기계 독해를 이용한 웹 기반 오픈 도메인 한국어 질의응답

최동현[a,b,O], 김응균[a], 신동렬[b]

카카오[a], 성균관대학교[b]

{heuristic.085, jason.eg}@kakaocorp.com, drshin@skku.edu

Web-Scale Open Domain Korean Question Answering with Machine Reading Comprehension

DongHyun Choi [a,b,O], EungGyun Kim[a], Dong Ryeol Shin[b]

Kakao Corp[a], Sungkyunkwan University[b]

요 약

본 논문에서는 기계 독해를 이용한 웹 기반 오픈 도메인 한국어 질의응답 시스템에 대하여 서술한다. 하나의 사용자 질의에 대하여, 본 논문에서 제안된 시스템은 기 존재하는 검색 엔진을 이용하여 실시간으로 최대 1,500 개의 문서를 기계 독해 방식으로 분석하고, 각 문서별로 얻어진 답을 종합함으로써 최종 답변을 도출한다. 실험 결과, 제안된 시스템은 평균적으로 2초 이내의 실행 시간을 보였으며, 사람과 비교하여 86%의 성능을 나타내었다. 본 논문에서 제안된 시스템의 데모는 http://nlp-api.kakao.com에서 확인 가능하다.

주제어: 기계 독해, 오픈 도메인, 한국어 질의 응답

1. 서론

기계 독해는 자연언어처리 분야의 중요 태스크 중 하나이다. 기계 독해는 일반적으로, 주어진 질문 - 문단 쌍에 대하여, 문단으로부터 질문에 대한 정답이 위치한 부분을 찾아내는 태스크로 정의된다. 최근 기계 독해 알고리즘의 급격한 발전은 SQuAD 데이터셋[1]에 상당 부분 기인한다. 최근의 BERT[2] 기반 기계 독해 알고리즘들은 심지어 사람보다도 더 높은 테스트셋 평가 성능을 보여주었다[1].

높은 성능을 가지는 기계 독해 알고리즘의 개발에 힘입어, 기계 독해 기반 오픈 도메인 질의응답 시스템을 만들고자 하는 연구가 진행되고 있다. 사용자 질의가 주어졌을 때, [3]에서 제안된 시스템은 TF-IDF 값을 이용하여 위키피디아에서 최대 5개의 관련 문서를 검색해낸 후, 얻어진 문서를 기계독해 알고리즘을 분석하여 답을 얻어낸다. [4]에서는 얻어진 문서의 각 문단들을 순위화하기 위한 폭포 모델을 제안하였다. 기계 독해 모델은 상위에 위치한 일부 문단에 대해서만 적용되었다. [5]에서는 실시간 오픈 도메인 질의응답 시스템을 만들기 위하여, 각 문서-질문 쌍을 실시간으로 분석하는 대신 위키피디아에 등장하는 모든 정답 후보를 전처리한 후, 향후 질문이 들어왔을 때 질문과 전처리된 정답 후보간의 유사도를 비교한 후 유사도가 가장 높은 정답 후보를 반환하는 시스템을 구성하였다.

현존하는 기계 독해 기반 오픈 도메인 질의응답 시스템은 위키피디아를 검색 도메인으로 제한한다. 이는 위키피디아의 문서들이 비교적 잘 정리되어 있으며 일반 웹 문서들에 비하여 신뢰도가 높기 때문이다. 그러나 검색 도메인을 위키피디아로 제한하는 것은 다음의 단점이 있다:

- 위키피디아 문서의 개수는 일반적인 웹 페이지의 문서보다 훨씬 적다. 영어 위키피디아의 문서 개수는 5,877,850개[2]이나, 색인된 영어 웹페이지의 개수는 최소 57억 개[3]에서 수백억 개[4]로 예측된다. 결과적으로, 위키피디아 기반 질의응답 시스템은 대부분의 롱테일 질의에 대한 답을 할 수 없다.
- 대다수의 언어에서 위키피디아 문서의 개수는 영어 위키피디아의 문서 개수에 비해 훨씬 적다. 예를 들어, 한국어 위키피디아는 459,864개의 문서만을 보유하고 있다.

본 논문에서는, 기계학습 기반 한국어 오픈 도메인 질의응답 시스템에 대하여 서술한다. 제안된 시스템은 일반적인 키워드 기반 검색 엔진에 의해 색인된 수십억 개의 한국어 웹 페이지를 이용한다. 신뢰도 이슈를 해결하기 위하여, 기계 독해 시스템을 이용하여 실시간으로 최대 1,500개의 웹 페이지로부터 후보 정답을 추출하며, 최종 정답은 얻어진 후보 정답들을 재차 분석하여 얻어진다.

[1] https://rajpurkar.github.io/SQuAD-explorer

[2] 2019년 6월 26일 기준

[3] https://www.worldwidewebsize.com

[4] https://www.google.com/search/howsearchworks

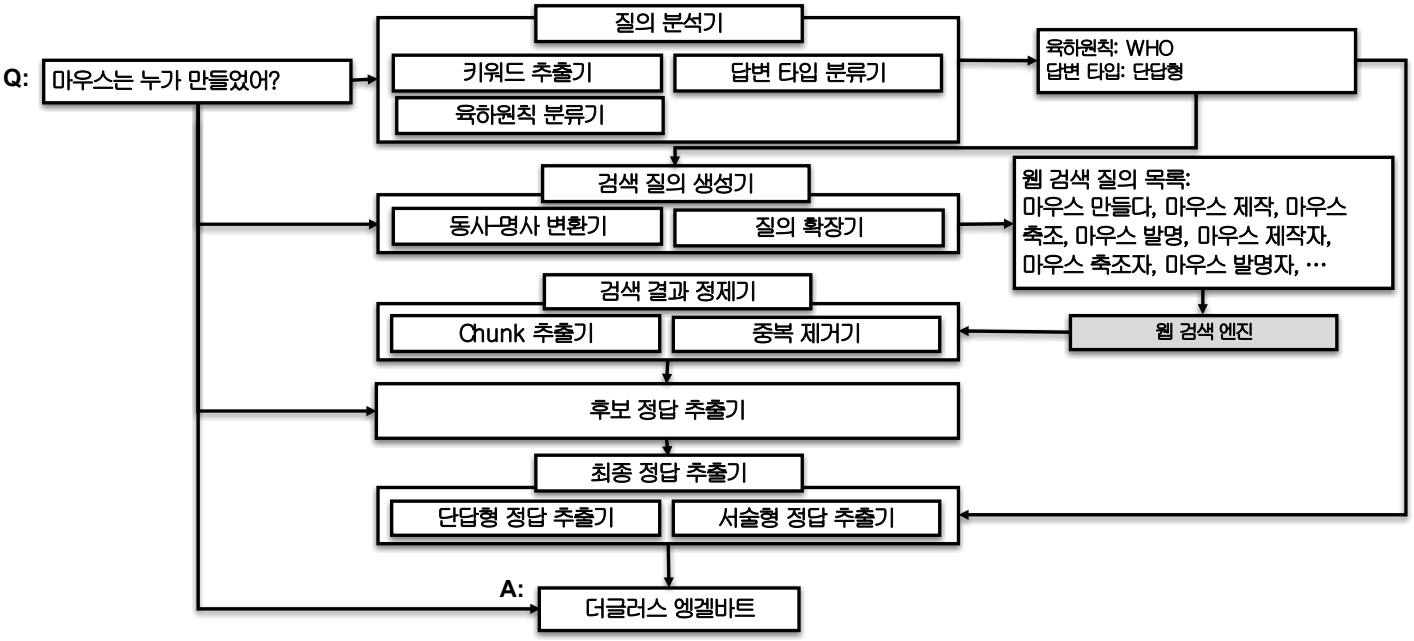

그림 1. 제안된 전체 시스템 구조

본 시스템에서 제안된 검색 도메인의 문서 개수(전체 한국어 웹 페이지) 는 위키피디아의 문서 개수보다 훨씬 많기 때문에, [5]와 같이 전체 검색 도메인을 전처리하는 방법은 적용하기 힘들다. 전체 실행 시간을 향상시키기 위하여, 높은 수준의 최적화를 통해 빠르고 효과적인 기계 독해 시스템이 개발되었다. 실험 결과에 따르면, 본 논문에서 제안된 시스템은 사용자 질의 1개당 평균적으로 1.6초의 실행 시간을 보였으며, 사람과 비교하여 86%의 정답률을 보였다.

2장에서는 제안된 시스템 구조에 대해서 서술한다. 3장에서는 실험 결과를 제시한다. 4장에서는 결론 및 향후 연구에 대하여 논한다.

2. 시스템 구조

그림 1은 제안된 시스템의 전체 구조를 나타낸다. 시스템은 5개의 부분 시스템 - 질의 분석기, 검색 질의 생성기, 검색 결과 정제기, 기계 독해기, 최종 정답 추출기 - 로 나뉘어질 수 있다. 표 1은 질문 2개의 처리 결과 예시를 보인다.

2.1 질의 분석기

질의 분석기는 먼저 주어진 질의 q에 대한 검색 키워드 목록 q_k를 추출한다. 이를 위해, q에 포함된 명사와 동사들 중 가장 높은 IDF값을 가진 상위 5개 단어를 추출한다. 이 때 사용되는 IDF값은 한국어 위키피디아를 이용하여 미리 계산된다.

또한, q의 육하원칙 카테고리 $WH(q)$를 얻어내기 위한 육하원칙 분류기가 개발되었다. 육하원칙 분류기는 주어진 질의 q를 WHAT, WHEN, WHERE, WHO, WHY, HOW 중 하나의 클래스로 분류한다. 얻어진 $WH(q)$값은 웹 검색 쿼리 확장 및 답변 타입 분류에 사용된다. 육하원칙 분류기를 구축하기 위하여, 총 112,993개의 한국어 질의가 6개 카테고리 중 하나로 어노테이션되었으며, 얻어진 말뭉치를 이용하여 [6]에서 제안된 합성곱 신경망 기반 문장 분류 시스템을 훈련하였다.

q에 대한 답변 타입 $AT(q)$는 질의 q에 대하여 기대되는 답변의 형태와 관련된 자질이다. 본 논문에서는 두 가지 가능한 답변 타입을 정의하였다. **서술형** 답변 타입은 정의나 추상화된 개념에 대한 설명과 같이, 어떤 서술이 필요한 답변을 요구하는 질의에 대한 답변 타입이다. 해당 카테고리에 속하는 질의의 예로는 "마우스가 뭐야", "컴퓨터가 뭐야" 등을 들 수 있다. 반면에, **단답형** 답변 타입은 기대되는 답변의 형태가 서술형이 아닌, 개체, 사람, 장소, 숫자 등인 질의에 대한 답변 타입이다. 해당 카테고리에 속하는 질의의 예로는 "마우스는 누가 만들었어", "경찰서 전화번호가 뭐야" 등이 있다.

본 논문에서는 $AT(q)$를 분류하기 위하여, $WH(q)$ 및 q_k에 기반한 간략한 규칙들을 정의하였다. 정의된 규칙은 표 1과 같다. 표 1에서, $c(q_k)$ 는 검색 엔진에 q_k를 질의하였을 때, 검색 결과의 개수, $c(P(q_k))$는 검색 엔진에 q_k를 구절 검색으로 질의하였을 때 검색 결과의 개수, r은 실험적으로 정의된 파라미터 값이다.

표 1. $AT(q)$ 분류를 위해 정의된 규칙.

규칙1. $WH(q)$ 가 WHERE 또는 WHEN → $AT(q) =$ **단답형**
규칙2. $WH(q)$ 가 WHY 또는 HOW → $AT(q) =$ **서술형**
규칙3. $WH(q)$ 가 WHAT 또는 WHO일 경우
규칙3-1. $c(P(q_k)) > r \cdot c(q_k)$→ $AT(q) =$ **서술형**
규칙3-2. $c(P(q_k)) \leq r \cdot c(q_k)$→ $AT(q) =$ **단답형**

표 2. 두 예시 질의에 대한 시스템 분석 결과.

사용자 질의	누가 마우스를 만들었어?	마우스가 뭐야?
검색 키워드 목록 q_k	마우스, 만들다	마우스
육하원칙 $WH(q)$	WHO	WHAT
답변 타입 $AT(q)$	단답형	서술형
검색 쿼리 리스트 S_q	마우스 제작, 마우스 축조, 마우스 발명, 마우스 창조, 마우스 창제, 마우스 제작자, 마우스 제작사, 마우스 발명자, 마우스 발명가, 마우스 만들다	마우스란$_p$, 마우스는$_p$, 마우스 뜻, 마우스 의미, 마우스 정의, 마우스 명칭
검색된 문서 개수	1,000	488
기계 독해 Chunk 수	409	183
최종 정답	더글러스 엥겔바트	컴퓨터와 사용자를 연결해주는 장치
전체 실행 시간	4.4초	1.6초
기계독해 실행 시간	2.1초	0.9초

2.2 검색 질의 생성기

검색 질의 생성기는 q_k를 이용하여 검색 엔진에 질의할 검색 쿼리 리스트 $S_q = \{(sq_1, p_1), (sq_2, p_2), \dots, (sq_n, p_n)\}$를 얻어낸다. 이 때, sq_k는 k번째 검색 쿼리이고, p_k는 구절 검색 여부이다.

한국어 동사는 다양한 활용형을 가지고 있기 때문에, 얻어진 동사를 바로 검색에 사용할 경우 충분한 검색 결과를 얻기 힘들다. 따라서, 검색 질의 생성기는 먼저 주어진 q_k의 동사들을 동일한 의미의 명사들로 변환하고자 시도한다. 357개의 엔트리를 가진 동사-명사 변환 테이블이 반자동으로 구축되었다.

그림 2. 동사-명사 변환 테이블 구축 과정

동사-명사 변환 테이블의 구축 과정은 다음과 같다. 먼저, 주어진 동사를 이용해 한영 사전의 표제어 검색을 수행한다. 해당 동사가 한영 사전의 표제어로 존재할 경우, 동사에 대응하는 영어 단어들이 순위화된 리스트 형태로 존재한다. 각 영어 단어들을 이용해 다시 영한 사전의 표제어 검색을 수행하였을 때, 대응하는 한국어 단어들에 (1) 원 한국어 동사가 포함되고, (2) 명사나, 명사로부터 파생된 동사가 존재할 경우, 해당 명사들을 추출하여 동사-명사 변환 테이블의 엔트리로 추가한다. 그림 2는 한국어 동사 "만들다"의 대응하는 명사 리스트인 ["창조", "제작", "생산"]을 얻어내는 방법에 대하여 도식화하였다. 얻어진 변환 테이블은 수작업으로 검수 및 수정되었다.

종종 q_k가 너무 일반적이라 적절한 문서를 검색해내지 못하는 경우가 존재한다. 예를 들어, 사용자 질의 "마우스가 뭐야" 에 대하여, 시스템이 검색 질의로서 추출된 검색 키워드 "마우스"만 사용한다면, 검색 결과는 마우스 회사, 마우스 판매 등 마우스의 정의와는 관련 없는 다수의 문서들을 포함하게 될 것이다. 이러한 문제를 해결하기 위하여, $WH(q)$ 및 $AT(q)$값에 기반해 특정 키워드들이 쿼리에 추가되었다. 표 3은 각 $WH(q)$ 및 $AT(q)$ 값에 따라 추가되는 검색 키워드를 나타낸다. 확장된 검색 쿼리의 예는 표 1에서 확인 가능하다.

표 3. 검색 키워드 확장 표. 작은 $_p$는 해당 키워드를 이용해 확장한 쿼리는 구절 검색을 수행함을 의미한다.

$WH(q)$	$AT(q)$	확장 키워드
WHAT	서술형	뜻, 의미, 정의, 명칭, 란$_p$/이란$_p$, 은$_p$/는$_p$
WHERE	단답형	장소, 위치, 주소
HOW	서술형	방법
WHY	서술형	원인, 이유

2.3 웹 검색

검색 질의 생성기에 의해 얻어진 검색 쿼리는 기존의 키워드 기반 한국어 웹 검색 엔진에 의하여 처리되어 관련성 있는 문서들을 검색해오게 된다. 각 검색 쿼리별로 최대 100개의 문서를 검색 결과로써 사용하고, 최대 검색 쿼리의 개수는 15개로 제한되었다. 따라서, 사용자 질의 1회에 대하여 시스템에서 최대 처리하는 문서의 개수는 $100 \times 15 = 1{,}500$개이다.

2.4 검색 결과 정제기

얻어진 검색 결과는 때때로 1만 개가 넘는 단어로 이루어져 있는데, 이는 기계 독해 시스템을 이용해 처리하기에는 너무 길다. 검색된 각각의 문서 D에 대하여, 검색 결과 정제기는 기계 독해의 입력으로 사용될 chunk의 목록 $c(D) = \{c_1, c_2, \ldots, c_c\}$를 도출한다. 각 chunk c_i는 D의 부분 문자열이며, D를 검색해내는 데 사용된 모든 키워드들을 포함하고 있다. 제안된 시스템에서 c_i의 최대 길이는 400 단어로 제한되었다.

각 검색 결과에서 추출된 chunk 목록들은 하나로 합쳐져 기계독해 모듈의 입력으로 사용될 chunk 목록 $C = \{c_1, c_2, \ldots, c_m\}$이 구축된다. 통합 과정에서, 다른 chunk와 unigram 이 70% 이상 겹쳐지는 chunk는 제거된다. 이는 때때로, 어떤 특정한 문단이 한 웹페이지에서 다른 여러 웹페이지로 복사되어 사용되는 경우가 종종 확인되기 때문이다. 중복된 chunk를 제거함으로써, 본 시스템은 복사된 문단에 의한 영향을 최소화하여 최종적으로 얻어진 정답의 신뢰도를 높일 수 있다.

표 1에서 질의에 대한 검색 결과의 개수와, 해당 검색 결과로부터 최종적으로 얻어진 chunk 개수의 예시를 확인할 수 있다.

2.5 후보 정답 추출기

기계 독해는 주어진 질문-문단 쌍에 대하여, 문단으로부터 질문의 답(answer span)을 추출해내는 작업이다. 본 시스템에서는 주어진 사용자 질의 q와 추출된 chunk 목록 $C = \{c_1, c_2, \ldots, c_m\}$ 에 대하여, 먼저 기계 독해 시스템의 입력으로 사용될 질문-문단 쌍 $L_{qp} = \{(q, c_1), (q, c_2), \ldots, (q, c_m)\}$을 도출한다. 각 쌍 (q, c_j)는 기계 독해 시스템의 입력으로 주어져 후보 정답 a_j를 얻어내는 데 사용된다. 후보 정답 추출기의 최종 출력은 각 질문-문단 쌍에서 추출된 후보 정답 리스트 $A = \{a_1, a_2, \ldots, a_m\}$이다.

그림 3은 본 논문에서 사용된 기계 독해 네트워크 구조를 보인다. 제안된 네트워크 구조는 크게 5개의 레이어로 구분된다.

임베딩 레이어. 시스템 실행 시간의 문제 때문에, ELMO[7]나 BERT[2]와 같은 기 훈련된 언어 모델 기반 임베딩 방법은 사용할 수 없었다. 대신, [6]에서 훈련된 한국어 GloVe 임베딩 벡터가 자모 단위 임베딩 벡터와 같이 사용되었다.

임베딩 인코더 레이어. 각 임베딩 벡터를 주변 문맥에 따라 업데이트하기 위하여, [8]에서 제안된 인코더 블록(encoder block)을 적용하였다. 그림 3의 오른쪽에 인코더 블록의 구조가 간략하게 도식화되었다. 하나의 인코더 블록은 N 개의 합성곱 레이어와 각각 하나씩의 셀프-어텐션(self-attention), 피드-포워드(feed-forward)레이어로 이루어져 있다. 제안된 시스템에서의 임베딩 인코더 레이어는 하나의 인코더 블록으로 이루어져 있으며, 인코더 블록 내의 합성곱 레이어의 수 N=4로 설정되었다. 임베딩 인코더 레이어의 결과값은 인코딩된 질의 $Q_e \in \mathbb{R}^{q_l \times d}$와 인코딩된 문단 $P_e \in \mathbb{R}^{p_l \times d}$로 표현 가능하다. 이 때, q_l과 p_l은 각각 질의와 문단의 길이를 나타낸다.

문단-질의 어텐션 레이어. 해당 레이어에서는 질의의 각

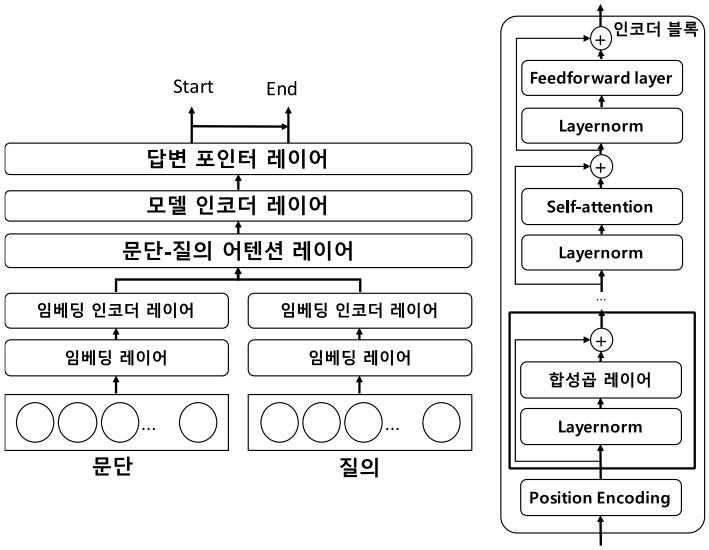

그림 3. 제안된 기계독해 신경망 구조

단어와 문단의 각 단어를 연결하고, 연결 정보를 이용하여 문단의 정보를 업데이트한다. 이를 위하여, [9]에서 제안된 iterative aligner가 적용되었다. Q_e와 P_e에 대하여, iterative aligner는 먼저 내적 어텐션 $E = P_e\{Q_e\}^{\mathrm{T}} \in \mathbb{R}^{p_l \times q_l}$을 계산한다. 이 때, 문단의 j번째 단어 w_j^P에 대한 질의 각 단어의 어텐션은 $softmax(E_{:j})$로 표현될 수 있다. 따라서, w_j^P 에 대한 어텐션 적용 질의 벡터는 $A_Q(w_j^P) = Q_e \cdot softmax(E_{:j})$의 식으로 계산될 수 있다. 각 단어에 대한 어텐션 적용 질의 벡터가 합쳐진 어텐션 적용 질의 매트릭스 $A_Q(P_e) \in \mathbb{R}^{p_l \times d}$는 [9]에서 제안된 의미 통합 유닛(semantic fusion unit, SFU) 를 이용하여 P_e와 통합된다. 의미 통합 유닛 $SFU(x, y)$는 다음 수식과 같이 정의된다:

$$\begin{aligned} \tilde{x} &= relu(W_r[x; y; x \circ y; x - y]) \\ g &= \sigma(W_g[x; y; x \circ y; x - y]) \\ SFU(x, y) &= g \circ \tilde{x} + (1 - g) \circ x \end{aligned} \quad (1)$$

위 수식에서, σ는 sigmoid 함수, $\circ$는 매트릭스 요소별(element-wise) 곱을 나타내며, W_r, W_g는 훈련되는 파라미터이다. 의미 통합된 문단 인코딩 P_e'은 $SFU(A_Q(P_e), P_e)$로 정의된다.

문단 내에서 멀리 떨어진 단어간의 의존을 고려하기 위하여, 위와 비슷한 방식으로 P_e'에 대한 자체 연결 정보가 계산되어 의미 통합 유닛을 통해 계산된다. 결과적으로, 문단-질의 어텐션 레이어의 결과값은 문단 인코딩 매트릭스 $P_f \in \mathbb{R}^{p_l \times d}$가 된다. 원 논문에서 이후 적용되는 LSTM 레이어는, 본 논문에서는 속도 향상을 위하여 생략되었다.

모델 인코더 레이어. P_f에 [8]에서 제안된 인코더 블록이 다시 적용되어 각 임베딩을 주변 문맥에 따라 업데이트한다. 기계 독해 시스템의 소요 시간을 줄이기 위하여, 모델 인코더 레이어는 하나의 인코더 블록으로 이루어졌으며, 각 인코더 블록은 N=5개의 합성곱 레이어를 가지도록 설계되었다. 이는 원 논문의 N=2개 합성곱 레이어를 가지는 7개 인코더 블록을 3회 반복한 네트워크 구조에 비하면 크게 단순화된 것이다. 모델 인코더 레이어의 출력은 문단 인코딩 매트릭스 $P_f' \in \mathbb{R}^{p_l \times d}$가 된다.

답변 포인터 레이어. [9]에서 제안된 방법이 문단에서 최종 답변 포인터를 추출하기 위하여 적용되었다. 주어진 P_f'에 대하여, 문단의 j번째 단어 w_j^P에 대한 시작/종료 확률 $p_s(j)$와 $p_e(j)$는 다음의 수식으로 계산된다:

$$\begin{aligned} s_q &= softmax(w_a \cdot Q_e^T) \cdot Q_e \\ p_s(j) &= \exp(w_{s2}^T \cdot \tanh(w_{s1}^T \cdot hc(P_f'[j], s_q))) \quad (2) \\ \tilde{s}_q &= SFU(s_q, P_s \cdot P_f') \\ p_e(j) &= \exp(w_{e2}^T \cdot \tanh(w_{e1}^T \cdot hc(P_f'[j], \tilde{s}_q))) \quad (3) \end{aligned}$$

위 수식에서, $hc(x,y) = [x; y; x \circ y; x - y]$로 정의되며, $P_f'[j]$는 P_f'의 j번째 열로 정의된다. 또한, $w_a, w_{s1}, w_{s2}, w_{e1}, w_{e2}$는 훈련되는 파라미터이다.

GPU를 사용하는 서버는 CPU만 사용하는 서버에 비하여 그 비용이 크게 올라가기 때문에, 실 서비스를 위해 전체 네트워크가 C++로 재구현되어 CPU 상에서 최적화된 속도로 수행되도록 하였다. Eigen[5] 라이브러리가 매트릭스곱을 위해 사용되었으며, 여러 layer들을 가능한 한 동시에 연산함으로써 추가적인 속도 향상을 노렸다. Compile flag는 –O3 –mfma –funroll-loops가 사용되었다.

2.6 최종 정답 추출기

최종 정답 추출기는 후보 정답 추출기로부터 얻어진 후보 정답 리스트에서 최종 정답 a_f를 도출해낸다. 최종 정답 추출 알고리즘은 답변 타입 $AT(q)$에 따라 달라지게 된다.

답변 타입이 단답형인 경우. 이 경우, 간단한 투표 알고리즘이 적용 가능하다. 중복 제거된 정답 후보 $U = \{u_1, u_2, \ldots, u_u\} \subseteq A$에 대하여, 각 u_i가 A에 등장한 횟수를 기반으로 u_i의 점수를 매긴다. 만약 u_i가 동일 문서의 서로 다른 chunk에서 여러 번 추출되었다면, 해당 추출로 인한 점수 증가 효과는 줄어들도록 설정한다. 또한, u_i가 다른 정답 후보 a_k의 부분 문자열이라면, u_i또한 a_k의 등장으로 인한 점수 중 일부를 점수로써 증가시켜 주도록 한다. U의 원소들 중 가장 큰 점수를 가진 u_f가 일단 선택되고, 좀 더 많은 정보를 사용자들에게 제공하기 위하여 u_f를 포함한 정답 후보들 중 등장 점수가 일정 설정값 이상이면서 가장 길이가 긴 정답 후보 a_f를 최종 정답으로써 사용자에게 반환한다.

답변 타입이 서술형인 경우. 이 경우, 후보 정답들의 표현 방식이 매우 다르기 때문에, 투표 알고리즘을 바로 적용하는 것은 불가능하다. 대신, 후보 답변들의 중심어에 투표 알고리즘을 적용한다. 한국어 명사구는 일반적으로 가장 마지막 단어가 중심어이기 때문에, 후보 답변의 가장 마지막 명사를 중심어로 결정하는 휴리스틱을 적용한다. 후보 정답들은 먼저 중심어를 이용해 필터링된 후, 다른 후보 정답들에 의해 많이 사용된 비중심어들을 가장 많이 포함한 후보 정답이 최종 정답 a_f로 도출된다.

3. 실험

먼저, 제안된 기계 독해 모델의 성능이 평가되었다. 다른 기계 독해 모델과 비교하기 위하여, SQuAD 1.1[1] 데이터셋이 제안된 기계 독해 모델을 영어에 대해 훈련시키기 위하여 사용되었다. 검증 데이터셋에서, 제안된 모델은 80.32%의 F1점수를 보였는데, 이는 BERT 이전 state-of-the-art 시스템의 성능과 비슷한 수치이다.

기계 독해 모델의 실행 시간 측정을 위하여, SQuAD 1.1 검증 데이터셋의 10,571 문단-질문 쌍이 이용되었다. 해당 데이터셋의 문단의 평균 길이는 143.36 단어이다. GPU와 CPU 간 실행 속도의 공평한 비교를 위하여, batch size는 1, CPU는 single-threaded로 실행되었다. 본 논문에서 제안된 C++ 기반 네트워크 추론기는 10,571 쌍을 처리하는 데 CPU[6] 상에서 총 641초(데이터 1개당 60.6 밀리초)가 소요된 반면에, tensorflow 구현 모델은 GPU[7]상에서 394초, CPU상에서 3,807 초가 소요되었다. C++ 기반 네트워크 추론기의 CPU 상에서의 성능 향상은 주로 네트워크에 특화된 모듈 최적화로부터 이루어졌다. 예를 들어, 합성곱 신경망 - bias - relu - 최대값 풀링의 4개 레이어는 C++ 기반 네트워크 추론기에서는 단 하나의 레이어로 통합되어 계산되었다. 한국어에 대한 기계 독해 네트워크를 구축하기 위하여, 총 201,468개의 한국어 질문-문단 쌍[8]이사용되었다.

전체 시스템의 성능을 평가하기 위하여, 육하원칙 카테고리 WHAT, WHEN, WHERE, WHO에 대하여 각 200개씩의 질문이 수집되었다. 각 질문에 대하여 두 명의 어노테이터가 답변을 구축한 후, 결과물이 수집되었다. 문제의 특성상, 하나의 질문이 여러 개의 답변을 가지는 것이 허용되었다. 표 4는 말뭉치에 포함된 질문-답변 예시를 보여 준다.

[5] http://eigen.tuxfamily.org

[6] Intel Xeon E5-2620

[7] Nvidia GeForce Titan X

[8] https://mindslab.ai 에서 얻어짐

표 4. 질문-답변 말뭉치 데이터 예시

질문	답변
토사구팽이 뭐야?	필요할 때 이용하다가 필요가 없어지면 버리는 것을 비유한 사자성어
소방서 전화번호가 뭐야?	119
아시아나 온라인 체크인은 언제부터 가능해?	출발 48시간 전부터
새로 밝혀진 이태원 살인 사건 진범이 누구야?	아서 패터슨, 아더 패터슨, 아더 존 패터슨

좀 더 실제적인 시스템 성능을 파악하기 위하여, 세번째 어노테이터가 직접 말뭉치에 포함된 질문에 대한 답을 찾는 작업을 수행하였다. 세 번째 어노테이터에게는 질문에 답하기 위하여, 말뭉치의 정답을 확인하는 것을 제외한 모든 정보 수집 활동 (웹 검색 포함)이 허용되었다. 표 5는 제안된 시스템의 실행 시간 및 평가 결과를 세 번째 어노테이터가 작성한 답변 평가와 함께 나타낸다. 표 5에서 **사람**은 세 번째 어노테이터가 작성한 답변의 F1 점수, **시스템**은 본 논문에서 제안된 시스템의 F1 점수, **T(Avg)**는 제안된 시스템의 질의당 평균 실행 시간(초), **T(Max)**는 제안된 시스템의 질의당 최대 실행 시간(초)을 나타낸다.

표 5. 시스템 성능 평가

	WHAT	WHEN	WHERE	WHO	ALL
사람	76.7	86.7	77.1	88.3	82.2
시스템	58.5	70.4	69.9	83.6	70.6
T(Avg)	1.8	1.5	1.9	1.0	1.6
T(Max)	3.9	3.1	4.1	4.7	4.7

표 5에서 관찰 가능하듯이, 제안된 시스템은 F1 70.6%의 성능을 보여 주었고, 이는 사람이 기록한 F1 점수 대비 86% 에 도달하는 것이다. WHAT 카테고리의 경우, 서술형 답변을 많이 포함하고 있어 다른 육하원칙 카테고리에 속하는 질의에 비해 낮은 성능을 보여 주었다. 또한, 시스템의 질의당 평균 실행 시간은 1.6초, 최대 실행 시간은 4.7초를 기록하여, 실시간으로 사용될 수 있는 실행 시간이 확인되었다. 본 논문에서 제안된 시스템의 데모는 http://nlp-api.kakao.com에서 확인 가능하다.

4. 결론

본 논문에서는 기계 독해를 이용한 웹 기반 오픈 도메인 질의응답 시스템에 대하여 서술하였다. 제안된 시스템은 사용자 질의를 평균적으로 1.6초 내에 처리 가능하였으며, 사람과 비교하여 86%의 성능을 보였으나, 아직 성능 향상의 여지가 많이 남아 있다. 현재 향후 연구로서 검색 쿼리 자동 확장, 개선된 질의 분류 알고리즘, 향상된 서술형 최종 답변 추출기 등이 연구되고 있다.

참고문헌

[1] P. Rajpurkar, R. Jia and P. Liang, Squad: 100,000+ questions for machine comprehension of text, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2383-2392, 2016.

[2] 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.

[3] D. Chen, A. Fisch, J. Weston and A. Bordes, Reading Wikipedia to answer open-domain questions, Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vol. 1, pp. 1870-1879, 2017.

[4] M. Yan, J. Xia, C. Wu, B. Bi, Z. Zhao, J. Zhang, L. Si, R. Wang, W. Wang and H. Chen, A deep cascade model for multi-document reading comprehension, arXiv preprint arXiv:1811.11374, 2018.

[5] M. Seo, J. Lee, T. Kwiakowski, A. P. Parikh, A. Farhadi and H. Hajishirzi, Real-time open-domain question answering with dense-sparse phrase index, arXiv preprint arXiv:1906.05807.

[6] 최동현, 박일남, 임재수, 백슬예, 이미옥, 신명철, 김응균, 신동렬, 한국어 대화 엔진에서의 문장 분류, 제 30회 한글 및 한국어 정보처리 학술발표 논문집, pp. 210-214, 2018.

[7] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark and L. Zettlemoyer, Deep contextualized word representations, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 1, pp. 2227-2237, 2018.

[8] 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, 6th International Conference on Learning Representations, 2018.

[9] M. Hu, Y. Peng, Z, Huang, X. Qiu, F. Wei and M. Zhou, Reinforced mnemonic reader for machine reading comprehension, Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 4099-4106, 2018.