딥러닝 모형 기반 한국어 개체명 연결

손대능○, 이동주, 이용훈, 정유진, 강인호
네이버 검색연구
danny.sohn@navercorp.com, dongju88.lee@navercorp.com, yhlee.95@navercorp.com,
youjin.chung@navercorp.com, ihkang@navercorp.com

Named Entity Linking Based on Deep Learning Model

Dae-Neung Sohn○, Dongju Lee, Yong-Hun Lee, Youjin Chung, Inho Kang
Naver Search Dept.

요 약

개체명 연결이란 문장 내 어떤 단어를 특정 사물이나 사람, 장소, 개념 등으로 연결하는 작업이다. 과거에는 주로 연결 대상 단어 주변 문맥에서 자질 공학을 거쳐 입력을 만들고, 이를 이용해 SVM이나 Logistic Regression 혹은 유사도 계산, 그래프 기반 방법론 등으로 지도/비지도 학습하여 문제를 풀어왔다. 보통 개체명 연결 문제의 출력 부류(class)가 사물이나 사람 수만큼이나 매우 커서, 자질 희소성 문제를 겪을 수 있다. 본 논문에서는 이 문제에 구조적으로 더 적합하며 모형화 능력이 더 뛰어나다 여겨지는 딥러닝 기법을 적용하고자 한다. 다양한 딥러닝 모형을 이용한 실험 결과 LSTM과 Attention기법을 같이 사용했을 때 가장 좋은 품질을 보였다.

주제어: 딥러닝, 개체명 연결, 개체명 중의성 해소, 문장 분류

1. 서론

개체명 연결(Named Entity Linking)은 문장 내 출현한 단어의 중의성을 해소하여 단 하나의 개념, 사물, 인물, 장소 등으로 연결하는 작업이다. 단순히 단어가 인명, 장소, 시간, 기업명 등인지만 구분하는 개체명 인식(Named Entity Recognition)보다 더 정교한 모형이 필요한 문제이다[1]. 개체명 연결 기술은 단어 의미가 결과물 품질에 영향을 주는 검색 엔진, 대화 시스템 등에서 중요한 구성 요소가 될 수 있다[2][3][4]. [그림 1]은 검색 엔진의 문서 인덱싱 과정에 개체명 연결 기술이 적용된 예이다. 사용자가 "거미 (곤충)"을 입력했을 때와 "거미 (가수)"를 입력했을 때 달라지는 검색 결과를 볼 수 있다.

[그림 1] 검색 결과 예시

보통 중의성 해소 및 연결 대상이 되는 개체는 고유 명사 수만큼이나 많다. 필요에 따라 그 수를 100단위에서 1만 단위까지 쓰는 경우도 있다. 구별 대상 개체명이 많아질수록 전통적인 모형[3]이나 자질 공학만으로는 소요 비용과 시간 대비 좋은 품질을 얻기 힘들어 질 수 있으며, 결과물 품질이 매우 중요한 상업용 검색 엔진에 적용하기 어려울 가능성이 높다.

본 논문에서는 최근 자연어처리, 이미지분석, 음성인식 등에서 괄목할만한 성과를 내고 있는 딥러닝 모형을 한국어 문장 내 개체명 연결에 적용하고자 한다. 구체적으로 Multilayer Perceptron(MP), Convolutional Neural Network(CNN), Long-Short Term Memory(LSTM), LSTM+Attention 모형 별 품질과 처리 속도 등을 실험을 통해 비교 검증할 것이다.

2. 관련 연구

기존에 주류를 이루는 방법론들은 주로 개체명을 나타내는 자질을 어떻게 잘 추출하는지에 주목했다. 위키피디아[5] 같은 지식베이스에서 개체명을 잘 표현하는 구절이나 어휘, 링크 따위를 수작업으로 설계된 함수(Hand Crafted Feature Function), 자질 공학(Feature Engineering)을 이용해 자질을 추출한 후 입력과 자질간 유사도, SVM, Logistic Regression 모형 등을 학습하여 분류 작업에 이용하는 방식이다[2][3][4]. 한국어 개체명 연결 문제에서도 위키 문서 내 단어와 링크 정보로 자질 추출 함수를 구축해 중의성 해소를 시도한 사례가 존재한다[6].

이와는 상반되게 딥러닝 모형은 자질 추출 함수 설계나 자질 공학이 필요치 않다. 대신, 문장 내 단어의 의미를 분산된 에너지 벡터 형태로 표현할 수 있는 연구인

"단어 임베딩(word embedding)[7]" 방법론이 소개되었다. 이로써 희소성 문제 완화 및 단어를 딥러닝 모형의 입력으로 변환 가능케 하는 이론적 토대가 마련되었다. 단어간 벡터 유사도로 해당 단어와 의미, 문맥 측면에서 가장 가까운 단어들을 추출하는 것도 가능하다[8]. 문장과 문맥을 모형화하기 위한 목적으로 기존의 언어 모형을 대체할 수 있는 Recurrent Neural Network(RNN)[9] 구조도 존재한다. 이를 바탕으로 학습과정에서의 안정성과 문맥 보존 기능 향상을 위한 메모리 구조가 포함된 LSTM[10]이 고안됐으며, 기계번역[11], 음성인식[12] 등에서 좋은 성과를 보이고 있다. 감성 분류[13], 문서 분류[14] 분야에서는 Convolutional Neural Network 모형이 잘 동작하는 사례도 존재한다. 한국어 문장 분류 시 개체명 연결 같은 매우 큰 출력 부류(#CLASS > 1,000)를 가질 수 있는 문제에 딥러닝 기법을 적용한 연구는 현 시점에서는 없는 것으로 파악된다.

3. 개체명 연결을 위한 딥러닝 모형

3.1 입출력 정의

개체명 식별자로는 위키의 제목을 사용하였다[1]. 예를 들면, 일본 만화 원피스는 "원피스_(만화)", 영국의 축구 클럽 첼시는 "첼시_FC"로 정의한다. 연결 대상이 될 중의성이 높은 단어를 찾기 위해 위키의 동음이의어[15] 페이지를 사용해 후보 개체명을 추출했다. 검색 엔진 내 사용자 질의 입력 빈도에 따라 상위 1,428개의 개체명을 최종 분류 후보로 선정했다. 문장 내 연결 대상 경계 인식은 단어가 위키 동음이의어의 표면형과 동일한지 여부로 결정하며, 실제 서비스 적용을 고려하여 오류 전파 최소화와 부작용 대응 편의를 위해 별도의 개체명 경계 인식 로직은 두지 않았다. 딥러닝 모형의 입력은 길이 최대 30의 형태소 순서열이고 이보다 짧은 것은 0으로 패딩(padding)한다. 출력은 1,428차원의 벡터로써 하나의 차원은 하나의 특정 개체명을 뜻하며 이는 모든 입력에 동일하게 고정돼 적용된다[표 1]. 대상 개체명 수는 필요에 따라 유동적일 수 있으며, 오류 함수(LOSS)에 따라 1만~10만 단위까지도 처리가 가능하다[7].

[표 1] 입출력 예시.

입력 예: “9일 오후 방송 한 mbc 예능 프로그램 우리 결혼 했어요 에서는 곽시양 ***<연결 대상>***김소연***</연결 대상>*** 이 화보 촬영 하는 모습 이 그려졌다 이날 김소연은 곽시양”
출력 예: “output vector: {… **김소연_(1980년): 0.65**, 김소연_(2002년): 0.05, 원피스_(의류):0.01, MBC_(방송국):0.01, … }”

3.2 실험에 사용한 딥러닝 모형들

[표 2]의 수식을 딥러닝 모형 정의에 사용한다.

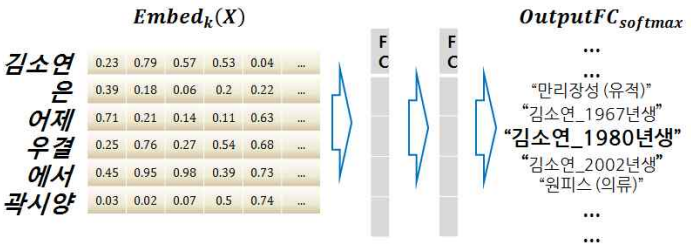
[표 2] 수식 정의

수식	설명
v	벡터 혹은 벡터로 표현되는 신경망 계층.
$S(v)$	벡터 혹은 계층의 차원 크기.
x_t	문장 내 t번째 출현한 한국어 형태소의 사전 번호.
X	입력: 최대 30차원의 x_t로 이루어진 어휘 벡터. $x_t \in X, S(X) = 30$.
y	출력: 개체명 별 점수 벡터. 본 연구에서는 1,428개의 개체명 사용. $S(y) = 1{,}428$.
$Embed_k(v)$	입력 형태소 별로 k차원의 벡터로 임베딩[7]. [4.장]의 학습데이터 만을 이용해 학습하였다.
$\#\mathrm{FC}_{\mathrm{relu}}(\mathrm{v})$	#개의 깊이로 중첩된 Fully Connected Hidden Layer + Rectifier Activation.
$OutputFC_{softmax}(v)$	Output Fully Connected Layer + Softmax Activation.
$Conv2d_{fsize}(v)$	필터 개수가 fsize인 2차원 convolution mapping[13].
$MaxPooling(v)$	convolution mapping 결과물을 합친 후 Max Pooling 작업 수행[13].
$\prod_{1}^{T} LSTMCell(Embed(x_t), h_{t-1})$	$t-1$단계의 은닉 계층 h와 t단계의 입력 임베딩 벡터 x를 인자로 사용하여 최대 입력 길이인 $T = 30$까지 전사(Propagation) 하는 LSTM 구조[10]. h_0=영벡터.
$Attention(h_1,...,h_{last})$	LSTM 전사 결과물 중 하나인 각 단계의 은닉 계층 상태 h_t를 가중 합하여 각각의 중요도를 반영하는 Attention기법[11].

3.2.1 Multilayer Perceptron (MP)

MP 구조는 신경망 깊이가 3이상인 feedforward fully connected neural network구조이다[16]. MP 구조는 구현이 용이하고 단순하다. 깊이에 따라 다르겠으나 일반적으로 학습 과정이 빠르고, 학습 과정에서 오류 전파 정보가 사라지는 현상인 Gradient Vanishing 문제가 RNN 구조보다 덜한 것으로 알려져 있다. 본 연구에 사용하는 모형 정의는 아래와 같다.

$$P(y|X) = OutputFC_{softmax}(\#FC_{relu}(Embed_k(X)))$$



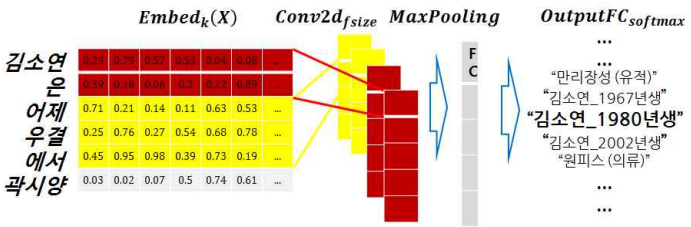
[그림 2] Multilayer Perceptron구조

[그림 2]는 형태소열 입력이 벡터로 임베딩된 후 2개의 fully connected layer를 거쳐 최종 출력으로 전사(Propagation)되는 것을 도식화한 것이다.

3.2.2 Convolutional Neural Network (CNN)

CNN 구조는 컴퓨터 비전 분야에 주로 사용되는 모형이다. 가로열 화소 정보를 하나의 벡터열로 표현하고, 이를 쌓아놓은 것으로 그림을 표현할 수 있다. 신경망 앞단의 계층에 그림 내 점과 선을 가장 잘 표현하는 정보를 거르는 일종의 여과망(filter)이 존재하며, 합성곱과 MaxPooling기법으로 오류 함수에 따라 문제를 푸는데 가장 적합한 정보 표현 방식을 자동으로 학습하게 된다[17]. 이를 그대로 문장 분류 문제에 적용한 사례가 있다[13]. 단어 임베딩 결과를 그림의 가로열로 간주하고, 이를 쌓으면 문장이 되어 전체 그림 정보를 처리하는 것과 같은 방식으로 문제에 맞는 모형을 학습할 수 있게 된다. 모형 정의는 아래와 같다.

$$P(y|X) = OutputFC_{softmax}(\#FC_{relu}(MaxPooling(Conv2d_{fzise}(Embed_k(X)))))$$



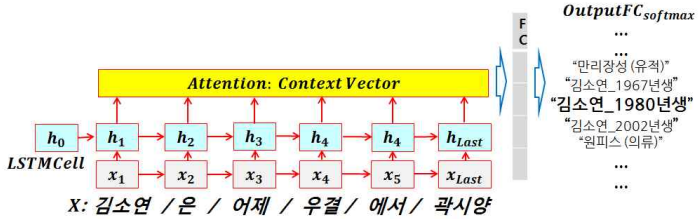
[그림 3] Convolution Neural Network 구조

[그림 3]은 임베딩된 벡터가 Conv2d의 여과망 크기에 따라 bi-gram 혹은 tri-gram단위로 합성곱되는 것을 보여준다. 이 여과망 집합에서 MaxPooling으로 문제에 가장 적합한 신경망 인자를 자동으로 학습하게 된다[13].

3.2.3 Long Term Short Memory (LSTM) + Attention

LSTM 모형[10]은 이전 단계의 은닉 계층 정보와 현 단계의 단어 임베딩 벡터 정보를 이용해 문장 전체를 순차적으로 전사하여 은닉 계층 상태에 표현되는 문맥 정보와 출력 벡터를 내놓는 RNN모형 중 하나이다. 기존 RNN과 다른 점은 내부에 메모리 구조가 있어 입출력과 은닉 계층, 메모리 갱신에 관여하는 4개의 게이트가 추가되어 있다는 것이다. 이로써 오류역전파 학습 시 깊이가 깊어질수록 자주 발생하는 Gradient Vanishing 문제를 해결할 수 있다[10]. 본 연구에서 사용하는 모형 정의는 아래와 같다.

$$P(y|X) = OutputFC_{softmax}(\#FC_{relu}(Attention(\prod_{1}^{T} LSTMCell(Embed(X), h_{t-1}))))$$



[그림 4] LSTM + Attention 구조

이 모형을 도식화하면 [그림 4]와 같다. 입력 단어가 순차적으로 임베딩되어 LSTMCell에서 각 단계별 은닉 상태에 문맥 정보가 생성되고, Attention을 거쳐 단계별 중요도가 고려되어 문맥 벡터로 표현된다.

3.3 학습 알고리즘

본 연구에서 사용하는 딥러닝 모형들은 출력 계층에 Softmax Activation을 한다. 이 경우 cross-entropy cost function을 정답 벡터 분포와 모형 결과 y간의 손실 함수(Loss Function)로 사용가능하며, 오류를 최소화하는 모형 인자 학습에는 역전파(back-propagation) 알고리즘을 이용한다[16]. 구체적인 손실 함수 정의는 다음과 같다.

$$LOSS(y_{정답}, y_c) = \sum_{k=1}^{\forall train} \sum_{c=1}^{S(y)} -y_{정답}^{k} \cdot \log(y_c^k)$$

학습단계에서는 과적합(over fitting)을 방지하기 위해 은닉 계층 당 50% 확률로 drop-out을 수행하였다. 대상 개체명이 1만 단위 이상으로 증가하여 가용 메모리, 계산량이 문제되는 경우, Hierarchical Softmax Activation[8]으로 대체하면 뚜렷한 품질 하락 없이 대응 가능하다.

4. 학습/평가데이터

학습데이터는 위키피디아 정보를 이용해 반자동으로 구축하였다. 각 개체명 별 위키 문서에서 프로필, 연고지 등이 담겨있는 정보창, 링크, 구절, 강조 어휘, 명사구 등을 모아 엔트로피 기반 정보 이득(Information Gain) 값으로 자질 선택을 수행했다. 개체명 별로 추출

[표 3] “원피스” 학습데이터 구축 용 질의 생성 예시.

개체명	자질	학습 문서 검색용 질의(& = Boolean And검색)
원피스_(만화)	오다 에이치로 / 몽키D루피 / 샹크스 / …	원피스 & 오다 에이치로 / 원피스 & 몽키D루피 / 원피스 샹크스 …
원피스	가을 드레스 / 겨울 치마 / 쉬폰 / …	원피스 & 가을 드레스/ 원피스 & 겨울 치마 / 원피스 & 쉬폰 …

된 자질을 개체명 어휘와 결합해 검색 질의로 만들어 검수를 거치고, 검색 엔진(네이버, naver.com)을 이용해 문서를 모았다. [표 3]은 “원피스” 관련 예시이다. 최종적으로 문서 내 해당 개체명 어휘 주변 좌우 최대 15 문맥까지만 모형의 입력으로 사용된다.

총 436,085건의 데이터가 모였으며, 각 개체명 별로 9:1의 비율로 각각 학습데이터와 평가데이터로 나누었다. 실험에는 개체명 당 평균 305건이 사용되었고, 표준편차는 169건이다. 데이터 전체의 형태소 사전 크기는 232,933이며 빈도3 이상인 92,472개만 사용하였다. 각각의 입력은 단 하나의 개체명 연결 대상만 갖는 걸로 가정하였으나, 딥러닝 모형의 출력 계층 특성상 하나의 입력당 다중 개체명 연결도 가능함을 알린다.

5. 평가

딥러닝 모형은 수동 설정이 필요한 은닉 계층 크기, 계층 깊이 등의 초차원 인자에 따라 품질이 달라지는 경우가 많다. 본 연구에서는 사전 실험을 통해 각 모형 별로 최선의 초차원 인자를 찾아 비교를 수행했음을 알린다. 연구 주제나 언어, 데이터의 특성에 따라 초차원 인자는 유동적으로 바뀔 수 있으며, 지면상 전체 사전 실험 결과는 생략한다. 평가 척도는 정확률과 재현율(Micro 방식)을 사용하였다.

[표 4] 모형별 품질.

모형	정확률	재현율	F1
Baseline	63.60%	63.60%	63.60%
Random Forest (Tree=5)	86.30%	82.51%	84.41%
Support Vector Machine (linear kernel)[2]	82.40%	79.42%	80.91%
MP	**94.73%**	**98.92%**	**96.82%**
CNN	**94.52%**	**98.85%**	**96.68%**
LSTM	**94.65%**	**99.22%**	**96.93%**
LSTM+Attention	**95.11%**	**99.23%**	**97.17%**

[표 4]는 각각의 모형별 실험 결과이다. Baseline은 연결 대상 어휘 별 가장 많이 나오는 개체명으로 연결했을 때의 결과이다. Random Forest(RF)와 Support Vector Machine(SVM) 모형의 결과도 전통적인 모형과의 비교를 위해 첨부했다. 위키 문서에서 명사구나 링크 키워드 등의 자질을 추출하고 자질 선택 과정을 거쳐 상위 10% 의 자질만을 RF와 SVM의 입력으로 사용했을 때의 결과이며, Milne[2]의 연구 결과와 유사한 품질을 보여주고 있음을 확인했다. 최종 실험 결과는 순서열 모형화 적합한 LSTM + Attention 이 F1 측정 기준 97.17%로 가장 좋은 품질을 보였으며, 그 뒤로 LSTM, CNN, MP 순의 결과값을 관찰하였다. 1,428개에 이르는 분류 대상으로 인해 SVM과 Random Forest의 모형화 능력으로는 딥러닝 모형 대비 좋은 품질을 보여주지 못했다. 참고로, 딥러닝 모형 품질의 정확률과 재현율은 출력 Softmax 계층의 1등 확률값 개체명이 임계값(0.5)을 넘은 경우만 모델이 분류했다 가정하고 얻은 결과물이다. 용도에 따라 임계값을 조정해 정확률/재현율 제어가 어느 정도 가능하다.

딥러닝 모형이 기존 모형 대비 잘 동작하는 이유로는 단어 자질 희소성 문제 완화와 의미 표현 방식을 들 수 있다. 아래는 "원피스"에 대해 기존 모형은 연결에 실패했으나 딥러닝 모형은 성공한 예제이다.

> "맨도롱 또똣 서이안 ***<연결대상>***원피스***</연결대상>*** 목지원"

예제에서 기존 모형은 "원피스_(의류)"와 관련이 높다고 추출된 자질이 없어 연결이 어려웠다. 하지만, 딥러닝 모형은 "배우" 문맥에서 자주 나타나는 "서이안", "목지원" 의 임베딩 벡터와 "여자", "배우" 문맥에서 자주 나오는 "원피스" 의 임베딩 벡터간 유사도가 가깝게 계산된다는 점 때문에 연결에 성공했다고 볼 수 있다. [표 5]는 "원피스+의류", "원피스+만화", "원피스+목지원"의 임베딩 벡터와 가장 근접한 임베딩 벡터 어휘들을 추

[표 5] “원피스+목지원”, “원피스+의류”, “원피스+만화” 임베딩 벡터 유사 어휘.

원피스+목지원	코사인 유사도	원피스+의류	코사인 유사도	원피스+만화	코사인 유사도
블라우스	0.601	***니트***	0.638	만화책	0.575
서이안	0.600	***블라우스***	0.621	칠무해	0.563
레니본	0.591	팬츠	0.595	징배	0.554
가디건	0.573	스커트	0.590	707화	0.553
니트	0.562	패션	0.586	극장판	0.537

[표 7] LSTM 모형 설정에 따른 품질과 초당 처리량 변화

LSTM 모형 설정	정확률	재현율	F1	초당 처리량
embedding차원=64, #FC=1, 은닉 계층 차원=64, Attention사용	95.09%	99.19%	97.14%	33,970
embedding차원=96, #FC=1, 은닉 계층 차원=96, Attention사용	95.10%	99.22%	97.16%	25,276
embedding차원=128, #FC=1, 은닉 계층 차원=128, Attention사용	**95.11%**	**99.23%**	**97.17%**	**21,847**
embedding차원=64, #FC=2, 은닉 계층 차원=64, Attention사용	94.70%	99.17%	96.93%	48,008
embedding차원=96, #FC=2, 은닉 계층 차원=96, Attention사용	94.28%	99.20%	96.74%	38,102
embedding차원=128, #FC=2, 은닉 계층 차원=128, Attention사용	94.32%	99.11%	96.72%	31,923
embedding차원=64, #FC=1, 은닉 계층 차원=64	93.04%	98.15%	95.60%	35,872
embedding차원=96, #FC=1, 은닉 계층 차원=96	93.60%	98.75%	96.18%	28,797
embedding차원=128, #FC=1, 은닉 계층 차원=128	94.03%	98.76%	96.40%	23,909
embedding차원=64, #FC=2, 은닉 계층 차원=64	91.19%	97.63%	94.41%	49,421
embedding차원=96, #FC=2, 은닉 계층 차원=96	92.13%	98.02%	95.08%	40,631
embedding차원=128, #FC=2, 은닉 계층 차원=128	94.65%	99.22%	96.93%	30,923

출한 것이며, "원피스+목지원" 은 "원피스+의류" 쪽에 겹치는 어휘가 많음을 알 수 있다.

딥러닝 모형간 품질 차이는 F1 측정 기준 0.5% 내외로 크지는 않았다. LSTM의 경우 Attention 기법을 사용한 것이 약간의 품질향상을 보였으며, 한국어 문장에서도 Attention 기법이 도움이 됨을 알 수 있었다.

[표 6] 딥러닝 모형 별 초당 처리량

모형	초당 처리량
MP	38,071
CNN	32,230
LSTM	30,923
LSTM+Attention	21,847

처리 속도도 비용 측면에서 모형 선택에 기준이 될 수 있다. 1회 Batch 당 GPU에 4096건을 입력해 측정한 초당 처리량 결과를 [표 6]에 첨부한다. 품질이 가장 좋은 LSTM+Attention 대비 MP가 1.7배의 정도 더 나은 처리량을 보여줬다. [표 7]은 초차원 인자 설정에 따른 LSTM 모형의 초당 처리량과 품질 변화를 관찰한 것이다. Attention 기법을 사용한 모형 설정이 전반적으로 좋은 품질을 보여주었으나 속도 하락이 존재하였다. 초당 처리량은 대상 도메인, 분류 대상, 초차원 인자 설정, OS와 GPU 등의 하드웨어의 최적화 알고리즘에 따라 달라질 수 있기에 본 수치는 참고 용도로만 활용되어야 한다.

5. 결론

본 논문은 한국어 입력 문장에 내 단어의 개체명 연결 문제를 여러 딥러닝 모형을 적용해 푸는 과정을 소개하고 각각의 품질을 관찰하였다. 전통적인 방법론보다 딥러닝 모형들의 품질이 우수하였으며, LSTM+Attention 방법이 가장 좋은 품질을 보였다.

연구 결과는 추후 검색 엔진 등에 적용할 수 있으며, 향후 연구로는 딥러닝 모형의 속도 최적화, 오류 사례에 대한 부작용 대응을 어떻게 할 것인가를 들 수 있다.

참고문헌

[1] Rosa Stern 외 2인, “A joint named entity recognition and entity linking system”, In Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data, pp.52-60, 2012.

[2] Olena Medelyan 외 2인, “Topic indexing with Wikipedia”, Proceedings of the Wikipedia and AI workshop at AAAI-08, 2008.

[3] David Milne and Ian H. Witten, "Learning to link with Wikipedia," Proceedings of the 17th ACM Conference on Information and Knowledge Management, 2008.

[4] Rada Mihalcea and Andras Csomai, “Wikify!: linking documents to encyclopedic knowledge”, Proceedings of the 16th ACM Conference on Information and Knowledge Management, 2007.

[5] https://ko.wikipedia.org

[6] 김용식, 최기선, "한국어 텍스트의 개체 URI 탐지: 품사 태깅 독립적 개체명 인식과 중의성 해소", 제26회 한국어 및 한국어 정보처리 학술대회 논문집, 2014.

[7] Tomas Mikolov 외 5인, “Distributed Representations of Words and Phrases and their Compositionality”, In Advances on Neural information Processing Systems, 2013.

[8] Tomas Mikolov 외 3인, “Efficient estimation of word representations in vector space”, arXiv preprint arXiv:1301.3781, 2013.

[9] Tomas Mikolov 외 4인, “recurrent neural network based language model”, In Proceedings of Interspeech, 2010.

[10] Sepp Hochreiter and Jürgen Schmidhuber, “Long Short-Term Memory”, Neural Computation. archive Volume 9 Issue 8, MIT Press, 1997.

[11] Dzmitry Bahdanau 외 2인, “Neural machine translation by jointly learning to align and translate”, arXiv preprint arXiv:1409.0473, 2014.

[12] Geoffrey Hinton 외 10인, “Deep neural networks

for acoustic modeling in speech recognition”, IEEE Signal Processing Magazine, 2012.
[13] Yoon Kim, “Convolutional neural networks for sentence classification”, Proc. Conference on Empirical Methods in Natural Language Processing, 2014.
[14] Siwei Lai 외 3인, “Recurrent convolutional neural networks for text classification”, In Proc. Conference of the Association for the Advancement of Artificial Intelligence, 2015.
[15] https://ko.wikipedia.org/wiki/분류:동음이의어_문서
[16] Rumelhart 외 3인, “Learning Internal Representations by Error Propagation”, Parallel distributed processing: Explorations in the microstructure of cognition, Volume 1: Foundations. MIT Press, 1986.
[17] Alex Krizhevsky 외 2인, “ImageNet classification with deep convolutional neural networks”, In Advances on Neural information Processing Systems, 2012.
[18] Srivastava, Nitish 외 4인, "Dropout: A simple way to prevent neural networks from overfitting", Journal of Machine Learning Research Vol 15, pp.1929-1958, 2014.