을 사용한다. 범위 매커니즘은 이전까지의 주의 분포를 더하여 현재 주의 분포를 생성할 때 같은 단어가 반복되는 것을 피한다.

$$c^t = \sum_{t'=0}^{t-1} a^{t'} \quad (13)$$
$$e_i^t = v^T \tanh(W_h h_i + W_s s_t + w_c c_i^t + b_{attn}) \quad (14)$$
$$cov\_loss_t = \sum_i \min(a_i^t, c_i^t) \quad (15)$$
$$loss_t = -logP(w_t^*) + \gamma cov\_loss_t \quad (16)$$

식 (13)에서 c^t는 t 시간 만큼의 주의 분포를 합한 범위 벡터(Coverage vector)이다. e_i^t는 범위 벡터가 포함된 디 코더의 t 시간의 주의 가중치이다. cov_loss_t는 주의 분포와 범위 벡터 간의 겹치는 부분을 전부 합하여 손실 함수로 정의하는데 균일한 단어 분포에 대한 페널티를 의미한다. 최종 손실 함수 $loss_t$는 cov_loss_t에 하이퍼 파리미터 γ를 곱하고 식 (9)과 더한다.

3. 실험 설정

패러프레이즈 데이터는 국립국어원에서 공개한 모두의

말뭉치[14]의 유사 문장 말뭉치를 일부 사용하였으며, 표 1은 하나의 패러프레이즈 데이터 예시이다. 데이터는 한 문장에 대해 5개의 패러프레이즈 문장이 하나의 세트로 구성되어 있다. 입력 문장과 패러프레이즈 문장 1의 '동안에 → 사이에', 패러프레이즈 문장 2의 '1분 → 60초' 등 단어 치환이 나타난 경우가 있으며, 입력 문장과 패러프레이즈 문장 3의 단어의 순서를 변경하는 것이 특징이다. 패러프레이즈 데이터 한 세트가 6문장이며, 이를 이용하여 총 3쌍의 병렬 데이터로 생성하며, 생성된 패러프레이즈 데이터는 표 2와 같다. 표 2 내 괄호는 표 1의 분류를 명시하였다.

학습 데이터는 25,000개 데이터셋을 사용했고, 실제 사용한 문장 쌍은 75,000쌍이다. 검증 데이터는 7,710쌍이고 평가 데이터는 총 600쌍을 이용하였다. Optimizer는 Adam을 이용하고 학습률은 0.001, 배치 사이즈는 32, 드롭 아웃(dropout)은 0.33, 인코더의 은닉층의 크기는 512, 디코더 은닉층의 크기는 1024이다.

표 1 패러프레이즈 데이터 예시

분류	예시 문장
입력 문장	'태극기 에펠탑' 은 가슴을 벅차오르게 했다.
패러프레이즈 문장 1	'태극기 에펠탑' 은 벅찬 감동을 안겨 주었다
패러프레이즈 문장 2	'태극기 에펠탑' 을 보고 감동을 받았다.
패러프레이즈 문장 3	'태극기 에펠탑' 을 보니 마음이 설레었다.
패러프레이즈 문장 4	'태극기 에펠탑' 을 읽고 감동을 받았다
패러프레이즈 문장 5	가슴을 벅차오르게 한 것은 '태극기 에펠탑' 이었다.

표 2 생성된 패러프레이즈 데이터 예시

입력 문장	출력 문장
'태극기 에펠탑' 은 가슴을 벅차오르게 했다. (입력 문장)	'태극기 에펠탑' 은 벅찬 감동을 안겨 주었다 (패러프레이즈 문장 1)
'태극기 에펠탑' 을 보고 감동을 받았다. (패러프레이즈 문장 2)	'태극기 에펠탑' 을 보니 마음이 설레었다. (패러프레이즈 문장 3)
'태극기 에펠탑' 을 읽고 감동을 받았다 (패러프레이즈 문장 4)	가슴을 벅차오르게 한 것은 '태극기 에펠탑' 이었다. (패러프레이즈 문장 5)

표 3은 어절 단위 성능의 문제점을 보여준다. 정답 문장을 기준으로 예측 문장의 어절 단위로 성능을 측정한다면, 예측 문장은 정답 문장과 동일한 어절이 아니므로 성능에 좋은 영향을 미치지 못한다. 하지만 예측 문장의 "가입했고" 라는 어절에서 정답 문장의 "(등록)했고" 에 대한 토큰은 동일하기 때문에 이러한 경우도 성능 측정에 고려되어야 한다고 판단하였다. 이를 토대로 토큰 단위 또한 성능을 측정하였다. 표 4는 실험에 대한 성능이며, 어절과 토큰 단위로 각각 평가하였다.

표 3 어절 단위의 성능 측정의 문제 예제

분류	해당 문장
입력 문장	오늘 보험에 가입했으며,혜택 또한 받았다.
정답 문장	오늘 보험에 등록**했고**,혜택 또한 받았다.
예측 문장	오늘 보험에 가입**했고**,혜택 또한 받았다.

표 4 실험 성능표

	어절 단위	토큰 단위
BLEU 1	0.337	0.512
BLEU 2	0.207	0.400
BLEU 3	0.129	0.316
BLEU 4	0.083	0.250
ROUGE_L	0.378	0.455
CIDEr	1.149	2.190

정성 평가는 총 4가지에 대한 항목으로 진행하고, 사람이 직접 수기로 평가 코퍼스에 대한 평가를 진행하였다. 표 5는 정성 평가의 기준과 그에 대한 예시이다. 입력과 동일한 문장은 출력 문장이 입력 문장과 동일하여 문장이 변경되지 않은 문장이고, 에러 문장은 문장이 성립이 되지 않거나, 입력 문장과 의미가 다른 문장이다. 단어만 변경된 문장은 입력 문장과 전체적인 구조는 같으나 특정 단어가 바뀌는 문장이며, 문장 구조 변경 문장은 입력 문장과 달리 단어의 어순이 바뀌는 문장이고, 단어 변경 또한 포함한다. 표 6은 정성 평가의 결과이다.

표 5 정성적 평가 기준 및 예시

분류	문장 분류	예시
입력과 동일한 문장	입력	그래서 메주와 첼리스트에서는 밥상을 모두 된장 음식으로 낸다.
	출력	그래서 메주와 첼리스트에서는 밥상을 모두 된장 음식으로 낸다.
에러 문장	입력	개최 20여 분 전, 그러나 협약식은 돌연 취소되었다.
	출력	개최 20여 분 전, 갑작스레 협약식은 취소
단어만 변경된 문장	입력	롯데제과가 가장 힘을 쏟고 있는 수출 국가는 인도다.
	출력	롯데제과가 제일 힘을 쏟고 있는 수출 국가는 인도다.
문장 구조 변경 문장	입력	2009년부터 현대모비스는 친환경차 사업에 주력하였다.
	출력	현대모비스는 2009년부터 친환경차 사업에 주력하였다.

표 6 정성 평가 결과

기준 분류	문장 수(비율)
입력과 동일한 문장	342(57%)
에러 문장	57(9.5%)
단어만 변경된 문장	99(16.5%)
문장 구조 변경 문장	102(17%)

정성 평가 결과를 분석했을 때 입력과 동일한 문장을 출력하는 경우가 57%의 비율을 보였다. 우리는 입력 문장과 동일한 문장을 출력하는 경우를 줄이기 위해서 그 다음 확률이 높은 문장을 생성하고자 빔서치(beam search)를 적용하여 진행해본다.

빔서치는 최고우선탐색(Best-First Search) 기법을 기본으로 하되 기억해야 하는 노드 수를 제한해 효율성을 높인 방식이다. 한 단계에서 단어를 빔 크기(beam size)인 n개 예측 후 조합한다. 예를 들어, 한 단계에서 3개의 단어를 예측, 각각의 단어를 이용하여 다음 단어를 3개 예측한다. 이를 반복하여 조합의 가능성이 가장 높은 것을 선택하는 방식이다. 우리는 빔 크기를 3으로 설정하였다.

빔 크기를 3으로 하여 입력 문장에 대해 Best 1,2,3의 문장을 출력한다. 출력 문장 중 Best 1과 입력 문장이 동일하다면 Best 2 문장을 선택하고, Best 2 문장 또한 입력 문장과 동일하다면 Best 3 문장을 선택하도록 하였다. 표 7은 제안 모델에 빔서치를 적용하여 생성한 결과의 토큰 단위의 성능이다. 이 때, 우리는 입력 문장과 출력 문장이 동일하지 않은 결과에 대해서만 성능을 비교한다. 표 8은 빔서치를 적용한 모델의 정성 평가의 결과이다.

표 7 각 모델의 토큰 단위 성능

	기존 모델 결과 (입력 문장 ≠ 출력 문장)	빔서치 적용한 모델 결과
BLEU 1	0.496	0.497
BLEU 2	0.377	0.384
BLEU 3	0.288	0.298
BLEU 4	0.224	0.234
ROUGE_L	0.432	0.459
CIDEr	2.049	2.041

표 8 빔서치를 적용한 모델 결과 정성 평가

기준 분류	문장 수(비율)
입력과 동일한 문장	-
에러 문장	88(15.03%)
단어만 변경된 문장	315(53.75%)
문장 구조 변경 문장	183(31.22%)

빔서치를 적용한 모델의 결과가 기존 모델에 비해 BLEU 4와 ROUGE_L가 성능 향상을 보였다. 그리고 기존 모델은 평가 데이터 600문장 중 입력 문장과 동일한 문장을 제외하면 278문장 밖에 생성되지 않았지만, 빔서치를 적용한 모델은 586문장을 생성한 결과를 보였으며 이는 성공한 패러프레이즈 문장 또한 더 다양한 표현을 나타내는 것을 확인할 수 있었다.

표 9 기존 모델과 빔서치 적용 모델의 결과 비교 예시

분류	모델 분류	예시
에러 문장	입력 문장	롯데홈쇼핑 채널에서는 좀 더 상세한 내용을 16일에 방영(오전 0시 50분)한다.
	기존 모델	롯데홈쇼핑 채널에서는 좀 더 상세한 내용을 16일에 방영(오전 0시 50분)한다.
	빔서치 모델	롯데홈쇼핑 채널에서는 16일에 방영(오전 0시 50분)은 좀
단어만 변경된 문장	입력 문장	7일에 민주당은 문화체육관광방송통신위 회의장에서 철수할 예정이다.
	기존 모델	7일에 민주당은 문화체육관광방송통신위 회의장에서 철수할 예정이다.
	빔서치 모델	7일에 민주당은 문화체육관광방송통신위 회의장에서 철수할 계획이다.
문장 구조 변경 문장	입력 문장	대회에 참가하기 위해 안 씨는 2월부터 몸만들기를 시작했다.
	기존 모델	안 씨는 대회에 참가하기 위해 2월부터 몸만들기를 시작했다.
	빔서치 모델	안 씨는 대회에 참가하기 위해 2월부터 몸만들기를 시작했다.

표 9에 명시한 대로, 에러 문장의 경우 기존 모델은 입력 문장과 동일한 문장을 출력하였고 빔서치 모델은 문장을 생성하는 도중에 끝이나 해당 문장은 문장이 되지 않고 문법적 오류를 가진다. 단어만 변경된 문장의 경우 기존 모델은 입력 모델은 입력 문장과 동일한 문장을 출력하였고 빔서치 모델은 '예정'을 '계획'과 같이 단어를 변경하여 문장을 생성하였다. 문장 구조 변경 문장은 기존 모델과 빔서치 모델 두 결과 다 동일한 올바른 패러프레이즈 문장을 생성했다.

5. 결론

본 논문에서는 패러프레이징 생성을 위해 기존의 Sequence-to-Sequence 구조 위에 포인터 생성 네트워크(Pointer Generation Network)를 추가한 패러프레이징 모델을 제안하고, 패러프레이즈 데이터셋을 이용하여 모델의 효용성을 검증했다. 제안한 모델의 성능을 측정해보기 위해 사람이 직접 구축한 유사 문장 코퍼스를 이용하였으며, 토큰 단위의 BLEU-4 0.250, ROUGE_L 0.455, CIDEr 2.190의 성능을 보였다.

기존의 Sequence-to-Sequence와 attention을 이용할 경우, OOV로 표시된 비이상적인 단어들로 인해 부정확한

패러프레이즈 결과를 불러올 가능성이 있다. 반면에, 제안 모델의 접근법은 포인터 생성 네트워크는 입력 문장에서 단어를 가져올지 생성할지를 결정하고, 단어의 반복을 피하고자 이전 단어들의 분포를 고려하여 단어를 생성하고자 한다. 하지만 생성된 문장의 결과로 분석했을 때 입력 문장과 동일한 문장이 나오는 점과 다양한 문장을 표현하지 못하는 문제가 있다. 동일한 의미를 유지하기 위해 동일한 단어를 출력하는 경우와 문장의 구조를 변경하거나 문장의 구조는 같지만 다른 단어로 대체하도록 하는 특징 때문에 입력 문장의 단어가 출력에 많이 나타나는 경향을 보인다.

이를 해결하기 위해 우리는 빔서치를 적용하였고 한 문장에 대해 3개의 문장을 출력하도록 하였으며, 입력 문장과 비교해가며 최종 문장을 선택하도록 하였다. 이를 진행하였을 경우 토큰 단위의 BLEU-4 0.234, ROUGE_L 0.459, CIDEr 2.041의 성능을 보였다. 하지만 기존 모델이 600문장 중 278문장의 패러프레이즈 문장을 생성하였고, 빔서치를 적용한 모델이 586문장의 패러프레이즈 문장을 생성함을 확인할 수 있었다. 빔서치를 적용한 모델의 경우 생성된 문장도 다양하며 양도 증가함을 확인하였다.

입력 문장과 빔서치의 결과인 문장들을 비교하면서 최종 문장을 선택하도록 결정하였다. 이를 입력 문장과 동일한 문장을 생성할 경우 보상(reward)을 적게 주고, 입력 문장과 다른 문장을 생성할 경우 보상을 높게 주는 방식의 강화학습을 적용하여 다양한 문장의 표현을 가지는 패러프레이즈를 생성하는 방법을 향후에 진행할 예정이다. 그리고 네트워크의 구조를 변경하거나 외부 지식을 네트워크에 녹여 사용할 수 있는 방법을 향후로 연구할 예정이다.

Acknowledgement

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00440, 위험 상황 초기 인지를 위한 ICT 기반의 범죄 위험도 예측 및 대응 기술 개발)

참고문헌

[1] McKeown, Kathleen R. "Paraphrasing using given and new information in a question-answer system." Technical Reports (CIS) (1980): 723.

[2] Meteer, Marie, and Varda Shaked. "Strategies for effective paraphrasing." In Coling Budapest 1988 Volume 2: International Conference on Computational Linguistics. 1988.

[3] Madnani, Nitin, and Bonnie J. Dorr. "Generating phrasal and sentential paraphrases: A survey of data-driven methods." Computational Linguistics 36, no. 3 (2010): 341-387.

[4] Bannard, Colin, and Chris Callison-Burch. "Paraphrasing with bilingual parallel corpora." In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), pp. 597-604. 2005.

[5] Pang, Bo, Kevin Knight, and Daniel Marcu. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. CORNELL UNIV ITHACA NY DEPT OF COMPUTER SCIENCE, 2003.

[6] Prakash, Aaditya, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. "Neural paraphrase generation with stacked residual lstm networks." arXiv preprint arXiv:1610.03098 (2016).

[7] Mallinson, Jonathan, Rico Sennrich, and Mirella Lapata. "Paraphrasing revisited with neural machine translation." In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, pp. 881-893. 2017.

[8] Li, Zichao, Xin Jiang, Lifeng Shang, and Hang Li. "Paraphrase generation with deep reinforcement learning." arXiv preprint arXiv:1711.00279 (2017).

[9] Gupta, Ankush, Arvind Agarwal, Prawaan Singh, and Piyush Rai. "A deep generative framework for paraphrase generation." arXiv preprint arXiv:1709.05074 (2017).

[10] Roy, Aurko, and David Grangier. "Unsupervised paraphrasing without translation." arXiv preprint arXiv:1905.12752 (2019).

[11] B. Pang, K. Knight, and D. Marcu, "Syntax-based alignment of multiple translations: extracting paraphrases and generating new sentences," in NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Morristown, NJ, USA, pp.102-109, (2003).

[12] Y. Shinyama and S. Sekine, "Paraphrase acquisition for information extraction," in Proceedings of the second international workshop on Paraphrasing, Morristown, NJ, USA, pp.65-71, (2003).

[13] See, Abigail, Peter J. Liu, and Christopher D. Manning. "Get to the point: Summarization with pointer-generator networks." *arXiv preprint arXiv:1704.04368* (2017).

[14] https://corpus.korean.go.kr/main.do, 2020-10-06.