Word2Vec의 IN-OUT Vector를 이용한 기계독해용 단락 검색 모델

김시형○, 박성식, 김학수
강원대학교, 컴퓨터정보통신공학과
sureear@kangwon.ac.kr, a163912@kangwon.ac.kr, nlpdrkim@kangwon.ac.kr

Paragraph Retrieval Model for Machine Reading Comprehension using IN-OUT Vector of Word2Vec

Sihyung Kim○, Seongsik Park, Harksoo Kim
Kangwon National University
Department of Computer and Communications Engineering

요 약

기계독해를 실용화하기 위해 단락을 검색하는 검색 모델은 최근 기계독해 모델이 우수한 성능을 보임에 따라 그 필요성이 더 부각되고 있다. 그러나 기존 검색 모델은 질의와 단락의 어휘 일치도나 유사도만을 계산하므로, 기계독해에 필요한 질의 어휘의 문맥에 해당하는 단락 검색을 하지 못하는 문제가 있다. 본 논문에서는 이러한 문제를 해결하기 위해 Word2vec의 입력 단어열의 벡터에 해당하는 IN Weight Matrix와 출력 단어열의 벡터에 해당하는 OUT Weight Matrix를 사용한 단락 검색 모델을 제안한다. 제안 방법은 기존 검색 모델에 비해 정확도를 측정하는 Precision@k에서 좋은 성능을 보였다.

주제어: 기계독해, 단락 검색 모델, Word2vec

1. 서론

기계독해는 문서의 한 단락과 질의를 입력하면 정답의 범위를 결과로 표시하는 분야이다. 최근 기계독해 분야는 SQuAD[1]와 같은 도전 과제에서 사람이 직접 답을 하는 것 보다 더 우수한 성능을 보이고 있다. 그러나 기계독해 모델에 모든 단락를 입력으로 사용할 수 없기 때문에 기계독해를 실용화하기 위해서는 질의와 관련있는 단락을 검색하는 것이 필수적이다. 이에 따라 최근 기계독해를 위한 검색 모델 연구에 대한 필요성이 대두되고 있다.

기존의 검색 모델은 주어진 질의에 대해 해당 질의의 어휘가 가장 많이 존재하는 단락을 검색하는 방법이 주로 사용되었다. 이를 발전시킨 TF-IDF를 사용한 방법은 각 어휘들의 가중치를 어휘 빈도수로 조정한 후 해당 가중치를 검색에 활용한다. 그러나 이러한 방법들은 질의와 같은 어휘가 존재하는 단락을 검색해 줄 수는 있지만 질의와 관련있는 단락을 검색하기는 어렵다. 예를 들어 “홍길동 선수가 결장한 이유” 라는 질의는 결장한 이유인 염좌나 골절이 아니라 홍길동이나 결장과 같은 어휘가 많이 매칭된다. 다시 말해 질의의 어휘에 해당하는 단락만 검색되는 문제가 있다. 이러한 문제를 해결하기 위해 본 논문에서는 기존의 검색 모델에 단어 임베딩을 결합하여 검색 순위를 재순위화 하여 성능을 향상시키는 방법을 제안한다.

2. 관련 연구

검색 모델의 성능을 향상시키기 위해서 주로 질의 정규화를 통해 성능을 향상시키거나 질의와 문서의 어휘를 다양하게 일치시키기 위해 질의 확장을 통한 성능을 향상시키려는 연구가 있었다[2-3]. 그러나 질의를 확장하는 방법은 관련있는 문서를 많이 검색 할 수 있지만, 정확한 문서를 높은 순위로 보여주기에는 많은 어려움이 따른다. 정확한 문서를 높은 순위로 보여주기 위해 다양한 연구에서는 재순위화(Reranking)를 이용한 후처리 방법을 사용한다[4-5]. 그 중 이진 분류 재순위화는 SVM, RankNet등을 사용하여 한 쌍의 문서중 어느 문서가 높은 순위인지 계산하는 이진 분류를 사용한다[6]. 다른 재순위화 방법으로는 적합성 피드백(Relevance Feedback)을 사용한 클러스터 기반 언어 모델[7-9]을 이용하여 문서를 검색한다. 이 방법은 문서와 문장을 동시에 색인 하고 문서 후보 점수와 문장 후보 점수를 각각 계산한 후 재순위화 할 때 문서 후보 점수와 문장 후보 점수를 각각 얼마나 반영할지를 하이퍼 파라미터 값으로 정한다. 그러나 각 후보 점수는 어휘의 의미까지 파악하여 해당 어휘와 유사한 주변 문맥에 해당하는 어휘를 반영하지 못하는 문제가 있다. 본 논문에서는 클러스터 기반 언어 모델에 추가로 질의의 각 단어의 단어 임베딩을 사용하여 해당 어휘의 주변 문맥에 해당하는 단락을 검색하여 후보 점수를 다시 계산하는 방법을 제안한다.

최근 단어 임베딩 방법은 NNLM[10], Word2vec[11], Glove[12], fastText[13], ELMo[14]등으로 다양하게 발

전해왔다. 그 중 Word2vec은 입력 단어 열을 벡터로 변환하는 IN Weight Matrix와 은닉 계층을 통과한 후 출력 단어열로 변환하는 OUT Weight Matrix를 가지는 특징이 있다. 일반적으로 Word2vec의 단어 임베딩을 사용할 때 IN Weight Matrix를 주로 사용한다. 본 논문에서는 IN Weight Matrix뿐만 아니라 OUT Weight Matrix를 같이 사용하여 문서를 재순위화 하는 방법을 제안한다.

3. 기계 독해를 위한 단락 검색 모델

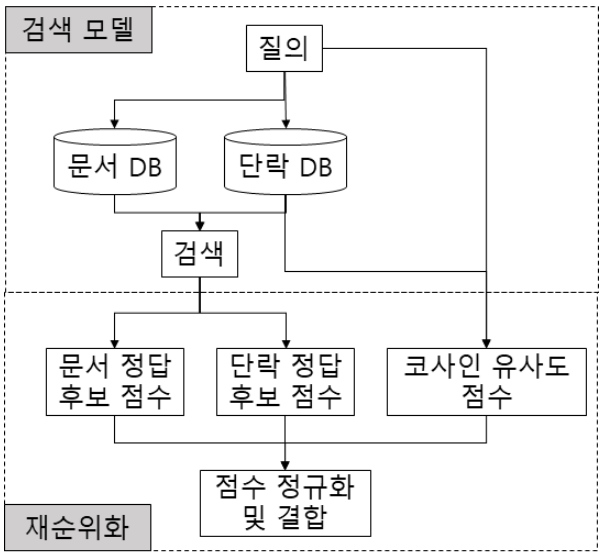

그림 1 단락 검색 모델 구조도

그림 1은 본 논문에서 제안하는 모델의 구조도를 보여준다. 그림 1과 같이 검색 모델에서는 질의에 대해 문서 검색과 단락 검색을 각각 수행한다. 그 이후 문서 후보 점수와 단락 후보 점수, 그리고 질의와 각 단락을 사용하여 계산한 코사인 유사도 점수를 결합하여 재순위화를 수행한다.

3.1 정답 후보 점수를 결합한 검색 모델

본 논문에서는 기계독해에서 입력 단위로 사용하는 단락을 검색하기 위해 문서와 단락을 각각 색인한다. 검색 모델은 BM25를 사용하여 문서와 단락을 검색하고, BM25의 파라미터는 k를 1.0, b를 0.18로 사용한다. 본 논문의 검색 모델은 문서 검색과 단락 검색을 수행 한 후 각 후보 점수를 다음과 같은 식을 사용하여 결합한다.

$$Score_{IR}(Q,P) = \alpha Score_{par}(Q,P) + (1-\alpha) Score_{doc}(Q,D) \quad (1)$$

식 (1)에서 Q는 질의이고, P는 단락이고, D는 문서이고, $Score_{par}$는 단락 정답 후보 점수, $Score_{doc}$는 문서 정답 후보 점수이고, α는 두 정답 후보 점수를 결합하기 위한 실험적 가중치이다. 식 (1)에서 후보 점수를 결합하기 위해 단락 정답 후보 점수와 문서 정답 후보 점수를 식 (2)를 사용하여 정규화한다.

$$z = \frac{x - m_i}{\sigma_i} \quad (2)$$

식 (2)에서 x는 정규화 대상이 되는 정답 후보 단락 점수, 정답 후보 문서 점수이고, m_i는 정답 후보 단락 점수의 평균, 정답 후보 문서 점수의 평균이고, σ_i는 검색된 정답 후보 단락 점수, 정답 후보 문서 점수의 표준편차이다.

3.2 단어 임베딩 점수를 결합한 검색 모델

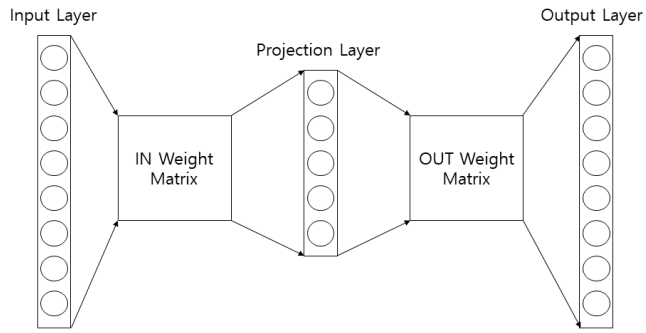

그림 2 Word2vec CBOW 모델 구조도

본 논문에서는 단어 임베딩을 검색 모델에 활용하기 위해 Word2vec을 사용한다. 그림 2는 Word2vec의 CBOW 모델의 구조도를 보여준다. 그림 2를 보면 CBOW 모델은 Input Layer는 단어를 One-hot encoding 한 벡터이고, 이를 IN Weight Matrix와 곱한 후, Projection Layer와 OUT Weight Matrix를 통과하여 Output Layer에서는 주변 문맥이 출력되도록 한다. Word2vec에서 학습한 단어 임베딩은 일반적으로 IN Weight Matrix에 해당하는 벡터를 단어 임베딩으로 사용한다. 그러나 본 논문에서는 Word2vec을 학습한 후 한 단어를 IN Weight Matrix에 해당하는 단어 임베딩과 OUT Weight Matrix에 해당하는 단어 임베딩으로 각각 나누어 사용한다. 식 (1)에 단어 임베딩을 사용한 점수를 결합하기 위해서 다음과 같은 점수 계산식을 사용한다.

$$Score_{emb}(Q,P) = \frac{1}{|Q|} \sum_{q_k} \frac{q_{i,k}^T \overline{P_o}}{||q_{i,k}|| ||\overline{P_o}||} \quad (3)$$

식 (3)에서 $Score_{emb}$는 단어 임베딩을 사용한 점수이고, $q_{i,k}$는 질의의 k번째 단어의 IN 벡터, $\overline{P_o}$는 단락의 각 단어의 OUT 벡터 평균을 구한 값이다. 식 (3)에서 질의나 단락의 단어 중에서 기존에 학습하지 않은 단어는 생략하여 계산한다. 식 (3)에서 계산한 후보 점수는 다른 후보 점수와 동일하게 식 (2)를 사용하여 정규화를 수행한다. 그 이후 단어 임베딩을 사용한 점수를 사용하기 위해 식 (1)을 다음과 같은 식으로 변경하여 사용한

표 1 각 가중치 검색 모델 실험 결과

	①			②			③			④			⑤			⑥			⑦		
가중치	α	β	γ	α	β	γ	α	β	γ	α	β	γ	α	β	γ	α	β	γ	α	β	γ
값	1.0	0.0	0.0	0.6	0.4	0.0	0.7	0.3	0.0	0.6	0.3	0.1	0.5	0.3	0.2	0.4	0.3	0.3	0.0	0.0	1
P@1	0.7309			0.7348			0.7361			0.7383			0.7382			0.7322			0.0561		
P@2	0.8121			0.8148			0.8163			0.8187			0.8199			0.8148			0.0957		
P@3	0.8451			0.8478			0.8487			0.8518			0.8536			0.8503			0.1283		
P@4	0.8639			0.8673			0.8674			0.8711			0.8723			0.8698			0.1571		
P@5	0.8775			0.8800			0.8803			0.8845			0.8856			0.8830			0.1830		
P@6	0.8868			0.8899			0.8907			0.8939			0.8949			0.8941			0.2086		
P@7	0.8949			0.8986			0.8988			0.9024			0.9041			0.9016			0.2304		
P@8	0.9008			0.9046			0.9050			0.9083			0.9101			0.9081			0.2514		
P@9	0.9060			0.9100			0.9105			0.9137			0.9151			0.9137			0.2707		
P@10	0.9107			0.9147			0.9146			0.9177			0.9192			0.9173			0.2904		

다.

$$Score_{IR}(Q,P) = \alpha Score_{par}(Q,P) + \beta Score_{doc}(Q,D) + \gamma Score_{emb}(Q,P) \quad (4)$$

식 (4)에서 $\alpha+\beta+\gamma=1$인 실험적 가중치이다. 식 (4)를 이용하여 단락의 어휘 일치도 뿐만 아니라 의미적으로 유사한 단락을 검색 할 수 있다.

4. 실험

4.1 실험 준비

본 논문에서는 단락 검색 모델을 실험하기 위해 위키피디아와 나무위키에서 수집한 문서 1,031개, 단락 7,803개를 사용하여 기계독해 질의 62,411개를 자체 구축하였다. 수집한 문서, 단락을 각각 색인 한 후, 구축한 질의를 검색하여 단락 후보 10개를 추출하였다. 실험은 α, β, γ를 변경하며 진행하였고, 성능은 Precision@k를 측정하였다. Precision@k는 단락 후보 top-k에서 정답 단락의 존재 여부를 측정하였다.

Word2vec은 Google에서 공개한 코드[1]를 사용하였고, 데이터는 위키피디아와 나무위키의 각주를 제외한 모든 텍스트를 사용하였다. Word2vec의 파라미터는 skipgram, size 100, window 5, sample 1e-4, negative 5, iter 100을 사용하였다. 또한 본 논문에서 사용한 문장들은 형태소 분석[15]을 통한 결과를 사용하였다.

4.2 실험 결과

표 1은 본 논문에서 제안한 검색 모델의 실험 결과이다. 표 1를 보면 α, β, γ를 각각 0.5, 0.3, 0.2로 설정한 ⑤번 모델이 P@1을 제외한 모든 지표에서 좋은 성능을 보였다. ①번 모델과 ②번 모델의 성능을 보면 단락만 색인하는 것이 아니라 문서도 색인하는 것이 검색 모델의 성능 향상에 도움이 되는 것을 알 수 있다. ②번 모델과 ④번 모델의 성능을 보면 질의에 해당하는 어휘의 주변 문맥에 해당하는 단락 어휘를 보는 것이 의미가 있음을 확인 할 수 있다. ⑦번 모델은 단어 임베딩만을 사용한 검색 모델로, 매우 낮은 성능을 보이고 있다. 이는 식 (3)에서 단락의 각 어휘 벡터의 평균을 구한 값을 질의의 각 어휘 벡터마다 계산하고 있어, 질의 어휘와 단락 어휘간의 일치도를 볼 수 없기 때문에 성능이 낮은 것으로 예상된다.

표 2 K-Nearest Neighbor 수행 결과

Weight Matrix	IN-IN		IN-OUT	
질의어	대학교/NNG	대학원/NNG	대학교/NNG	대학원/NNG
1	대학교/NNG	대학원/NNG	대학/NNG	대학/NNG
2	대학/NNG	학부/NNG	대학교/NNG	생/XSN
3	고등학교/NNG	대학/NNG	학교/NNG	대학원/NNG
4	의대/NNG	학과/NNG	**진학/NNG**	석박사/NNG
5	대학원/NNG	의대/NNG	**졸업/NNG**	**졸업/NNG**
6	학부/NNG	경영학과/NNG	하버드/NNP	대학교/NNG
7	캠퍼스/NNG	대학교/NNG	고등학교/NNG	**전공/NNG**
8	칼리지/NNG	석사/NNG	**입학/NNG**	**과학/NNG**
9	경영학과/NNG	법학/NNG	프린스턴/NNP	**교수/NNG**
10	아주대학교/NNP	행정학과/NNG	스탠퍼드/NNP	**진학/NNG**

표 2는 IN Weight Matrix의 단어 임베딩 공간에서 질의어의 단어 벡터를 추출한 후 IN Weight Matrix(IN-IN)와 OUT Weight Matrix(IN-OUT)에서 각각 K-NN(K-Nearest Neighbor)을 코사인 유사도를 사용하여 수행한 결과이다. 표 2를 보면 IN-IN의 "대학교/NNG"를 검색한 결과는 대학교와 비슷한 의미의 형태소들이 나타나는 반면 IN-OUT의 결과는 비슷한 의미뿐만 아니라 "진학/NNG"나 "졸업/NNG"와 같은 해당 단어의 주위 문맥에서 나타날 수 있는 형태소가 등장하는 것을 확인 할 수 있다. 또한 "대학원/NNG"를 검색한 결과에서도 IN-OUT에서의 결과는 "전공/NNG"나 "교수/NNG"와 같이 대학원의 의미와는 다르지만 대학원이라는 형태소의 주위 형태소가 근접해서 등장하는 것을 확인 할 수 있다.

5. 결론 및 향후 연구

본 논문에서는 Word2vec의 IN Weight Matrix와 OUT

1) https://code.google.com/archive/p/word2vec/

Weight Matrix를 사용하여 기존 검색 모델의 성능을 향상시킨 모델을 제안하였다. 또한 어휘의 IN 벡터를 IN Weight Matrix와 OUT Weight Matrix에 각각 K-NN을 수행하여 OUT Weight Matrix에서 주위 문맥에 해당하는 어휘가 근접하여 나타남을 보였다. 향후 연구로는 본 모델과 단어 임베딩만을 사용한 검색 모델을 향상 시킨 모델과의 비교 실험을 진행할 예정이다.

감사의 글

본 연구는 엔씨소프트 산학연구용역 과제의 지원을 받아 수행되었음.

참고문헌

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

[2] A. B. Abacha and P. Zweigenbaum, "Medical question answering: translating medical questions into sparql queries," Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium. ACM, pp. 41-50, 2012.

[3] A. R. Aronson and T. C. Rindflesch, "Query expansion using the UMLS Metathesaurus," Proceedings of the AMIA Annual Fall Symposium. American Medical Informatics Association, pp. 485, 1997.

[4] T. Joachims, "Optimizing search engines using clickthrough data," In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 133-142), ACM, 2002.

[5] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," In Proceedings of the 22nd international conference on Machine learning (pp. 89-96), ACM, 2005.

[6] B. Song, "Deep Neural Network for Learning to Rank Query-Text Pairs," arXiv preprint arXiv:1802.08988, 2018.

[7] 이현구, 김민경, 김학수, "의학문서 질의응답을 위한 정답 스닛핏 검색," 정보과학회논문지, 43(8), 927-932, 2016.

[8] J. J. Rocchio, "Relevance feedback in information retrieval," 1971.

[9] 김시형, 김진태, 김학수, 최맹식, "질의 확장 및 재순위화를 이용한 기계독해용 검색 모델," 한국정보과학회 2018 한국소프트웨어종합학술대회 논문집, 620-622, 2018.12.

[10] 이창기, 김준석, 김정희, "딥 러닝을 이용한 한국어 의존 구문 분석," 한글 및 한국어 정보처리 학술대회, 2014.

[11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality," In Advances in neural information processing systems, pp. 3111-3119, 2013.

[12] J. Pennington, R. Socher and C. Manning, "Glove: Global vectors for word representation," In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543, 2014.

[13] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," arXiv preprint arXiv:1712.09405, 2017.

[14] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," arXiv preprint arXiv:1802.05365, 2018.

[15] 최맹식, 김학수, "기계학습에 기반한 한국어 미등록 형태소 인식 및 품사 태깅," 정보처리학회논문지 B, 18(1), 45-50, 2011.