KIPS Tr. Software and Data Eng.
Vol.7, No.7 pp.259~266 pISSN: 2287-5905

https://doi.org/10.3745/KTSDE.2018.7.7.259

Document Classification using Recurrent Neural Network with Word Sense and Contexts

Jong-Min Joo† · Nam-Hun Kim†† · Hyung-Jeong Yang††† · Hyuck-Ro Park†††

ABSTRACT

In this paper, we propose a method to classify a document using a Recurrent Neural Network by extracting features considering word sense and contexts. Word2vec method is adopted to include the order and meaning of the words expressing the word in the document as a vector. Doc2vec is applied for considering the context to extract the feature of the document. RNN classifier, which includes the output of the previous node as the input of the next node, is used as the document classification method. RNN classifier presents good performance for document classification because it is suitable for sequence data among neural network classifiers. We applied GRU (Gated Recurrent Unit) model which solves the vanishing gradient problem of RNN. It also reduces computation speed. We used one Hangul document set and two English document sets for the experiments and GRU based document classifier improves performance by about 3.5% compared to CNN based document classifier.

Keywords : Document Classification, Word2vec, Doc2vec, Recurrent Neural Network, Gated Recurrent Unit

단어의 의미와 문맥을 고려한 순환신경망 기반의 문서 분류

주 종 민† · 김 남 훈†† · 양 형 정††† · 박 혁 로†††

요 약

본 논문에서는 단어의 순서와 문맥을 고려하는 특징을 추출하여 순환신경망(Recurrent Neural Network)으로 문서를 분류하는 방법을 제안한다. 단어의 의미를 고려한 word2vec 방법으로 문서내의 단어를 벡터로 표현하고, 문맥을 고려하기 위해 doc2vec으로 입력하여 문서의 특징을 추출한다. 문서분류 방법으로 이전 노드의 출력을 다음 노드의 입력으로 포함하는 RNN 분류기를 사용한다. RNN 분류기는 신경망 분류기 중에서도 시퀀스 데이터에 적합하기 때문에 문서 분류에 좋은 성능을 보인다. RNN에서도 그라디언트가 소실되는 문제를 해결해주고 계산속도가 빠른 GRU(Gated Recurrent Unit) 모델을 사용한다. 실험 데이터로 한글 문서 집합 1개와 영어 문서 집합 2개를 사용하였고 실험 결과 GRU 기반 문서 분류기가 CNN 기반 문서 분류기 대비 약 3.5%의 성능 향상을 보였다.

키워드 : 문서 분류, Word2vec, Doc2vec 순환신경망, GRU

1. 서 론

최근에 인터넷을 통해 정형 혹은 비정형의 뉴스, 블로그, SNS(Social Network Service) 등의 정보가 많이 발생하고 있다. 불필요한 정보는 걸러내고 필요한 것만 검색하기 위해서는 데이터를 미리 분류하는 문서 분류과정이 필요하다[1]. 이를 위해 단어의 빈도 및 의미를 고려한 문서 분류 방법에 대한 많은 연구가 수행되었다.

문서를 분류하기 위해서는 먼저 문서를 컴퓨터가 이해할 수 있도록 표현해야 한다. 문서를 표현하는데 가장 많이 사용되는 Tf-iDf(Term Frequency-Inverse Document Frequency)는 한 문서 내에 특정 단어가 등장하는 빈도수와 해당 단어가 전체 문서 집합에서 등장하는 빈도수를 통해 문서의 특징을 표현한다[2]. 단어의 빈도수를 표현하는 방법에서 불린(Boolean) 빈도, 로그(Log) 스케일 빈도 등 여러 가지 방법이 있지만 최대 스케일 값을 1로 고정시킬 수 있는 증가 빈도 방법이 많이 사용된다. 문서의 빈도수는 전체 문서 중 발생하는 빈도수의 역수를 취하기 때문에 값이 무한히 커질 우려가 있다. 따라서 이를 방지하기 위해 로그 값을 취한다. Tf-iDf 모델은 문서를 벡터로 표현해주는 쉽고 빠른 방법으로 현재까지도 많이 사용되고 있고 성능 또한 우수한 편이다. 하지만 단순히 문서를 빈도수에 의해서만 벡터로 표현하기 때문에 단어의 중의적 의미

※ 이 논문은 2017년 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2017M3C4A7068187).
※ ITRC_"본 연구는 미래창조과학부 및 정보통신기술진흥센터의 대학ICT 연구센터 육성지원사업의 연구결과로 수행되었음"(IITP-2018-2016-0-00314).
† 비 회 원 : 전남대학교 전자컴퓨터공학부 석사과정
†† 비 회 원 : 전남대학교 전자컴퓨터공학부 석사
††† 종신회원 : 전남대학교 전자컴퓨터공학부 교수
Manuscript Received : March 8, 2018
First Revision : May 8, 2018
Accepted : May 12, 2018
* Corresponding Author : Hyung-Jeong Yang(hjyang@jnu.ac.kr)

를 표현하지 못하고 단어사이의 문맥적인 관계를 고려하지 못하는 문제가 발생한다.

지금까지의 문서 분류 방법은 베이시안 분류, 최근접 이웃 기법(k-Nearest Neighbor), 서포터 벡터 머신(Support Vector Machine), 신경망(Neural Network) 등의 다양한 기계학습 기법과 통계적 추론을 사용해왔다[3]. 합성곱 신경망(Convolutional Neural Networks)은 일반적으로 2차원 영상에 최적화된 신경망이지만 지역적 정보를 보존하여 문맥을 이해할 수 있기 때문에 자연어 처리 분야에서 높은 성능을 내고 있다. 그러나 CNN은 인접해 있는 몇 개의 단어들만을 고려하기 때문에 성능에 한계가 있다. 순환신경망(Recurrent Neural Network)[8]은 문맥 정보를 보존하면서 셀(cell)이라는 개념을 도입하여 먼 거리에 있는 단어들도 기억하기 때문에 이러한 문제를 해결할 수 있다.

본 논문에서는 효율적인 문서 색인을 위해 word2vec과 doc2vec을 사용한다. word2vec을 사용해 일반적인 Tf-iDf 방식보다 단어를 의미 있게 표현하고 doc2vec을 통해 문맥을 고려하여 문서를 벡터로 표현한다. 추출된 특징 벡터를 분류하기 위해 순환신경망(RNN)을 사용한다. RNN은 일반적인 신경망 학습에서 기억능력을 추가하여 그 전의 입력들까지 기억하는 방식으로 자연어 처리 분야에서 널리 사용되고 있다. 실험결과 RNN에서 그라디언트가 소실되는 문제를 해결해주고 계산 량을 줄이는 GRU(Gated Recurrent Units) 방식이 CNN을 사용한 모델보다 약 3.5%의 우수한 결과를 보였다.

2. 관련 연구

문서를 분류하기 위해서는 컴퓨터가 이해하는 언어로 변경하는 색인 작업을 해야 한다. 색인은 문서 내 단어의 빈도수와 위치를 기반으로 벡터화하를 수행하는 것이다. 벡터공간모델인 Tf-iDf는 문서에 등장하는 단어들의 중요도를 나타내는 특성 값을 사용하여 문서를 벡터형태로 바꿔준다[4]. 출현률을 기반으로 문서의 특징벡터를 추출하고 특징 벡터들을 문서 분류에 사용하는 방법이 가진 문제점은 문서에 포함된 단어들만이 문서의 특징으로 영향을 미친다는 것이다. 이러한 Tf-iDf 벡터공간모델은 문맥과 의미를 고려하지 않기 때문에 성능에 한계가 있다.

[5]의 연구에서는 감정 자질의 효과적인 추출 방법과 추출된 감정의 가중치를 강화한 한국어 문서 감정 분류 방법을 제시하였다. 먼저 감정 자질 추출을 위해 영어 단어 유의어 정보를 이용하여 자질들을 확장하였고, 영한사전을 통해 확장된 자질들을 번역하여 감정 자질들 추출한다. 추출된 감정 자질로 카이 제곱 통계량을 통해 감정 강도를 구한다. 마지막으로 긍정 문서에서는 긍정 감정 자질만 강화하고 부정 문서에서는 부정 감정 자질만 강화하여 서포터 벡터 머신으로 분류한다. 실험 결과 일반적인 Tf-iDf 모델보다 성능 향상을 보였다. 그러나 영한사전을 통한 번역이 확실치 않고 문맥을 고려하지 않아 단어의 감정값이 확실치 않다는 단점이 있다.

[6]의 연구에서는 이러한 단점을 보완하기 위해 문서 내의 단어 간 유사도를 사용하여 출현하지 않은 단어의 특징 값을 간접적으로 평가하고 같은 문서 내에서 출현한 유사한 단어들은 가중치를 주는 word2vec 알고리즘을 제안하였다. 이 방법은 단어가 가지는 의미를 고려할 수 있는 장점이 있다. word2vec 알고리즘은 Tf-iDf 알고리즘과 다르게 단어간의 의미적 유사도를 벡터로 표현할 수 있다. 그러나 문서 내에서 단어의 순서로 인해 생기는 문맥을 고려할 수 없다는 단점이 있다.

자동으로 문서를 분류하기 위해서는 어느 범주로 문서를 분류할 것인지를 결정하기 위해서 문서 분류 규칙을 정한다. 문서 분류를 위해 학습에 사용되는 알고리즘은 규칙 기반 방법, 확률 기반 방법, 결정트리를 이용하는 방법, SVM을 이용하는 방법 등 다양한 방법이 있다[3]. 이러한 방법들은 고전적인 방법으로 대부분 확률에 의한 결정 방법을 보여주고 최근에 제시되는 딥러닝 기반의 분류기에 비해서 낮은 성능을 보이고 있다.

[22]의 연구에서는 word2vec과 doc2vec을 활용하여 문서를 벡터화하고 합성곱 신경망을 분류기로 사용하였다. 합성곱 신경망은 기존의 신경망에 필터기능을 추가하여 2차원 영상에 최적화된 신경망 알고리즘이다[7]. 합성곱 신경망은 자연어 처리 분야에서도 높은 성능을 기록하지만 평균 단어 수가 16개로 이루어진 짧은 문장에 한해서 좋은 성능을 보인다는 한계가 있다[8]. 또한 구문 전체와 의미론적 관계가 복잡한 전체 문장을 다루기 어렵다는 단점을 가지고 있다.

[8], [20]에서는 합성곱 신경망 대신 LSTM (Long-Short Term Memory) 알고리즘과 word2vec 모델을 적용하여 문서 분류를 수행하였다. LSTM을 사용하여 긴 시퀀스의 입력에도 과거의 입력을 이용하여 효과적인 문서 분류가 가능하게 하였다. 그러나 문장을 하나의 벡터로 표현하지 못하였고 GRU에 비해 학습 시간이 오래 걸린다는 단점이 있다.

3. 본 론

본 논문에서는 문서 내의 단어의 의미와 순서를 문서 벡터로 표현하기 위해 단어의 의미와 문맥을 고려하는 word2vec과 doc2vec을 이용하여 문서를 표현하고, 순환 신경망 분류기를 이용하여 문서를 분류하는 방법을 제안한다. 전체적인 시스템 구조도는 Fig. 1과 같다.

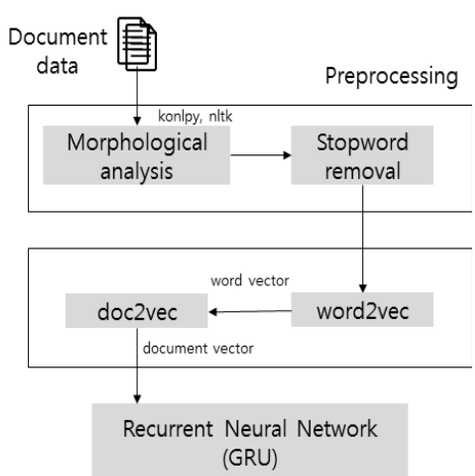

Fig. 1. Document Classification Process

한 문서는 동사, 명사, 조사, 형용사, 부사 등 다양한 품사로 구성되어있지만 모든 품사를 문서의 특징으로 사용하면 불필요한 정보가 다수 포함된다. 본 논문에서는 명사를 문서의 주요 특징으로 보고 형태소 분석을 통해 명사만을 추출한다. 추출된 명사 중에서도 다수의 문서에서 출현하는 단어는 문서간의 분별력을 낮추기 때문에 제거한다. 추출된 단어의 의미와 문맥을 고려하기 위해 word2vec을 사용하여 단어 벡터를 생성하고 doc2vec을 사용하여 문서를 벡터화한다. 마지막 단계로 문서 벡터를 RNN의 한 방법인 GRU 분류기를 통해 문서 분류한다.

3.1 단어의 의미와 맥락을 고려한 이용한 문서 표현

기존의 Tf-iDf는 단어 빈도수와 문서에서 출현하는 빈도수를 사용하여 단어를 표현한다. 따라서 각 문서를 단어의 가중치값으로 표현하여 문서 유사도를 구할 때 하나의 벡터로 사용한다. Tf-iDf의 단점은 단어와 단어 사이의 개념적 유사도를 고려하지 못한다는 것이다. 예를 들어, 어떠한 문서에서는 '한글'이라고 표현하였고 또 다른 문서에서는 '한국어'라고 표현하였다면 Tf-iDf 방식에서는 서로 다른 단어로 인식하기 때문에 다른 가중치값으로 표현될 수 있다. 그러나 개념적으로는 같은 의미이기 때문에 서로 유사한 값을 부여하면 의미를 고려한 문서의 특징 추출이 가능하다.

word2vec[9]은 위와 같은 문제를 단어가 의미를 갖는 다차원 공간으로 벡터화함으로써 해결해준다. word2vec은 연속적 워드 임베딩(Continuous Word Embedding) 학습 모형이다. word2vec은 기존에 단어를 벡터화시키는 연구보다 계산량을 매우 줄임으로써 빠른 학습을 가능하게 하였다. word2vec은 기본적으로 '비슷한 의미를 가지는 단어는 같은 맥락에 출현한다'는 것을 전제로 한다.

'쇼트트랙은 국내에서 계속 훈련한 뒤 현지로 날아간다.'[10]

위와 같은 문장에서 '쇼트트랙', '국내', '훈련'이라는 단어는 비슷한 주제에서 쉽게 나올 수 있는 단어들이다. 즉 '쇼트트랙'이라는 단어가 나왔으면 '국내', '훈련'이라는 단어가 출현할 확률이 높다는 것을 말한다. word2vec은 문맥 속에서 출현하는 여러 단어들을 학습시켜 주위의 다른 단어가 출현할 때의 가중치 행렬을 구하는데 가중치 행렬의 각 벡터는 단어들의 벡터가 된다.

word2vec에는 스킵그램(Skip-gram)과 CBOW (Continuous Bag of Words) 두 가지 방식이 있는데 본 논문에서는 스킵 그램 방식을 이용한다. 스킵그램은 하나의 단어를 통해 주위의 다른 단어들을 유추하는 방법이다. Fig. 2는 스킵그램 모델을 나타내고 각 변수에 대한 설명은 Table 1과 같다. 또 다른 방법으로 CBOW가 있는데 스킵그램이 하나의 단어를 통해 주위의 여러 단어들을 유추하는 방법이라면 CBOW는 주위의 여러 단어들로부터 하나의 단어를 유추하는 방법이다. 결과적으로 CBOW와 스킵그램은 같은 작업을 거치고 비슷한 단어 벡터를 생성하게 된다. 본 논문에서는 일반적으로 성능이 더 우수하다고 알려진 스킵그램 방식을 이용하여 문서를 벡터로 표현한다[13].

Table 1. Skip-Gram Variable Description

Variable	Description
x_k	input vector
W_{VN}	$[V \times N]$ size weighting matrix
h_i	hidden layer
y_j	output vector

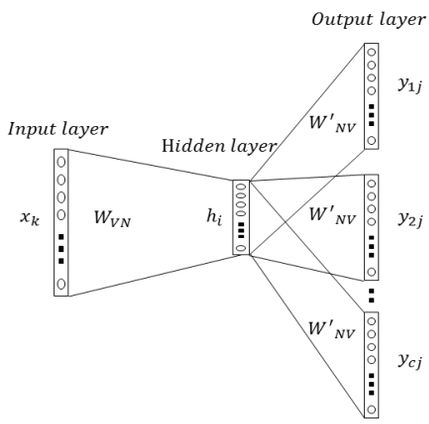

Fig. 2. Skip-gram Model of Word2vec

'쇼트트랙은 국내에서 계속 훈련한 뒤 현지로 날아간다.'와 같은 문장에서 '쇼트트랙'이라는 단어가 입력 데이터로 주어진다면 '국내', '훈련'이라는 단어를 출력 데이터로 학습시킨다. 본 논문에서는 입력층(Input layer)에 주어진 단어들을 원핫 인코딩(one-hot encoding) 방식으로 입력한다. 원핫 인코딩 방식은 단어를 하나의 1의 값을 갖는 비트와 0의 값을 갖는 나머지 비트로 이루어진 것을 말한다. 예를 들어, 'black', 'blue', 'red', 'white', 'yellow'을 벡터로 바꾸어 입력하려 할 때 일반적인 이진 코드를 사용한다면 Table 2의 두 번째 열처럼 나타내고 원핫 인코딩 방식을 사용하면 Table 2의 세 번째 열처럼 나타낸다.

원핫 인코딩은 기계학습에서 범주형 변수뿐만 아니라 분류 문제에서 학습에 사용되는 정답 데이터를 구하기 위해서 사용되며 감성 분석이나 텍스트 생성 모델을 만들 때도 사용되는 방법이다[11]. 원핫 인코딩 방식으로 입력된 벡터는 은닉층(Hidden layer)에 전달되고 출력층(Output layer)에 가중치 행렬(Weight Matrix)이 곱해져서 보내진다. 최종적으로 소프트맥스(softmax) 계산을 통해 정답 단어의 벡터와 비교하여 오차를 계산한 후 오류 역전파를 통해 수정한다. 이와 같은 과정을 반복해 가중치 행렬을 수정하고 충분히 작은 오차가 나오게 되면 가중치 행렬벡터가 입력으로 들어왔던 단어들의 벡터가 된다.

Table 2. Binary Code Example

	Binary code	One-hot encoding
'black'	001	10000
'blue'	010	01000
'red'	011	00100
'white'	100	00010
'yellow'	101	00001

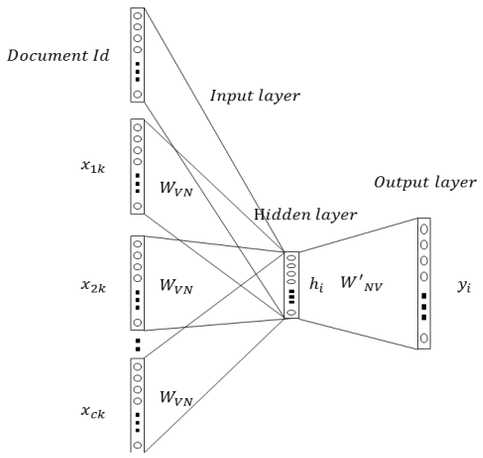

Fig. 3. DM Model of Doc2vec

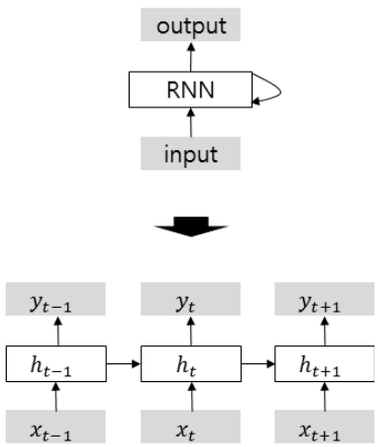

Fig. 4. RNN Basic Structure

doc2vec[12]은 word2vec을 발전시켜 하나의 문장 또는 하나의 문서를 벡터화시키는 방법이다. word2vec과 유사하게 DM(Distributed memory)과 DBOW(distributed bag of words) 두 가지 방식이 있다. doc2vec은 word2vec의 확장이기 때문에 사용 패턴이 유사하다. 즉, 기존의 word2vec에 문서 매트릭스 하나를 추가하여 doc2vec을 만든다. Fig. 3은 doc2vec의 DM 모델의 네트워크를 보여주고 각 변수는 Table 1과 같다. word2vec과 마찬가지로 학습을 하고 최종 학습된 가중치행렬에서 문서 벡터를 추출한다. '쇼트트랙', '국내'라는 단어를 입력데이터로 주고 '훈련이라는' 단어를 출력데이터로 학습할 때 추가적으로 해당 문서용 아이디를 입력하여 같이 학습한다. 최종적으로 학습이 끝나면 추가적으로 입력했던 문서 아이디가 문서용 벡터가 된다. 본 연구에서는 word2vec에서 일반적으로 성능이 더 우수하다고 평가되는 word2vec의 스킵그램 방식과 doc2vec의 DM 모델을 통해 문서를 벡터로 표현한다[13].

3.2 GRU를 이용한 문서 분류

본 논문에서는 문서 분류를 위해 순환 신경망(RNN) 중 GRU를 사용한다. RNN의 기본 구조는 Fig. 4와 같다. 일반적인 인공 신경망 방식인 순방향 신경망(Feed Forward Neural Network)은 입력층의 데이터가 모든 노드를 한 번씩만 지나가게 되지만 RNN에서는 은닉층(Hidden layer)의 결과가 다시 입력으로 들어가게 된다. 즉, 과거에 입력되었던 데이터의 결과값이 다음 입력 데이터와 함께 은닉층 노드에 전달된다. 이와 같은 구조는 일반적인 인공신경망이 할 수 없는 기억능력을 만들어줌으로써 순서에 의미가 있는 시계열 데이터에 적합하게 해준다. 따라서 순서가 중요한 음성, 문자 등 자연어 처리 분야에 적합하다.

RNN은 BPTT(Backpropagation Through Time) 방법으로 오차값을 계산하는데 모든 노드가 곱하기 연산만으로 이루어졌기 때문에 그라디언트가 소실(Vanishing Gradients Problem)되는 문제가 발생한다. LSTM[14]은 셀(cell)이라는 개념을 도입하여 그라디언트 소실 문제를 해결한다. Fig. 5는 LSTM 모델의 셀을 보여주고 Table 3은 셀의 각 변수에 대한 설명이다.

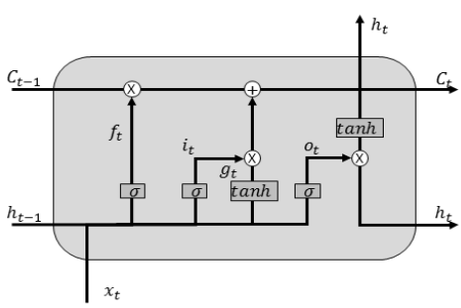

Fig. 5. Structure of a Cell

Table 3. Cell Variable Description

Variable	Description
x_t	Input value of the cell
h_t	Output value of the cell
C_t	Cell state
f_t	forget gate value
i_t, g_t	input gate value
o_t	output gate value
σ	sigmoid function
tanh	hyperbolic tangent function

시그모이드 함수는 S자형의 완만한 커브 형태를 보이는 대표적인 로지스틱 함수이다. 모든 실수 입력값을 0에서부터 1까지 출력해주기 때문에 확률적으로 해석할 때 유용하다. Equation (1)은 시그모이드 함수식을 보여준다.

$$sigmoid(x) = \frac{1}{1+e^{-x}} \tag{1}$$

시그모이드와 비슷한 함수로 하이퍼볼릭 탄젠트가 있다. 시그모이드와 유사한 형태를 보이지만 가장 큰 차이점은 출력값이 -1부터 1까지이다. Equation (2)는 하이퍼볼릭 탄젠트 함수식을 보여준다.

$$\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1} \quad (2)$$

셀은 망각 게이트(forget gate), 입력 게이트(input gate), 출력 게이트(output gate)로 이루어진다. 망각 게이트는 이전 cell의 상태를 얼마만큼 망각할 것인지 결정한다. Equation (3)은 망각 게이트 수식으로 현재 입력 x_t와 전 셀의 출력값 h_{t-1}에 각각의 가중치 행렬을 곱한 후 시그모이드 함수를 통해 0부터 1 사이의 값을 출력한다. 그 후 시그모이드 출력값을 이전 셀의 상태값 C_{t-1}와 곱한다. 따라서 시그모이드를 통해 출력되는 값이 0에 가까워질수록 이전 셀의 상태를 많이 잃게 된다. W_{xh_f}는 x_t와 h_{t-1}에 각각 적용되는 가중치행렬이고 $\_f$, $\_i$, $_o$는 각각 망각, 입력, 출력 게이트에서 사용된다.

$$f_t = \sigma(W_{xh\_f}x_t + W_{hh\_f}h_{t-1} + b_{h\_f}) \quad (3)$$

입력 게이트는 현재 입력과 이전 노드의 출력값을 통해 얼마만큼 셀에 입력할 것인지를 결정한다. Equation (4), Equation (5)는 현재 입력값 x_t과 전 셀의 출력값 h_{t-1}으로 시그모이드 함수와 하이퍼볼릭 탄젠트 함수를 사용하여 현재 셀에 입력될 값을 정한다. Equation (6)을 통해 이전 셀의 상태값에 현재 셀의 상태값을 입력한다.

$$i_t = \sigma(W_{xh\_i}x_t + W_{hh\_i}h_{t-1} + b_{h\_i}) \quad (4)$$

$$g_t = \tanh(W_{xh\_g}x_t + W_{hh\_g}h_{t-1} + b_{h\_g}) \quad (5)$$

$$C_t = f_t \otimes C_{t-1} + i_t \otimes g_t \quad (6)$$

출력 게이트는 셀의 값을 얼마만큼 다음 셀에게 넘겨줄지를 결정한다. Equation (7)을 통해 다음 셀에 넘겨줄 값을 정한다. Equation (8)에서 현재 셀의 상태를 하이퍼볼릭 탄젠트 함수를 사용해 -1부터 1까지 변환한 후 곱해주어 현재 셀의 출력값으로 반환한다.

$$o_t = \sigma(W_{xh\_o}x_t + W_{hh\_o}h_{t-1} + b_{h\_o}) \quad (7)$$

$$h_t = o_t \otimes \tanh(C_t) \quad (8)$$

시그모이드와 하이퍼볼릭 탄젠트 함수로만 이루어진 일반적인 RNN은 곱하기 노드만으로 이루어져있기 때문에 역전파 알고리즘을 수행하게 되면 그라디언트값이 소실되는 문제가 발생한다. LSTM은 셀 개념을 도입하여 세 개의 게이트를 통해 더하기 노드를 추가하여 이러한 문제를 해결한다.

GRU(Gated Recurrent Unit) 방법은 LSTM의 망각 게이트와 입력 게이트를 하나의 업데이트 게이트(update gate)로 통일하고 출력 게이트 대신에 리셋 게이트(reset gate)를 추가하였다[15]. LSTM과 마찬가지로 그라디언트 소실 문제를 해결함과 동시에 게이트를 일부 생략함으로써 계산량을 낮춘 셀의 구조다. Fig. 6은 GRU의 구조를 보여준다.

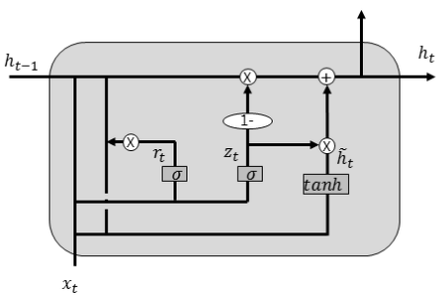

Fig. 6. Structure of GRU

과거 정보를 얼마만큼 반영할지를 결정하기 위한 리셋 게이트와 과거 정보와 현재 정보를 어떻게 조합할지 결정하는 업데이트 게이트는 Equation (9), Equation (10)으로 정의한다.

$$r_t = \sigma(W_{xh\_r}x_t + W_{hh\_r}h_{t-1}) \quad (9)$$

$$z_t = \sigma(W_{xh\_z}x_t + W_{hh\_z}h_{t-1}) \quad (10)$$

GRU에서는 먼저 현 시점에서 유지할 정보를 Equation (11)과 같이 정의한다. 현재의 정보는 $W_{xh}x_t$가 되고 과거 정보는 $W_{rh}h_{t-1}$가 된다. 그리고 과거 정보를 얼마나 반영할지 리셋 게이트 r_t를 통해 결정한다. 리셋 게이트는 시그모이드 함수를 통해 값을 가지므로 0부터 1사이의 값을 갖는다. 따라서 0이라면 과거의 정보는 모두 지우고 1이라면 과거 정보를 모두 기억하는 것을 의미한다. 리셋 게이트 값과 상관없이 현재의 정보는 무조건 반영된다.

$$\tilde{h_t} = \tanh(W_{xh}x_t + r_t \otimes W_{rh}h_{t-1}) \quad (11)$$

수정된 현재의 정보를 출력값으로 내보내기 위한 업데이트 절차는 Equation (12)와 같다. 과거의 정보 h_{t-1}과 현재의 정보 $\tilde{h_t}$를 업데이트 게이트 z_t값을 통해 결정한다. 업데이트 게이트 또한 시그모이드 함수로 0부터 1의 값을 가지므로 0의 값을 갖는다면 과거 정보는 모두 지우고 현재 정보만을 기억하게 된다.

$$h_t = z_t \otimes h_{t-1} + (1-z_t) \otimes \tilde{h_t} \quad (12)$$

4. 실험 및 결과

4.1 데이터 집합

본 논문에서 분류시스템의 학습과 분류 실험을 위하여 사용된 데이터는 '한국일보 20000[10]' (HKIB-20000), '20-news[16]', 'AG news[17]'이다. 한국일보-20000 실험 데이터 집합은 한국일보 기사 중 20,000건을 추출하여 3단계 분류체계의 모든 노드에 기사를 할당하여 구축한 문서 범주화용 실험 데이터 집합이다. Table 4는 실험에 사용된 문서 범주와 해당 문서 개수를 보여준다.

'20-news'는 영문으로 작성되었고 18,828개의 신문기사가

주제별로 20개의 범주로 분류되어있다. 실험 비교를 위해 범주의 경계가 확실히 구분되는 12개의 범주만을 사용하였다. 다음 Table 5는 실험에 사용된 문서 범주와 해당 문서 개수를 보여준다.

'AG news'는 총 127,600개의 신문기사를 모은 것으로 4개의 범주로 분류되어 있다. 각 범주에는 31,900개의 문서가 존재한다. Table 6은 실험에 사용된 데이터 집합의 설명이다.

Table 4. HKIB-20000 Category

Class	Number of documents
Diseases	196
Bank	282
Science	188
Baseball	325
Broadcasting	518
Buddhism	141
Army	774
Civil engineering	272
computer	481
Travel tourism	182

Table 5. 20-News Category

Class	Number of documents
comp.os.ms-windows.misc	985
comp.sys.ibm.pc.hardware	982
rec.autos	990
rec.sport.baseball	994
sci.med	990
sci.crypt	991
sci.space	987
talk.politics.misc	775
talk.religion.misc	628
alt.atheism	799
misc.forsale	972
soc.religion.christian	997

Table 6. Description of data sets

	Number of documents	Number of class	Language
HKIB-20000	20,000	10	Korean
20-news	18,828	12	English
AG news	127,600	4	English

4.2 실험 방법

실험을 위해 형태소 분석은 파이썬의 Konlpy[18] 라이브러리에서 twitter 패키지를 사용하여 실시하였고 DF(Document Frequency)를 추출하여 빈도수가 50%가 넘는 어절은 불용어로 처리하였다. 데이터의 90%를 훈련데이터로 사용하고 10%를 테스트 데이터로 사용하였다. word2vec을 이용하여 단어를 벡터로 표현하고 이 단어벡터를 doc2vec에 문서 아이디와 함께 입력하여 문서 벡터를 생성한다. 문서 벡터를 RNN의 GRU 분류기에 입력으로 사용해 정확도(Accuracy)를 구하여 성능을 평가하였다. GRU의 문서벡터 입력차원은 실험을 통해 가장 높은 정확도를 보이는 100차원으로 결정하였다. 정확도는 Equation (16)과 같으며 정답 레이블과 일치하는 수를 측정된 전체 문서 개수로 나눠준다(Table 7). 실험에 사용된 RNN 파라미터 설정은 Table 8에서 보여준다.

$$\mathrm{Precision} = \frac{TP}{TP+FP} \tag{13}$$

$$Recall = \frac{TP}{TP+FN} \tag{14}$$

$$F1-measure = \frac{2\times \mathrm{Precision}\times Recall}{\mathrm{Precision}+Recall} \tag{15}$$

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} \tag{16}$$

Table 7. Accuracy Evaluation Metrics

		True condition	
		Positive	Negative
Predicted condition	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Table 8. GRU Parameter Variable

Parameter	Value
learning late	0.001
hidden layer	100
batch size	100
epoch	50

4.3 실험결과

Table 9는 word2vec과 doc2vec이 문서의 특징을 얼마나 잘 보여주는지 알기 위해 다른 자질들과 비교 실험한 것을 보여준다. 분류기는 GRU 모델을 사용하였다. 성능 평가 모델은 Equation (16)과 같은 정확도와 F1-measure로 측정하였다.

Table 9. Experimental Results with Data Sets

Features + classifier	HKIB-2000		20-news		AG News	
	Acc	F1	Acc	F1	Acc	F1
tfidf+GRU	0.857	0.843	0.796	0.799	0.832	0.846
word2vec+GRU	0.870	0.859	0.811	0.803	0.841	0.838
word2vec+ doc2vec+GRU	0.901	0.904	0.882	0.879	0.877	0.865

실험 결과 대부분의 경우에 tfidf 보다 word2vec 특징이 더 좋은 결과를 보이고 word2vec+doc2vec 특징이 실험대상 데이터 집합에 대해 가장 좋은 실험결과를 보였다.

Table 10은 다른 방법들과의 결과를 비교하여 보여준다. 본 논문에서 제안하는 word2vec과 doc2vec을 활용하고 RNN의 GRU 분류기를 사용하였을 때 정확도가 더 높은 것을 확인 할 수 있다. 또한 word2vec 모델만을 사용했을 때보다 doc2vec 모델을 같이 사용할 경우 성능이 향상되는 것을 확인하였다.

Table 10. Experimental Results with All Datasets

	HKIB-2000	20-news	AG News
tfidf+SVM [19]	0.831	0.775	0.826
word2vec+CNN [20]	0.854	0.730	0.835
word2vec+LSTM [20]	0.842	0.830	0.833
Discriminative LSTM [21]	-	-	0.851
word2vec+doc2vec+CNN [22]	0.875	0.847	0.832
word2vec+doc2vec+LSTM	0.889	0.846	0.847
word2vec+doc2vec+GRU	**0.901**	**0.882**	**0.877**

'한국일보-20000'에서 정확도가 90%가 넘은 것은 한글 문서에서도 단어의 의미와 문맥을 고려한 word2vec과 doc2vec 방법이 효과가 있다는 것을 알 수 있다. 또한 일반적으로 CNN이 자연어처리 분야에서 LSTM보다 더 높은 성능을 내왔지만[8, 22] 실험 결과 LSTM의 발전된 형태인 GRU가 CNN보다 성능이 평균 3.5% 좋은 것을 확인할 수 있었다.

doc2vec에서 표현되는 문서의 특징 벡터 크기에 따른 실험 성능을 알아보기 위해 50, 100, 200, 300으로 설정하여 실험하였다. 데이터는 'AG News'를 사용하였고 분류기를 GRU로 사용하여 실험한 결과 Fig. 7과 같으며 특징 벡터의 크기가 100일 때 가장 좋은 성능을 보였다.

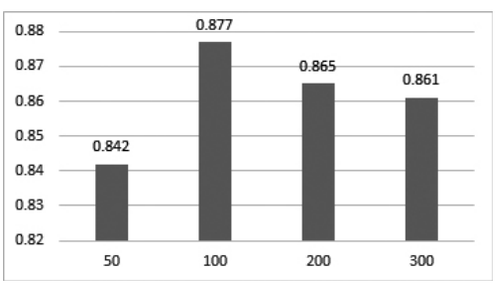

Fig. 7. Accuracy Per Feature Vector Size

실험의 학습 반복 수(epoch)에 따른 실험 성능을 알아보기 위해 10, 30, 50, 70, 90으로 설정하여 'AG News' 데이터에 대해 GRU를 사용하였다. 실험 결과 실험 반복 회수가 커질수록 실험 성능 또한 향상되지만 일정 수 이상 지나면 큰 차이가 없는 것을 확인하였다(Fig. 8).

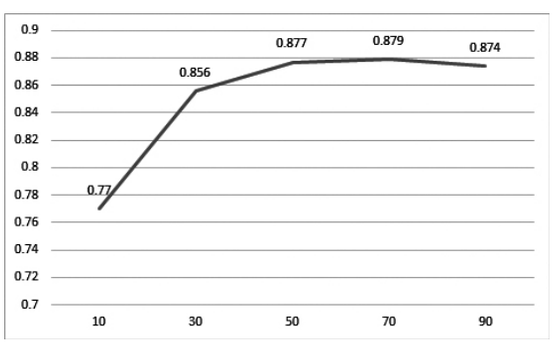

Fig. 8. Accuracy Per Epoch

5. 실험결과 분석

자질 비교 실험에서 모든 경우에 word2vec+doc2vec의 성능이 가장 우수했다. 고전적인 tfidf 방법도 준수한 성능을 보였지만 딥러닝 기반의 word2vec과 doc2vec을 사용했을 때 좀 더 문서를 잘 표현한 것을 알 수 있다.

HKIB-2000 문서 집합의 경우 한글로 작성되어져 있고 하나의 문서가 A4 한 장 분량의 내용이 담겨져 있어서 특징 추출단계에서 상대적으로 충분한 정보를 포함하고 있어서 실험 성능이 높게 나온 것으로 판단된다. 반면에 AG News는 문서의 개수가 다른 문서 집합에 비해 월등히 많긴 하지만 한 문서의 크기가 2~3 문장으로 이루어져서 문서를 나타내는 특징 벡터를 구성하기에는 충분하지 않기 때문에 성능이 다른 실험 집합과 비교해 다소 낮게 나온 것으로 판단된다. 20-news는 문서의 개수와 문서의 분량이 HKIB-2000과 비슷하지만 클래스 개수가 12개로 많아서 성능이 HKIB-2000에 비해 낮게 나온 것으로 판단된다.

실험 결과 다른 분류기 모델보다 GRU 모델을 사용했을 때 성능이 향상되었는데, 이것은 특정 단어가 출현할 확률이 가까이 인접해 있는 단어뿐만 아니라 멀리 떨어져 있는 단어의 영향력도 고려되었기 때문이라고 판단된다. 따라서 GRU 모델이 CNN이나 LSTM보다 자연어 처리 분야에서 좀 더 좋은 성능을 보였다.

본 논문에서는 형태소 분석을 통해 문서에서 명사만을 추출하였다. 그러나 문맥을 이해하기 위해서는 명사뿐만 아니라 형용사나 동사같은 주요 품사도 필요하다. 향후 연구에서는 품사를 고려하여 실험 성능을 높일 수 있는 체계적인 분석이 필요하다.

6. 결 론

본 논문에서는 문서분류 성능을 향상시키기 위해 딥 러닝 기반의 색인 방법과 분류 모델을 사용하였다. 단어의 빈도기반 문서 벡터 표현 방법인 Tf-iDf 방식의 단점을 해결하는 방법인 단어의 의미와 문맥을 고려할 수 있는 word2vec과 doc2vec을 사용하여 문서를 벡터로 색인하였다. RNN 분류기를 통해 다른 모델과 비교하여 성능이 향상된 것을 확인하였다. RNN 분류 모델 중에서도 과거의 입력들을 장기간 기억하면서 그라디언트 소실 문제를 해결해주고 계산량을 줄여주는 GRU모델을 사용하였다. 실험 결과 CNN 분류모델보다 GRU 모델에서 약 3.5% 정도 성능이 향상되었다.

References

[1] J. H. Kim, J. H, Kim, K. M. Kim, and B. T. Zhang, "Large-Scale Text Classification with Convolution Neural Networks," *Korean Information Science Society Conference Proceedings*, pp.792-794, 2015.

[2] P. Soucy and G. W. Mineau, "Beyond TFIDF weighting for text categorization in the vector space model," *IJCAI*, Vol. 5, 2005.

[3] C. H. Lee, Chang and S. C. Park, "BPNN Algorithm Using SVD for Korean Document Classification," *Journal of the Korea Industrial Information System Society*, Vol.15, No.2 pp.49-57, 2010.
[4] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," Communications of the ACM, Vol.18, No.11, pp.613-620, 1975.
[5] J. W. Hwang and Y, J, Ko, "A Studyon Sentiment Features Extractionand Their Weight Boosting Methodfor Korean Document Sentiment Classification," *Journal of KISS: Computing Practice and Letters*, Vol.14, No.3, pp.336-340, 2008.
[6] Y., Goldberg and O. Levy, "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722*, 2014.
[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, 2012.
[8] J. M. Kim and J. H. Lee, "Text Document Classification Based on Recurrent Neural Network Using Word2vec," *Journal of Korea Institute of Intelligent Systems*, Vol.27, No.6, pp.560-565, 2017.
[9] T. Mikolov, I. Sutskever, K. Chen, G. S., Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," Advances in *Neural Information Processing Systems*, 2013.
[10] HANTEC Data Set [Internet], http://www.kristalinfo.com/TestCollections/#hkib
[11] M. Cassel, and F. Lima, "Evaluating one-hot encoding finite state machines for SEU reliability in SRAM-based FPGAs," *On-Line Testing Symposium, 2006. IOLTS 2006. 12th IEEE International*, IEEE, 2006.
[12] J. H. Lau, and T. Baldwin, "An empirical evaluation of doc2vec with practical insights into document embedding generation," *arXiv preprint arXiv:1607.05368*.
[13] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," *International Conference on Machine Learning*, 2014.
[14] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," pp.850-855, 1999.
[15] K. Cho, B. Van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
[16] 20Newsgroups Data Set [Internet], http://qwone.com/~jason/20Newsgroups/
[17] Text Classification Data Sets [Internet], http://goo.gl/JyCnZq
[18] Python Package for Natural Language Processing [Internet], http://konlpy.org/en/v0.4.4/
[19] J. Y. Lee, "A Study on the Improvement of Document Classification Performance of SVM Classifier Using Document Similarity," *Journal of the Korean Society for Information Management*, Vol.22 No.3, pp.261-287, 2005.
[20] J. M. Kim and J. H. Lee, "A study on RNN based document classification using Word2vec," *Journal of Korea Institute of Intelligent Systems*, Vol.27, No.6, pp.560-565, 2017.
[21] Jiang, Z., Zhang, S., & Zeng, J. "A hybrid generative/discriminative method for semi-supervised classification," *Knowledge-Based Systems*, Vol.37, pp.137-145, 2013.
[22] N. H. Kim and H. J. Yang, "Classification of Hangul Documents Based on CNN Using Document Indexing Method Considering Meaning and Order of Words," *Korean Computer Education Association Conference Paper*, Vol.21, No.2, pp.41-45, 2017.

주 종 민
https://orcid.org/0000-0001-7742-7932
e-mail : whdals099@gmail.com
2016년 전남대학교 전자컴퓨터공학부(학사)
2016년~현 재 전남대학교 전자컴퓨터공학부 석사과정
관심분야 : 기계학습, 자연어 처리

김 남 훈
https://orcid.org/0000-0001-6628-8690
e-mail : knh3767@naver.com
2015년 조선대학교 경제학부(학사)
2018년 전남대학교 전자컴퓨터공학부 (석사)
관심분야 : 자연언어처리, 기계학습, 영상처리

양 형 정
https://orcid.org/0000-0003-3024-5060
e-mail : hjyang@jnu.ac.kr
1991년 전북대학교 전산통계학과(학사)
1993년 전북대학교 전산통계학과(석사)
1998년 전북대학교 전산통계학과(박사)
2003년~2005년 카네기멜런대학교 연구원
2005년~현 재 전남대학교 전자컴퓨터공학부 교수
관심분야 : 영상처리, 데이터 마이닝, 기계학습

박 혁 로
https://orcid.org/0000-0002-5594-6675
e-mail : hyukro@jnu.ac.kr
1987년 서울대학교 컴퓨터공학과(학사)
1989년 KAIST 전산학과(석사)
1996년 KAIST 전산학과(박사)
1994년~1998년 한국과학기술정보연구원 선임연구원
1999년~현 재 전남대학교 전자컴퓨터공학부 교수
관심분야 : 정보검색, 자연언어처리, 기계학습