추가 데이터 및 도메인 적응을 위한 기계독해 질의 생성

이현구○, 장영진, 김진태*, 왕지현*, 신동훈*, 김학수
강원대학교 컴퓨터정보통신공학과
NLP Center Language AI Lab, 엔씨소프트*
nlphglee@kangwon.ac.kr, Dan_yon@kangwon.ac.kr, Kjt1505@ncsoft.com*, KorJhwang@ncsoft.com*,
dhshin@ncsoft.com*, nlpdrkim@kangwon.ac.kr

Question Generation of Machine Reading Comprehension for Data Augmentation and Domain Adaptation

Hyeon-gu Lee○, Youngjin Jang, Jintae Kim*, JiHyun Wang*, Donghoon Shin*, Harksoo Kim
Kangwon National University Computer and Communication Engineering
NLP Center Language AI Lab, NCSOFT*

요 약

기계독해 모델에 새로운 도메인을 적용하기 위해서는 도메인에 맞는 데이터가 필요하다. 그러나 추가 데이터 구축은 많은 비용이 발생한다. 사람이 직접 구축한 데이터 없이 적용하기 위해서는 자동 추가 데이터 확보, 도메인 적응의 문제를 해결해야한다. 추가 데이터 확보의 경우 번역, 질의 생성의 방법으로 연구가 진행되었다. 그러나 도메인 적응을 위해서는 새로운 정답 유형에 대한 질의가 필요하며 이를 위해서는 정답 후보 추출, 추출된 정답 후보로 질의를 생성해야한다. 본 논문에서는 이러한 문제를 해결하기 위해 듀얼 포인터 네트워크 기반 정답 후보 추출 모델로 정답 후보를 추출하고, 포인터 제너레이터 기반 질의 생성 모델로 새로운 데이터를 생성하는 방법을 제안한다. 실험 결과 추가 데이터 확보의 경우 KorQuAD, 경제, 금융 도메인의 데이터에서 모두 성능 향상을 보였으며, 도메인 적응 실험에서도 새로운 도메인의 문맥만을 이용해 데이터를 생성했을 때 기존 도메인과 다른 도메인에서 모두 기계독해 성능 향상을 보였다.

주제어: 기계독해, 정답 후보 추출, 질의 생성, 도메인 적응

1. 서론

기계독해(Machine Reading Comprehension)는 주어진 문맥에서 관련된 질문을 해결하는 질의응답 모델이다. 최근 기계독해 모델은 좋은 성능을 보여 다양한 도메인에 적용되고 있다. 그러나 새로운 도메인의 기계독해 모델은 도메인에 해당하는 데이터를 새로 구축해야하는 문제가 있다. 또한 충분한 수의 데이터가 확보되지 않으면 성능이 감소한다. 기존 연구에서는 추가 데이터 확보를 위해 번역[1]이나 학습 데이터에 나타나는 정답을 통해 질의 생성[2]을 하여 문장의 형태를 바꾸는 형식(paraphrasing)으로 데이터를 증가시켰다. 그러나 문장의 형태를 바꾸는 경우 새로운 정답 유형을 처리 할 수 없어 새로운 도메인에 적용이 어렵다. 또한 이러한 문제를 해결하기 위해 개체명을 정답으로 하여 질의 생성을 하는 경우 개체명 범주에 포함되지 않는 개체명이나 구(phrase)의 경우 정답으로 사용할 수 없는 문제가 있다. 본 논문은 기계독해 학습에 필요한 추가 데이터 확보(Data augmentation) 및 다른 도메인 적응(Domain adaptation)을 해결하기 위해 듀얼 포인터 네트워크 기반 정답 후보 탐지 및 포인터 제너레이터 기반 질의 생성 모델을 함께 사용한다. 본 논문에서 사용된 질의 생성 모델은 개체명 인식기가 찾을 수 없는 정답 후보 및 구도 정답으로 생성할 수 있으며 그로 인해 기존 데이터에 존재하지 않는 새로운 질문이 추가되어 기계독해 모델의 성능을 향상시켰다.

2. 관련 연구

기계독해는 SQuAD v1.1[3]을 통해 다양한 모델이 연구되었다. BiDAF[4], R-Net[5]은 인코딩 계층, 주의집중 계층, 출력 계층으로 기본 구조를 정립하였고 높은 성능을 보였다. 그 이후 많은 연구들은 성능을 향상시키기 위해 모델 위주의 연구가 진행되었다.[6] 또한 성능 향상을 위해 추가 데이터를 확보하는 방법이 연구되었으며 데이터를 번역[1], 질의 생성[2] 등 다양한 방법으로 추가 데이터를 확보하였다. [1]에서 번역 방법은 같은 정답을 가지는 질의를 다른 형태의 문장으로 표현만 바꾸었고 [2]는 질의 생성을 통해 새로운 질의는 추가하였으나 기존 학습 데이터의 정답을 기반으로 생성하였다. 두 연구 모두 기존의 정답만으로 데이터를 추가하기에 새로운 정답 유형은 처리할 수가 없다. 또한 개체명 인식기를 사용하여 질의 생성을 한다하여도 개체명 인식기의 범주에 포함되지 않는 개체명이나 구와 같은 형식의 정답은 생성할 수 없는 문제가 발생한다. 본 논문은 이러한 문제를 해결하기 위하여 듀얼 포인터 네트워크 기반 정답 후보 탐지 모델을 통해 정답 후보를 탐지하고 그것을 기반으로 질의를 생성하여 개체명 인식기에 영향을 받지 않는 추가 질의를 통해 기계독해 모델의 성능을 향상시키고 새로운 도메인에 적용시키는 방법을 제안한다.

3. 질의 생성을 통한 기계독해 성능 향상

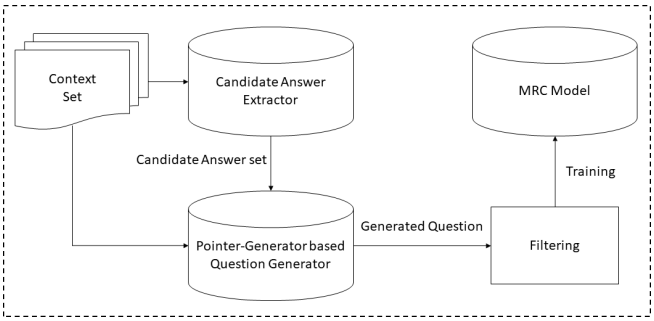

그림 1. 제안 모델의 흐름도

그림 1은 제안 모델의 흐름을 나타낸다. 전체 구조는 정답 후보 추출, 질의 생성, 생성 질의 필터링, 기계독해 모델 학습의 4단계로 구성된다. 정답 후보 추출은 문맥에 나타나는 모든 정답 후보를 추출하며 찾아진 정답 후보들은 질의 생성에 입력으로 사용된다. 질의 생성은 정답 후보와 문맥을 통해 정답이 나타날 수 있는 형식의 질문을 생성하는 단계이다. 생성 질의 필터링은 오류를 많이 포함하고 있는 생성 질의를 제거하여 생성된 질의의 품질을 향상시키는 단계이다. 마지막으로 기계독해 모델 학습은 생성된 추가 질의를 통해 학습하는 단계이다.

3.1. 정답 후보 추출

정답 후보 추출은 최대한 많은 후보를 찾아내기 위해 듀얼 포인터 네트워크를 사용한다. 기존에 정답 후보 추출은 BIO 태깅 방식을 사용하였으나 그림 2와 같은 안긴 정답을 처리하지 못하는 문제가 발생한다.

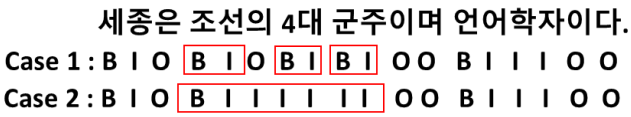

그림 2. BIO 태깅에서 발생하는 문제의 예

그림 2에서 “조선의 4대 군주”는 정답 후보가 될 수 있으나 “조선”, “4대”, “군주”도 각각 정답 후보가 될 수 있다. 그러나 BIO 태깅 시 case 2와 같이 “조선의 4대 군주”를 정답 후보로 찾아내면 나머지 3개의 예시는 찾아 낼 수 없다. 듀얼 포인터 네트워크를 사용한 방법은 그림 3과 같으며 예제와 같이 안긴 정답들을 모두 찾아 낼 수 있다.

그림 3. 안긴 정답을 처리하는 듀얼 포인터 네트워크

그림 3에서 실선은 정방향 포인터 네트워크이고 점선은 역방향 포인터 네트워크이다. 정방향 포인터 네트워크는 “조선의 4대 군주”가 포인팅되었고 역방향 포인터 네트워크에서는 “조선”, “4대”, “군주”가 각각 포인팅되어 “조선의 4대 군주” 안에 있는 안긴 정답들을 모두 찾아 낼 수 있다.

3.2. 질의 생성

질의 생성은 포인터 제너레이터[7]를 사용한다. 기존 생성에 많이 사용되는 Sequence-to-Sequence 모델[8]은 생성 확률만을 사용하며 이는 문맥에 나타나는 표현을 생성하는데 많은 오류를 포함하게 된다. 따라서 포인터 제너레이터를 사용하여 복사 확률을 추가해 문맥에 나타나는 표현을 더욱 잘 표현 할 수 있도록 해야 한다. 그림 4는 질의 생성에 포인터 제너레이터를 적용한 구조도이다.

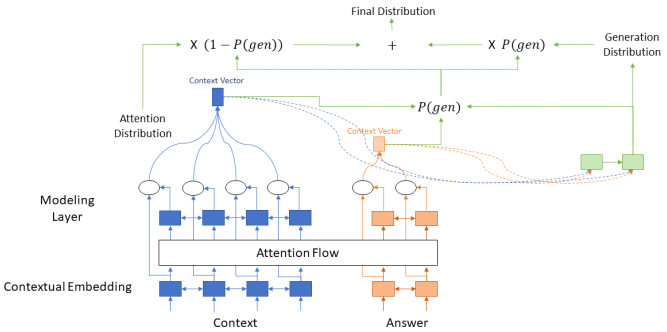

그림 4. 질의 생성을 위한 포인터 제너레이터

그림 4에서 입력은 문맥과 정답 후보가 되며 기존의 포인터 제너레이터의 인코더와 다르게 문맥과 정답 후보 간의 관계를 파악할 수 있도록 Attention Flow 계층[4]을 추가한다. 마지막으로 인코더 계층의 출력은 Attention Flow 계층 직전의 RNN(Contextual Embedding)과 Attention Flow 계층 이후의 RNN(Modeling Layer)의 출력을 식 1과 같이 합쳐서 사용한다.

$$gate_n = \sigma(FNN(AttentionFlow_n))$$
$$Encoder_n = (1-gate)*RNN_n^1 + gate*RNN_n^2 \quad (1)$$

식 1에서 $AttentionFlow_n$은 AttentionFlow 계층의 n번째 단어의 출력값이며 그 결과를 FNN과 sigmoid 활성화 함수를 통해 0~1의 값으로 표현한다. RNN_n^1은 AttentionFlow 계층에 입력되기 전의 RNN의 값이며 RNN_n^2은 AttentionFlow 계층 이후 사용된 RNN의 값이다.

3.3. 생성 질의 필터링

생성 질의 필터링은 생성된 문장에서 잘못된 어휘가 포함된 경우 기계독해 학습에 노이즈로 작용할 수 있어 불필요한 노이즈 데이터를 제거하기 위해 사용한다. 그림 5는 잘못 생성된 문장에서 각 단어의 생성 확률을 나타낸 것이다.

단어	Lig투자증권이	2016	년	7	월	다른	재무	적
생성 확률	0.060	0.252	0.988	0.953	0.998	0.44	0.66	0.989

단어	조부영	이	1990	년	촬영	하	ㄴ	것
생성 확률	0.226	0.732	0.397	0.994	0.076	0.486	0.784	0.036

그림 5. 잘못 생성된 문장의 단어 생성 확률의 예

그림 5의 첫 번째 예제는 “LIG투자증권이” 라는 어휘가 잘못 생성된 어휘로 기존 잘 생성된 나머지의 경우보다 확률값이 매우 낮은 것을 볼 수 있다. 두 번째 예제는 “촬영”, “것” 이 잘못 생성되었는데 역시 확률이 매우 낮은 것을 알 수 있다. 즉, 생성 확률이 낮다는 것은 생성 당시 애매하였고 이는 오류 확률을 향상시키므로 이러한 확률 값들을 식 2를 통해 필터링한다.

$$score = \frac{1}{N}\sum_{n=1}^{N} p(w_n) \quad (2)$$

식 2에서 $p(w_n)$은 n번째 단어의 생성 확률을 나타내며 모든 생성확률의 평균값을 필터링 점수로 사용한다. 본 논문에서는 실험을 통해 필터링 점수가 0.65 미만인 경우 노이즈가 많이 포함된 데이터로 판단하고 필터링을 하게 된다.

3.4. 기계독해 모델 학습

본 논문에서 기계독해 모델 GF-Net[9]를 사용하며 식 3과 같은 손실 함수(Loss function)를 통해 학습한다.

$$L(\theta) = -\frac{1}{N}\sum_{i}^{N}[\log(p^1_{y^1_i}) + \log(p^2_{y^2_i})] \quad (3)$$

식 1에서 $p^1_{y^1_i}$와 $p^2_{y^2_i}$는 예측된 확률 분포에서 정답의 시작 위치 y^1_i와 끝 위치 y^2_i의 확률 분포이다. 최적화는 AdaDelta[10]를 사용했으며 학습률 0.5 decay 0.999를 사용하였다.

4. 실험 및 평가

4.1. 실험 준비

본 논문에서는 실험을 위해 KorQuAD[11]와 자체 구축한 경제, 금융 데이터를 사용한다. KorQuAD는 학습 데이터 60,407개 개발 데이터 5,774개가 공개 되어 있으며 개발 데이터를 평가 데이터로 사용한다. 경제, 금융 데이터는 관련 도메인의 위키피디아, 나무위키를 통해 구축하였으며 학습 데이터 62,411개 평가 데이터 1,498개이다.

4.2. 실험 평가

본 논문에서 실험 지표는 완전 일치율(Exact Match)과 F1-점수(F1-score)를 사용한다. 완전 일치율은 모델의 응답과 정답이 완전히 일치하면 1 아니면 0으로 계산하고 F1-점수는 정답과 출력 간의 부분 일치 성능을 나타낸다. 실험은 추가 데이터로 인한 성능 향상 실험과 도메인 적응 실험 두 가지로 구성된다.

먼저 추가 데이터 실험은 질의 생성을 통해 만들어진 데이터를 추가했을 때 성능 변화를 알아보기 위한 실험이다. 표 1은 추가 데이터 실험의 결과이다.

표 1. 데이터 및 생성 데이터 추가에 따른 성능 변화

데이터	변화	Exact Match(%)	F1-score (%)
KorQuAD	-	73.26	87.71
	+ 생성 데이터	**73.31**	**87.89**
경제, 금융	-	73.10	87.81
	+ 생성 데이터	73.10	**88.14**

표 1에서 KorQuAD는 KorQuAD 학습 데이터를 사용해 학습한 내용이며 생성 데이터는 KorQuAD 학습 데이터에 나타나는 문맥만 이용하여 생성한 897개의 데이터이다. 경제, 금융 데이터도 마찬가지로 학습 데이터만을 학습하고 생성 데이터는 경제, 금융 학습 데이터에 나타나는 문맥만을 이용하여 생성된 1,254개의 데이터이다. 실험 결과 두 개의 데이터 셋 모두에서 성능이 향상되는 것을 볼 수 있었다. KorQuAD에서는 완전 일치율 0.05%p, F1-점수 0.18%p 상승되는 것을 보였고 경제, 금융 데이터에서는 F1-점수만 0.33%p 향상되는 것을 볼 수 있었다.

다음 실험으로 도메인 적응 실험에 앞서 도메인 변화에 따른 성능 변화는 표 2와 같다. 표 2는 KorQuAD 학습 데이터를 통해 학습한 모델에 도메인이 동일한 KorQuAD 평가 데이터와 도메인이 다른 경제, 금융 평가 데이터로 평가한 결과이다.

표 2. 도메인 변화에 따른 성능 변화

데이터	Exact Match(%)	F1-score (%)
KorQuAD 평가 데이터(dev set)	73.25	87.71
경제, 금융 평가 데이터	69.03	84.09

표 2에서 경제, 금융 평가 데이터의 성능을 보면 경제, 금융 학습 데이터를 통해 학습한 후 성능을 측정한 표 1의 결과(완전 일치율 73.10%, F1-점수 87.81%) 대비 완전 일치율 4.7%p, F1-점수 3.72%p가 하락한 것을 볼 수 있다. 즉, 도메인이 다를 경우 모델의 성능이 감소함을 알 수 있다.

도메인 적응을 실험하기 위해 경제, 금융 학습 데이터에 나타나는 문맥만을 이용하여 질의 생성을 한 1,254개의 생성 데이터를 KorQuAD 학습 데이터와 함께 학습한다. 표 3은 생성 데이터를 함께 학습했을 때의 성능 변화를 나타낸다.

표 3. 생성 데이터를 추가했을 때 성능 변화

데이터	Exact Match(%)	F1-score (%)
KorQuAD 평가 데이터(dev set)	73.59	87.94
경제, 금융 평가 데이터	69.56	84.74

표 3에서 경제, 금융 평가 데이터의 결과를 보면 생성 데이터를 추가하기 전보다 완전 일치율 0.53%p, F1-점수 0.65%p만큼 향상 된 것을 볼 수 있다. 기존 경제, 금융 학습 데이터만을 학습한 것만큼의 성능은 나오지 않았지만 성능 향상을 보여 도메인 적응에 어느 정도 효과가 있음을 알 수 있었다. 또한 KorQuAD 평가 데이터에서도 완전 일치율 0.34%p, F1-점수 0.23%p만큼 향상된 것을 볼 수 있는데 이는 다른 도메인의 데이터지만 추가 데이터가 생겨나 성능이 향상된 것을 알 수 있었다.

5. 결론 및 향후 연구

본 논문에서는 정답 후보 탐지와 질의 생성을 통해 추가 데이터 확보 및 도메인 적응이 기계독해에 효과가 있음을 보였다. 실험 결과 추가 데이터 확보는 두 가지 도메인에서 모두 효과가 있었다. 도메인 적응은 새로운 도메인에서 정답이 전혀 부착되어 있지 않은 문맥만으로 질의를 생성하여 학습했을 때, 생성 데이터를 추가하지 않았을 때보다 성능이 향상됨을 보였다. 향후 연구로 생성 질의 필터링이 너무 많은 생성 문장을 삭제하기에 이에 따른 개선안을 추가할 예정이고 다양한 도메인, 많은 문맥을 통해 생성 데이터를 많이 생성했을 때의 성능 변화를 보고자한다.

감사의 글

본 연구는 엔씨소프트 산학연구용역 과제의 지원을 받아 수행되었음

참고문헌

[1] A. W., Yu, D. Dohan, M. T. Luong, R. Zhao, K. Chen, M. Norouzi and Q. V. Le, Qanet: Combining local convolution with global self-attention for reading comprehension. arXiv preprint arXiv:1804.09541, 2018.

[2] X. Yuan, T. Wang, C. Gulcehre, A. Sordoni, P. Bachman, S. Subramanian, S. Zhang and A. Trischler, Machine comprehension by text-to-text neural question generation. arXiv preprint arXiv:1705.02012, 2017.

[3] P. Rajpurkar, J. Zhang, K. Lopyrev and P. Liang, SQuAD: 100,000+ Questions for Machine Comprehension of Text, arXiv preprint arXiv:1606.05250, 2016.

[4] M. Seo, A. Kembhavi, A. Farhadi and H. Hajishirzi, Bidirectional attention flow for machine comprehension, arXiv preprint arXiv:1611.01603, 2016.

[5] W. Wang, N. Yang, F. Wei, B. Chang and M. Zhou, Gated Self-Matching Networks for Reading Comprehension and Question Answering, Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol 1, pp. 189-198, 2017.

[6] J. Devlin, M. W. Chang, K. Lee and K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805, 2018.

[7] A. See, P. J. Liu and C. D. Manning, Get to the point: Summarization with pointer-generator networks. arXiv preprint arXiv:1704.04368, 2017.

[8] I. Sutskever, O. Vinyals and Q. V. Le, Sequence to sequence learning with neural networks. In Advances in neural information processing systems, pp. 3104-3112, 2014.

[9] 이현구, 김학수, "GF-Net 자질 선별을 통한 고성능 기계독해", 2018 한국컴퓨터종합학술회의, pp. 598-600, 2018.

[10] M. D. Zeiler, ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701, 2012

[11] 임승영, 김명지, 이주열. "KorQuAD: 기계독해를 위한 한국어 질의응답 데이터셋" 한국정보과학회 학술발표논문집, pp. 539-541, 2018.