졸 업 논 문

제목: 자연어처리 기술을 활용한 생성형 문서요약

2018 년 12 월 17 일

한국외국어대학교 공과대학 정보통신공학과

지도 교수 : 김 희 동

학 번 201303829 성 명 황 정 용

학 번 201301711 성 명 손 효 빈

제 목 : 자연어처리 기술을 활용한 생성형 문서요약

대 학 :공 과 대 학　　학 과 : 정 보 통 신 공 학 과

학 번 : 201303829　　성 명 : 황 정 용
201301711　　손 효 빈

이 논문을 제출 하오니 승인하여 주십시오.

2018 년 12 월 17 일

성 명 : 황 정 용 (인)
손 효 빈 (인)

위 학생의 논문 제출을 승인함.

2018 년 12 월 17 일

지 도 교 수 : 김 희 동 (인)

목차

그림목차

수식목차

1. 서론

1.1. 주제 선정 배경

인터넷과 정보통신 기술이 발달하면서 이제는 누구나 수 많은 정보에 접근 가능하게 되었다. 그러나 정보에 접근하기는 쉬워졌으나 이러한 정보를 잘 이용하는 것은 다른 문제이다. 자신에게 필요하고 유용한 정보만을 적은 시간 안에 획득하고 이용할 수 있다면 보다 효율적으로 인터넷과 정보통신 기술을 활용할 수 있게 될 것이다. 또한 과거 일부의 사람만이 정보의 확산에 기여했던 것과는 다르게 최근에는 소셜 네트워크 서비스(SNS, Social Networking Service)의 확산에 따라 정보의 생성 및 전파가 그 어느 때와 다르게 활발하고 다양하게 이루어지고 있다. 이와 같이 누구나 정보의 생산과 전파가 가능하게 되어 많은 장점도 있으나 정제되지 않은 데이터들이 범람하고 있다. 한정된 시간 내에 신뢰성, 의미, 가치 있는 정보를 찾아내 위해서는 정보의 요약은 반드시 필요하다. 최근 한 조사결과에 따르면 현대인들의 미디어 소비에서 요약형 정보가 훨씬 선호되고 있음을 알 수 있는데 이 결과를 그림 1 를 통해 나타내었다.

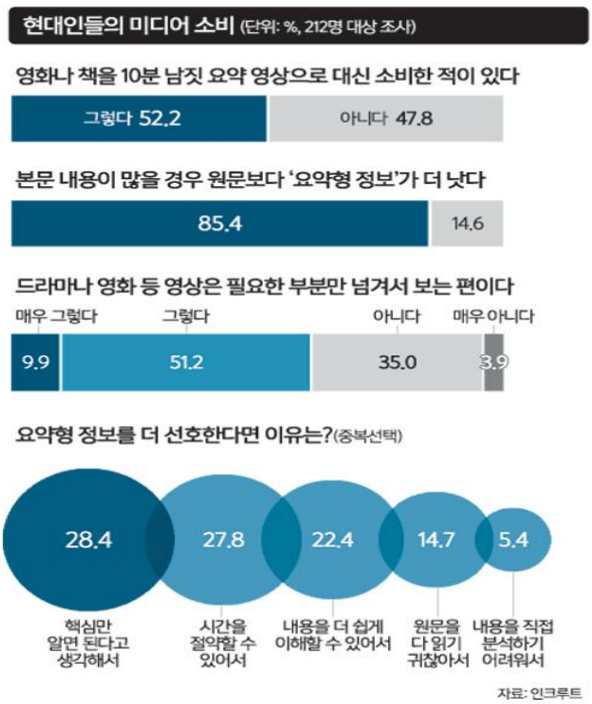


그림 1. 현대인들의 미디어 소비

출처: http://www.segye.com/newsView/20180622004444

그림 1 에서 전체 응답자의 85.4% 는 요약형 정보를 선호하는 것으로 나타났고 요약형 정보의 선호 이유로는 핵심만 알면 된다고 생각하는 것과 시간을 절약할 수 있는 것, 내용을 더 쉽게 이해할 수 있는 것 등의 이유를

들었다. 이처럼 요약형 정보는 현대인의 정보 이용에 있어서 중요한 역할을 담당하고 있고, 요구되고 있는 상황이다. 이에 따라서 요약기술 또한 요구 되고 있다. 현재 인터넷 상에서 생산되고 공유되는 정보의 대다수는 텍스트 정보이다. 따라서 많은 줄의 텍스트 데이터의 내용을 파악하고 중복적인 정보를 제거하는 문서요약 기법의 필요성이 대두되고 있으며 중요성 또한 점점 커지고 있다 본 논문에서는 수 많은 텍스트 정보들을 효율적으로 확보, 이용하기 위한 방법으로 NLP(Natural Language Processing, 자연어 처리)를 활용하여 텍스트 정보들을 요약하는 방법들에 대해 연구하려 한다.

1.2. 연구 목표

문서요약은 주어진 원문서 내용을 이해하여 원문서 보다 짧은 축약된 문서 또는 문장을 생성하는 것이다. 문서요약에서 가장 중요한 것은 입력문서를 정확히 분석하고, 중요한 정보를 선택하고, 덜 중요한 정보는 효율적으로 걸러내고, 관련 내용들을 잘 조합해서 사람들이 이해하는 문장을 만들어 내야 한다. 문서요약의 방법은 크게 2 가지로 추출형 문서요약(Extractive text summarization)과 생성형 문서요약(Abstractive text summarization)으로 구분할 수 있다. 추출형 문서요약은 원본 문서에 존재하는 단어, 구 또는 문장을 그대로 추출하여 사용자에게 적합한 요약문을 생성한다. 생성형 문서요약 방법은 원본 문서를 토대로 새로운 문장을 생성하여 자연스러운 요약문을 생성하는 방법이다. 그림 2 에서 추출형 문서요약과 생성형 문서요약의 차이의 이해를 돕기 위한 예시를 나타내었다.

Original Text: *Alice and Bob took the train to visit the zoo. They saw **a baby giraffe, a lion, and a flock of colorful tropical birds.***
Extractive Summary: *Alice and Bob visit the zoo. saw **a flock of birds.***
Abstractive summary: *Alice and Bob visited the zoo and saw **animals and birds.***

그림 2. 추출형 문서요약과 생성형 문서요약의 예시

출처:https://ai.googleblog.com/2016/08/text-summarization-with-tensorflow.html

위 그림 2 에서 원본문서는 위에서 언급한 두 가지 방법(추출형 문서요약 과 생성형 문서요약)으로 각각 요약되었다. 추출형 문서요약 에서는 원본 문서로부터 그대로 구와 단어가 추출되어 시제와 목적어가 정확히 요약문에 담기지 못했다. 생성형 문서요약에서는 원본문서의 시제와 목적어의 의미가 요약문에 잘 표현되었다. 그림 2 처럼 추출형 문서요약은 원본문서에 존재하는 그대로 추출해서 문장을 생성하기 때문에 인간이 읽기에 자연스럽지 못하여 가독성이 부족하고, 정보의 누락이 발생할 수 있는 단점이 있다. 추출형 문서요약은 알고리즘을 통해 원본문서 전체를 계산하여 중요하다고 판단되는 부분만을 추출해서 배치하는 작업이므로 생성형 문장요약보다 상대적으로 쉬운 접근방법이다. 따라서 기존의 문서요약 기술의 연구는 추출형 문서요약 모델을 중심으로 진행되어 왔다. 반면 생성형 문서요약은 자연어의 이해 및 문장 생성기술이 필수적이었으므로 추출형 문장요약보다 구현상 어려움이 있다. 그런데 최근 기계학습이 발달하면서 구글에서는 기계번역에서

자주 사용되는 Sequence to Sequence Model 을 기반으로 생성형 문서요약을 수행 하였고 코드를 공개하였다. 구글에서는 영어에 대해 문서요약을 진행하였고 성과를 거두었다. 위에서 추출형 문서요약 보다 생성형 문서요약이 더 좋은 이유를 설명하였다. 본 연구에서는 구글에서 제공한 오픈소스를 활용해 한국어에서 생성형 문서요약을 수행해보고 추출형 문서요약과 성능을 비교해 보았다. 비교를 위한 추출형 문서요약 방법은 TextRank 방식을 선택하였고 이와 함께 생성형 문서요약의 제한사항과 한계점에 대해 확인해 볼 것이다.

2. 관련 기술동향 조사 및 분석

문서요약은 관점에 따라 여러가지 방법으로 분류될 수 있으나 본 논문에서는 추출형 방법과 생성형 방법을 중심으로 설명한다.

2.1. 추출형 문서요약 방법(Extractive text Summarization)

현재 네이버와 다음은 뉴스 페이지에서 자동 요약 기능을 제공하고 있다. 기사 본문의 키워드, 문장 중요도 등을 판단하여 최대 3 문장까지 제공하며 추출형 요약 방법으로 요약문을 추출하고 있다. 이처럼 추출형 문서요약은 문서내의 단어, 구 또는 문장 중 중요하다고 판단되는 부분을 추출하여 요약문을 완성하는 것이다. 그렇다면 무엇을 기준으로 스코어(가중치)를 매기고 중요한 부분을 판단할 것인가가 핵심이 될 것이다. 이러한 방법에는 단어 빈도수 기반, 정보 추출 등 현재까지 발표된 다양한 방법이 있다.

본 절에서는 추출형 문서요약 방법 중 우리가 선택한 TextRank 알고리즘에서 사용되는 기술들을 간략히 소개한다. 추출형 문서요약의 장점으로는 원 문서에 있는 문장을 선택하여 요약문을 생성하므로 구현이 비교적 쉽고 문법적으로 완성이 되어있는 문장들을 추출하므로 문법적으로 오류가 날 가능성이 적다. 그리고 문장 또는 단어를 변경하거나 추가되는 것이 없음으로서 문장의 의미가 훼손되지 않는다. 그러나 이것이 단점으로도 작용하여 추출형 문서요약에서는 원본문서에 없는 단어는 사용이 불가능하다. 따라서 요약문을 추상화시키거나 좀 더 이해하기 쉽게 하기 만들기 위한 작업들이 추가되어야 하기 때문에 사실상 불가능하거나 매우 복잡해진다..

2.1.1. Graph-based ranking model

원본문서 내의 문장들을 연결된 그래프 구조로 표현하여 노드들의 관계를 중심으로 요약문을 작성하는 방법이다. 문서내의 각 문장들을 노드로 하고 노드간의 유사성(Similarity)을 계산하고 노드를 연결함으로써 그래프를 만든다. 유사성이 높은 부분들이 서로 연결되고 이를 통해 각각의 문장을 Ranking 할 수 있다. 그 중 높은 값을 갖는 부분들을 추출하면 추출형 요약문이 완성된다. 대표적으로 알려진 방법으로는 TextRank 알고리즘이 있다. TextRank 는 PageRank 알고리즘에서 영향을 받은 방법이다. PageRank 는 구글 검색의 기초가 되는 방법으로 각 인터넷 페이지 간의 관계를 링크시키고 이 관계로부터 페이지를 평가하는 방법이다. TextRank 는 이를

문장에 응용한 방법이다. 각 노드는 단어나 문장 등으로 하고 엣지는 단어, 참조,문장의 유사도등을 활용한다. 이를 이용하여 PageRank 와 유사하게 관계들로부터 문장을 평가한다. 그림 3 은 노드간의 유사성을 그래프로 그리는 것의 이해를 돕기 위한 예시 이다.

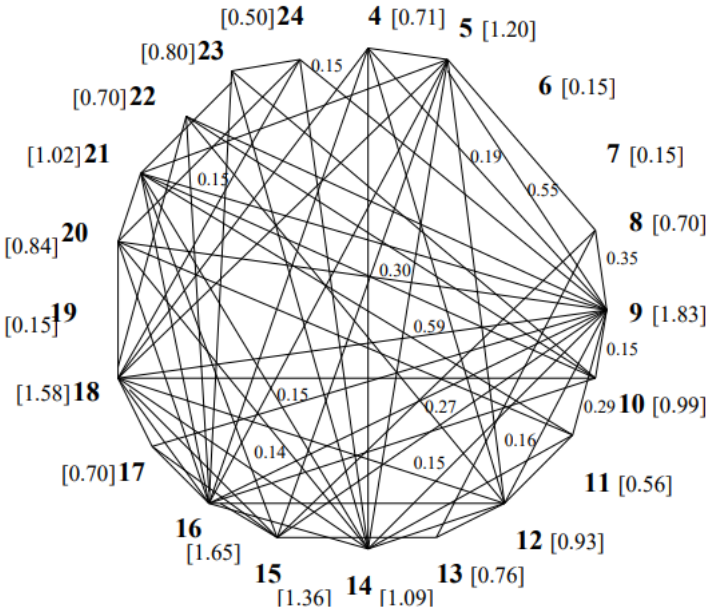


그림 3. 문장 추출을 위한 그래프 생성의 예

출처:TextRank: Bringing Order into Texts

2.1.2. TF-IDF

각 문장(노드) 사이의 유사성을 계산하는 방법으로 TF-IDF(Term Frequency - Inverse Document Frequency)가 연구되었다. 예를 들어 우리는 직감적으로 한 특정 문서 내에서 특정 단어의 출현빈도가 높다면 그 단어를 우리가 필요한 정보를 갖는 중요한 단어라고 유추할 수 있다. 그러나 해당 단어가 방금 확인한 문서뿐만 아니라 대다수의 다른 문서에서도 자주 등장하는 단어라면 그 단어의 중요도는 상대적으로 낮아질 것이다. TF-IDF 는 이런 상황을 가정하고 매우 일반적인 단어에는 가중치를 적게 줌으로써 가중치를 적용하는 방법이다. 이 가중치는 식 (1)과 같이 계산된다.

$$q(w) = f_d(w) * log \frac{|D|}{f_d(w)} \qquad (1)$$

식 (1) 의 $f_d(w)$ 는 TF(Term frequency)는 단어빈도, |D|는 전체 문서의 수, $f_d(w)$는 단어 w가 포함된 문서의 수를 의미한다. $f_d(w)$가 높으면 중요한 단어라고 판단할 수 있다. 그러나 해당 단어가 전체 문서에서 매우 자주 사용되는 경우에 이는 이 단어 매우 일반적인 단어라는 것을 의미한다. IDF (Inverse Document Frequency)는 한 단어가 전체 문서에서 얼마나 공통적으로 나타나는지를 나타내며 $log \frac{|D|}{f_d(w)}$에 해당한다.

2.2. 생성형 문서요약(Abstractive Text Summarization) 방법

생성형 문서요약은 문서의 전체 내용을 이해하고 사람이 이해하기 쉬운 새로운 문장을 생성할 수 있다는 점에서 추출형 문서요약보다 장점을 가지고 있다. 하지만 문법적으로 정확하고 문맥에 맞는 자연스러운 문장을 만드는 것이 매우 어렵다. 기본적으로 언어가 가진 유연성과 데이터의 차원의 문제등 기술적으로 구현에 한계점을 갖고 있다. 한계점 중 하나로는 입력 데이터의 차원이 고정되어 있다는 것이다. 언어는 유연성을 갖는 것으로서 한 문장에서도 여러 단어로 이루어지게 된다. 이 단어들은 유사한 의미를 가질 수도 있고 연관되어 있을 수도 있으만 한편 완전히 다른 의미를 가질 수도 있다. 따라서 문서를 데이터로 입력할 때 입력 데이터의 차원이 매우 높아지게 되고되고 이를 실제 구현하는 데에는 많은 어려움이 있다. 그러나 최근 기계학습에서의 기술 진보와 더불어 심층 학습을 적용하여 문장 번역 분야에서는 많은 성과를 이루어 냈고 여기에 사용된 기술들을 적용하여 위에서 언급한 문제들을 해결하려는 도전들이 문서요약에서도 이루어졌다. 2015 년 Facebook 에서는 RNN 을 활용한 생성형 문서요약을 논문으로 발표하였다[12]. 2016 년 Google 에서도 Sequ ence to Sequence Model 을 기반으로 생성형 문서요약을 수행하였다.

2.2.1. Sequence to Sequence

Sequence to Sequence 는 LSTM(Long Short-Term Memory), GRU(Gated Recurrent Unit) 등 RNN(Recurr ent Neural Network) Cell 을 길고 깊게 쌓아서 복잡하고 방대한 Sequence Data 을 처리하는 데 특화된 모델이다. 그림 4 는 Sequence to Sequence Model 을 나타내고 있다. Sequence to Sequence Model 은 크게 인코더(Encoder)와 디코더(Decoder) 두 파트로 나뉘게 된다. 이 Encoder-Decoder 모델의 경우 주로 기계번역에서 사용되었던 방법으로 이를 문서요약에 적용하는 다양한 방법들이 연구되고있다. 먼저 입력의 경우(Encoder 의 입력)에 Source language(번역하고자 하는 언어)을 넣게 되고 Hidden Layer(LSTM)들을 거쳐 하나의 Context Vector 로 인코딩된다. 출력의 경우(Decoder 의 출력)는 Target language(번역 된 언어)를 넣게 되어서 훈련시키는 방법이다. 즉 인코딩된 Context Vector 는 디코더에 입력되어 디코더는 적합한 자연어 응답을 생성한다.

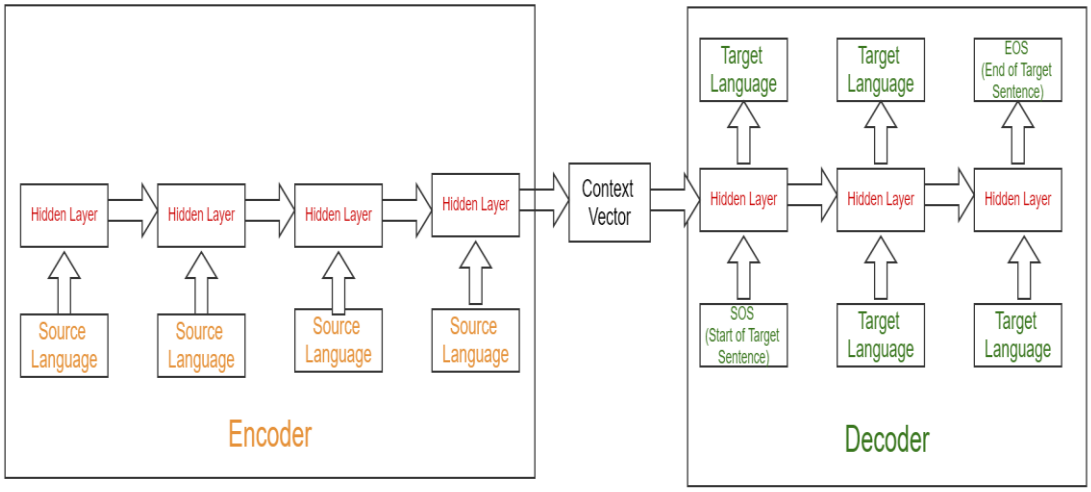


그림 4. Sequence to Sequence Model

2.2.2. RNN (Recurrent Neural Network)

RNN 은 히든노드가 방향을 가진 엣지로 연결돼 순환구조를 이루는(Directed cycle)인공신경망의 한 종류이다.그림 5 에서 RNN 의 기본구조를 나타내었다. x_t는 입력, y_t는 출력을 의미한다. 현재 상태의 Hidden state h_t는 직전 시점의 Hidden state h_{t-1}을 받아 갱신되는 구조이다. RNN 의 장점 중 하나는 이전의 정보를 현재의 문제를 해결하는데 활용할 수 있다는 점이다. 따라서 음성, 문자 등 순차적으로 등장하는 데이터 처리에 적합한 모델로 알려져 있다. 그러나 본 논문과 같이 문장의 길이가 일정하지 않고 다양한 길이의 정보를 가질 때 즉, 이전 정보와 그 정보가 필요로 한 곳의 거리가 크다면 장기의존성 문제가 발생한다. 이에 대하여 RNN 으로는 장기의존성 문제를 해결 할 수 없다는 것이 이전 연구들로부터 알려져 있다. 다음 절에서는 이와같은 RNN 의 한 종류로서 장기의존성 문제를 극복하는 방법을 소개한다.

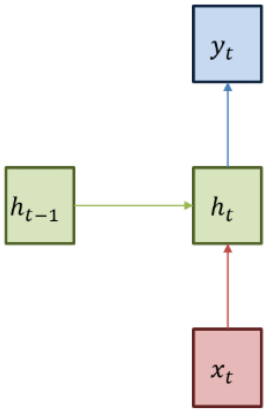


그림 5. RNN 의 기본구조

출처: https://ratsgo.github.io/natural%20language%20processing/2017/03/09/rnnlstm/

2.2.3. RNN-LSTM

본 논문의 목표인 자연언어처리를 활용한 생성형 문장 요약을 하기 위해서는 문서와 이를 이루는 문장이란 데이터가 갖는 특성에 대해 정확한 인식이 필요하다. 텍스트 데이터는 이미지와는 다르게 서로 인접한 단어와 Discrete 하다는 특성을 갖는다. 이는 특정단어또는 문장이 주변부 단어 또는 문장과 연관성이 높을 수도 있고 낮을 수도 있음을 의미하며 생성형 요약을 구현하기 위해서는 문장의 길이가 길어져도 각각의 정보를 오래 기억할 수 있어야한다. 예를 들어 "나는 학교에서 밥을 영희와 먹었다." 라는 문장에서 주어인 '나'와 '영희'는 위치적으로 서로 가장 떨어져 먼 거리에 위치해 있다. 그런데 주어인 나와 영희의 관계가 가족이라면 나머지 단어들 '학교', '밥' 등 보다 '나'는 '영희'와 연관성이 더 높다. 신경망은 이 문장을 받아들일 때 '나'라는 정보를 멀리 떨어진 '영희' 라는 정보와 연관지을 수 있어야한다. 이러한 텍스트가 갖는 특징에대한 방안으로 Hidden Layer 에 LSTM 을 사용하게 된다.

LSTM 은 장기 의존성을 해결하기 위한 방안이다. LSTM 은 Forget gate 와 Input gate 가 특징으로 Forget gate 는 과거 정보를 잊기 위한 게이트이다. Input gate 는 현재 정보를 기억하기 위한 게이트이다. 두 게이트 모두 앞에 σ(시그모이드)를 곱하여 0~1 사이에 값을 가지게 된다. σ(시그모이드)를 곱한 Forget gate 와 Input gate 를 통해 과거의 정보와 현재의 정보를 얼마나 기억할 것인가를 정하여 장기 의존성의 문제를 해결한다. 그림 6 과 그림 7 에서 이러한 특성을 보여주는 LSTM 에 대하여 나타내었다. 식 (2), (3)은 그림 7 의 Forget Gate 와 Input Gate 의 식을 나타낸다.

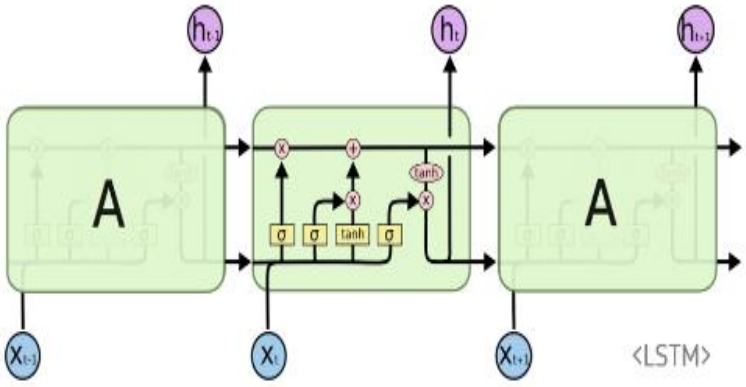


그림 6. LSTM 의 기본구조

출처:https://ratsgo.github.io/natural%20language%20processing/2017/03/09/rnnlstm/

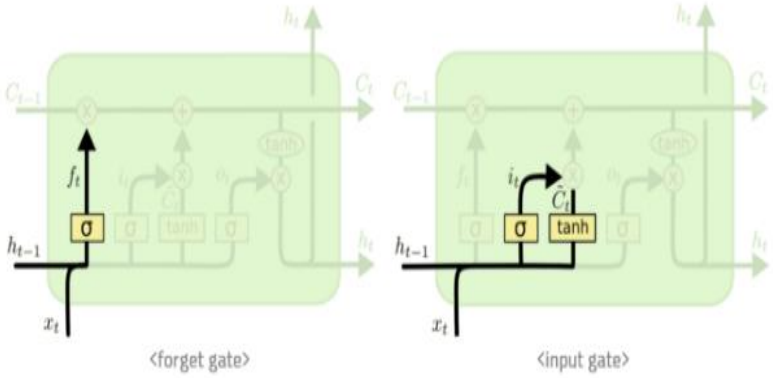


그림 7. Forget gate, Input gate

출처:https://ratsgo.github.io/natural%20language%20processing/2017/03/09/rnnlstm/

$$\text{Forget gate}: \ f_t = \sigma\left(W_{x\mathrm{h}\_f} x_t + W_{\mathrm{hh}\_f} \mathrm{h}_{t-1} + b_{\mathrm{h}_f}\right) \qquad (2)$$

$$\text{Input gate}: \ i_t = \sigma(W_{x\mathrm{h}\_i} x_t + W_{\mathrm{hh}\_i} \mathrm{h}_{t-1} + b_{\mathrm{h}\_i}) \qquad (3)$$

LSTM 을 사용함으로써 얻을 수 있는 또 다른 장점은 Sequence to Sequence Model 의 Input Data 차원이 고정되어 있다는 문제점을 해결할 수 있다는 것이다. 다양한 Input Data 크기를 받아들이기 위하여 고정된 큰 크기의 입력 벡터를 사용한다.

2.2.4. Attention(주의 기구)

인간이 문장을 이해하는 특징을 살펴보게 되면 문장을 읽을 때 모든 단어의 의미를 천천히 기억해가며 읽지 않는다. 문장 중에서도 중요한 단어들만 더 중요도를 가지고 이해한다. 기계번역의 경우 입력 문장이 매우 긴 경우 번역 문장의 앞쪽 단어들을 번역하기 위해 입력 문장 중 앞 부분에 해당하는 단어들이 더 중요하게 사용될 것이고, 마찬가지로 뒤쪽 몇 단어들을 번역하기 위해 입력 문장의 뒷 부분에 해당하는 단어를 중요하게 볼 것이다. 이밖에도 텍스트는 여러가지 이유로 단어나 문장들의 중요도를 결정하는 방법이 복잡하다. 단순히 Sequence to Sequence 에 만으로 학습을 하면 Source Language 벡터의 길이가 긴 경우, Source Language 벡터 모두를 똑같이 참조하게 되고 정확도가 떨어지게 된다.

이를 해결하기 위한 방법으로 고안된 것이 Sequence to Sequence 로 하여금 중요한 부분만 주의(Attention)하게 만드는 기법이 Attention 기법 이다. Attention 기법은 Source Language 의 Hidden state 를 참조하여 새로운 Context 벡터를 만들어 특정벡터에 가중치를 더 줄 수 있다.그림 8 에서 Attention 의 구조를

살펴볼 수 있다. 기존의 모델들과는 다르게 Attention network 가 추가된 것을 확인할 수 있고 이를통해 특정 부분에 중요도를 높여주는 효과를 얻을 수 있다.

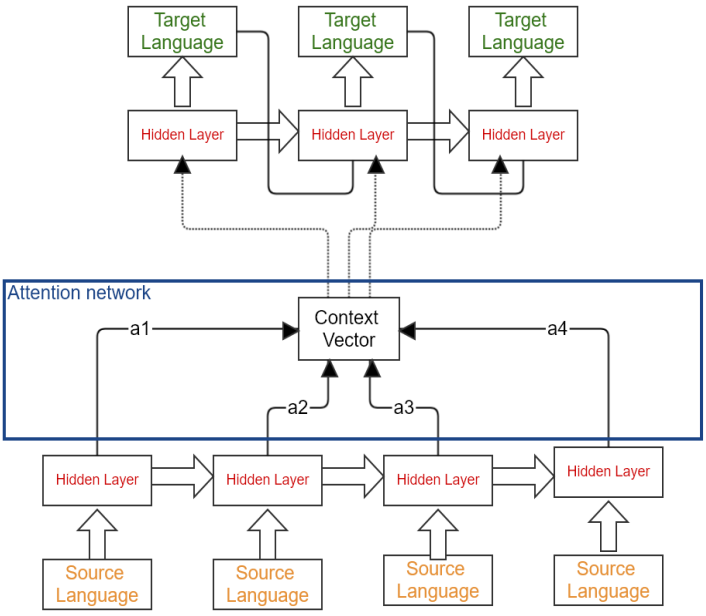


그림 8. Attention

3. 추출형 문서요약 –TextRank

TextRank 알고리즘은 2004 년 Mihalcea 에 의해 발표된 graph-based ranking model 기반의 추출형 문서요약 방법이다. 본 절에서는 TextRank 의 구현방안과 결과에 대해 설명한다. [11]

3.1. Text Rank 구현 방안

TextRank 는그림 9 에 나타난 과정으로 추출형 문서요약문을 출력한다.

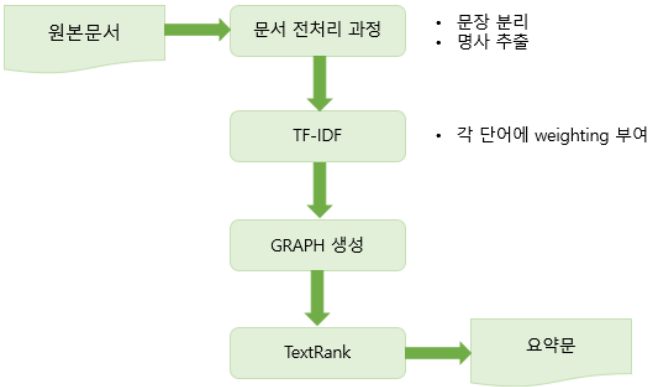


그림 9. TextRank 구성

1) 문서 전처리과정

TF-IDF 로 단어들의 가중치를 주는 것을 구현 하기 위해 먼저 입력이 되는 원본문서를 문장 단위로 분리하고 여기서 각각의 명사를 추출하는 전처리 과정을 수행한다. 여기서는 KoNLPy 이용한다. KoNLPy 는 한국어 정보처리를 위한 Python 패키지로서 품사 Tagging 을 이용해 명사만 분류해 낼 수 있다.

2) TF-IDF

2.1.2 절 TF-IDF 에서 언급한 TF-IDF 를 이용해서 각각의 단어의 가중치를 구한다. 가중치가 높으면 중요한 단어로 판단한다.

$$q(w) = f_d(w) * log \frac{|D|}{f_d(w)} \quad (4)$$

3) Text Rank 적용.

$$\boldsymbol{TR(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum V_k \in Out(V_j) w_{jk}} TR(V_j)} \quad (5)$$

위는 TextRank 식을 나타낸 것이다. TR 을 Return 하여 값이 큰 순으로 중요한 문장으로 판별한다. $TR(V_i)$ 는 단어에 대한 TextRank 값을 의미한다. w는 단어의 가중치이다. 여기서의 가중치는 사용자가 정해진 값을 그대로 사용하게 된다. d 는 0.85 라는 고정된 상수를 사용하였다. 위 식의 계산결과에 높은 순으로 정렬한다는 것이다. 즉, 사람이 글을 논리적으로 이해하고 중요한 부분을 인지하는 것과는달리 참조가 많이 된 것(참조가 많이 되면 중요한 문장이다 라는 아이디어로 알고리즘을 구현한 것이다.)을 순서대로 보여주는 것이다.

3.2. TextRank 결과 및 한계

TextRank 을 사용하여 요약문을 생성한 뒤 결과를 확인하였다. 그림 10 과 그림 11 에서 각각 실험에 사용된 원본문서의 일부와 TextRank 수행결과를 보였다. 비교해보면 원본문서에서 중요하다고 판단되는 문장들을 그대로 가져와서 요약문장을 추출한 것을 알수 있다. 이들 문장을 살펴보면 문서 내에서 가중치가 높은 단어들인 '소득', '공제', '카드' 등의 단어가문장 내에 비교적 많이 포함 돼있음을 확인할 수 있다.

◇연봉 5000만원대가 84%로 카드 공제 가장 많이 받아

신용카드 소득공제를 받으려면 총소득의 4분의 1 이상을 카드로 써야 한다. 4분의 1 문턱에서 1원이라도 모자라면 한 푼도 공제받을 수 없다. 수억원대의 고소득자일수록 카드로 소득의 4분의 1 이상을 쓰기 쉽지 않기 때문에 소득이 높을수록 공제를 못 받는 경우가 늘어난다. 2015년 기준으로 근로소득세를 내는 사람 중에서 카드 소득공제를 받는 비율이 가장 많은 소득 구간은 연봉 5000만~6000만원이었다. 전체의 84%가 혜택을 봤다. 연봉 6000만~8000만원 사이에서도 83%, 8000만~1억원 구간은 79%가 공제를 받았다.

반면 1억~2억원 사이에서는 공제받은 사람 비율이 68%로 줄었고, 2억~3억원인 사람은 31%만 카드 공제를 받은 것으로 나타났다. 국세청 관계자는 "월급쟁이가 카드를 쓰는 용도가 다들 엇비슷해서 고소득자라고 해서 카드 사용액이 월등하게 늘어나지는 않기 때문에 소득이 높을수록 '4분의 1 문턱'을 넘기가 어려워지는 경향이 있다"고 말했다. 또 고소득자들이 외국에 나가 카드를 많이 사용하지만, 해외에서의 카드 사용액을 일절 공제해 주지 않는 점도 영향이 있는 것으로 분석된다.

◇소득 낮은 부부는 한 사람 명의의 카드 써야 유리

소득이 연 5000만원에 못 미치는 서민층에서도 카드 공제를 받는 사람들의 비율이 낮아지는 경향을 보인다. 연봉 4000만원대는 81%, 3000만원대는 73%, 2000만원대는 55%만 카드 소득공제를 받은 것으로 나타났다.

이런 현상이 나타나는 이유는 우선 저소득층일수록 허리띠를 졸라매는 경향이 있어 '4분의 1 문턱'을 넘지 못하는 경우가 상당하기 때문인 것으로 추정된다. 또 소득이 넉넉하지 않은 맞벌이의 경우 한 사람 명의로 카드를 발급받아 한 사람의 카드 사용액이 '4분의 1 문턱'을 넘도록 몰아주고, 나머지 배우자는 카드 소득공제를 포기하는 절세 노하우를 보여주는 사례도 많다고 국세청 관계자들은 설명했다. 서울지역의 한 세무사는 "부부가 각

그림 10. 사용한 원본문서 일부

출처:http://v.media.daum.net/v/20170611192209012?rcmd=r

연봉 5000 만원 대가 84%로 카드 공제 가장 많이 받아 신용카드 소득 공제를 받으려면 총소득의 4 분의 이상을 카드로 써야한다.
2015 년 기준으로 근로 소득세를 내는 사람 중에서 카드 소득공제를 받는 비율이 가장 많은 소득 구간은 연봉 5000 만 ~6000 만원이었다.
연봉 4000 만원대는 81%, 3000 만원대는 73%, 2000 만원대는 55%만 가드 소득 공제를 받은 것으로 나타났다.
Keywords: ['소득', '공제', '카드', '사람', '사용', '한도', '만원', '경우', '문턱', '신용카드']

그림 11. Text Rank 결과

이 절에서 추출형 문서요약 방법중에서 TextRank 알고리즘을 수행하고 그 결과를 보였다. 결과에서 확인할 수 있듯이 문서내의 중요부분을 추출하여 요약문을 양호한 품질의 잘 작성하였다. 하지만 인간이 긴 글을 읽을 때에는 이 방법과 같이 잦은 빈도로 출현하는 단어도 중요하게 보지만 전후 문맥과 단어의 의미를 상황에 맞게 이해하면서 글을 읽는다. 이러한 점에서 단순히 빈도 중심의 추출 방법으로는 인간에게 읽히기 쉽고 정확한 요약문을 얻기에는 아직 부족함을 확인 할 수 있다. 결국 문서의 요약은 요약문 내의 문장들끼리 서로 연관성이 높아야하고 의미, 개념적 연결성이 높은 글이 되어야 한다. 따라서 추출 정보를 기반으로 하여 빈도수만을 고려하지 않고, 문장 간의 의미적 연결성을 높인 다중 문서요약 기법이 필요하다.

4. 생성형 문서요약 – 영어 문서요약

4.1. 텐서플로우를 이용한 구글 문서요약

Google Text summarization 은 Tensorflow 를 이용한 Sequence to sequence model 을 적용하였으며 뉴스기사에서 제목을 생성하는 생성형 문서요약을 수행하였다. 그림 12 에서 입력으로 뉴스기사를 넣고 제목을 생성하였을 때 생성된 제목의 예를 나타내었다.

Input: Article 1st sentence	Model-written headline
metro-goldwyn-mayer reported a third-quarter net loss of dlrs 16 million due mainly to the effect of accounting rules adopted this year	mgm reports 16 million net loss on higher revenue
starting from july 1, the island province of hainan in southern china will implement strict market access control on all incoming livestock and animal products to prevent the possible spread of epidemic diseases	hainan to curb spread of diseases
australian wine exports hit a record 52.1 million liters worth 260 million dollars (143 million us) in september, the government statistics office reported on monday	australian wine exports hit record high in september

그림 12. Google Text Summarization Example

출처:https://ai.googleblog.com/2016/08/text-summarization-with-tensorflow.html

4.2. 훈련 데이터

Google 은 LDC(Linguistic Data Consortium)의 Annotated English Gigaword 라는 Data 를 활용했다. 해당 Dataset 은 약 천만개의 document 의 뉴스기사 자료이다. 입력 데이터로 뉴스기사를 활용해 Training 을 하는 이유는 다른 문서자료보다 뉴스기사의 문법이 비교적 정확하고 내용의 다양성을 확보할 수 있기 때문에 신문기사를 Training data 로 사용한다. 따라서 기사의 본문을 원본문서로 기사의 제목을 요약문으로 학습시킨다. 본 논문에서는 한국어 문서요약을 목표로 하기 때문에 대량의 한국어 기사 데이터를 확보하는 것이 과제로 될 것이다. 그림 13 에서 보인 바와 같이 뉴스의 기사 본문 중 일부 부분과 기사의 제목 부분을 따로 추출하여 data set 을 만들어서 활용한다.

ENCODE(뉴스 기사의 본문)	DECODE(뉴스 기사의 제목)
US President Donald Trump on Wednesday accused the Organization of Petroleum Exporting Countries of driving up oil prices, in a fresh swipe at the cartel's agreement to cap production. "Oil prices are too high, OPEC is at it again. Not good!" he wrote on Twitter.	Trump accuses OPEC of driving up oil prices.

그림 13. Input Data 형식

4.3. Vocab File

학습데이터에서 사용된 단어들의 빈도수가 저장된 파일이다. 요약문 생성시 이 파일에서 단어를 참조하여 요약문을 생성한다. 따라서 Vocab file 의 단어 수가 충분히 없으면 만족스러운 요약문을 생성하지 못한다. Google 에서 수행한 영어의 경우 Vocab file 에서 참조하는 단어의 개수를 200K 개 로하여 훈련시켰다. 그림 14 는 ToyData 의 vocal file 을 나타내었다.

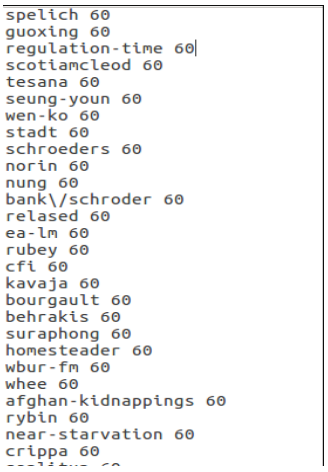


그림 14. Google Vocab File example

4.4. Example 수행 결과

Google 에서 제공한 example 을 수행하고 그림 15 에 출력 결과를 보였다. 보이는 바와 같이 to, as, of 와 같은 전치사만 제대로 출력되고 그밖의 단어들은 제대로 출력하지 못하였음을 확인할 수 있다. 이와같은 만족스럽지 못한 결과를 얻은 이유를 분석해보면 제공된 training data 의 수가 충분하지 않고 Vocab file 의 데이터가 실제 학습 데이터에서 사용되지 않은 부적절한 단어가 다수 포함되어 있는 것을 확인하였다. 따라서 우선 구글이 제시한 바와같이 20 만 개의 Vocabulary size 를 확보하는 것을 목표로 하였다.

```
output=<UNK> <UNK> <UNK> <UNK> as <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> as <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> .
output=to to <UNK> <UNK> <UNK> <UNK> to <UNK> .
output=<UNK> <UNK> <UNK> <UNK> of <UNK> <UNK> from <UNK> <UNK> .
output=<UNK> <UNK> <UNK> to <UNK> for <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> <UNK> for <UNK> .
output=<UNK> <UNK> to <UNK> <UNK> for <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> for <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> to <UNK> for <UNK> <UNK> .
output=<UNK> <UNK> <UNK> for <UNK> <UNK> .
output=<UNK> of <UNK> to be <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> to <UNK> for <UNK> <UNK> <UNK> .
output=<UNK> for <UNK> .
output=<UNK> <UNK> <UNK> <UNK> of <UNK> <UNK> from <UNK> <UNK> .
output=to to <UNK> <UNK> <UNK> <UNK> to <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> .
output=<UNK> of <UNK> to be <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> of <UNK> <UNK> from <UNK> <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> for <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> <UNK> for <UNK> .
output=<UNK> <UNK> <UNK> <UNK> .
output=<UNK> for <UNK> .
output=<UNK> for <UNK> .
output=<UNK> <UNK> <UNK> <UNK> <UNK> <UNK> <UNK> .
output=<UNK> <UNK> <UNK> to <UNK> for <UNK> <UNK> <UNK> .
output=<UNK> <UNK> in <UNK> <UNK> .
output=<UNK> <UNK> <UNK> to <UNK> for <UNK> <UNK> <UNK> .
output=<UNK> for <UNK> .
```

그림 15. Example data 출력 결과

4.5. 학습

Google Text Summarization 은 Vocab File 과 입력 데이터 (뉴스) 으로 훈련 과정을 실시하게 된다. 훈련 과정은 정답 요약문과 생성된 요약문의 차이인 loss 를 줄이는 방향으로 수행된다. Google Text Summarization 의 running_avg_loss 는 최대 12 에서 최소 0 으로 설정되어 있고 학습을 거듭하면서 수치는 낮아지게 된다. 머신러닝에서 loss(손실)이란 관측값(실제값)과 예측값(출력값)의 차를 말하는데 loss 가 작을수록 정확한 결과를 출력할 수 있다. 그림 16 에서는 Google Text Summarization 모델의 학습과정을 나타내었다. 보이는바와 같이 전체 데이터의 평균 손실을 낮추는 방향으로 학습이 진행된다.

```
running_avg_loss: 4.920805
running_avg_loss: 3.598699
running_avg_loss: 3.326588
running_avg_loss: 3.283831
running_avg_loss: 2.074936
running_avg_loss: 2.487434
running_avg_loss: 1.389228
running_avg_loss: 1.018900
running_avg_loss: 1.813420
running_avg_loss: 0.765589
running_avg_loss: 1.021877
running_avg_loss: 1.254545
running_avg_loss: 1.096022
running_avg_loss: 0.909049
running_avg_loss: 0.843910
running_avg_loss: 0.908698
running_avg_loss: 0.947617
running_avg_loss: 1.916891
running_avg_loss: 0.825155
running_avg_loss: 1.688306
running_avg_loss: 1.017905
```

그림 16. Running_avg_loss

5. 생성형 문서요약 – 한국어 문서요약

본 논문에서 진행하는 생성형 한국어 문서요약 모델의 구성은 그림 17 과 같다.

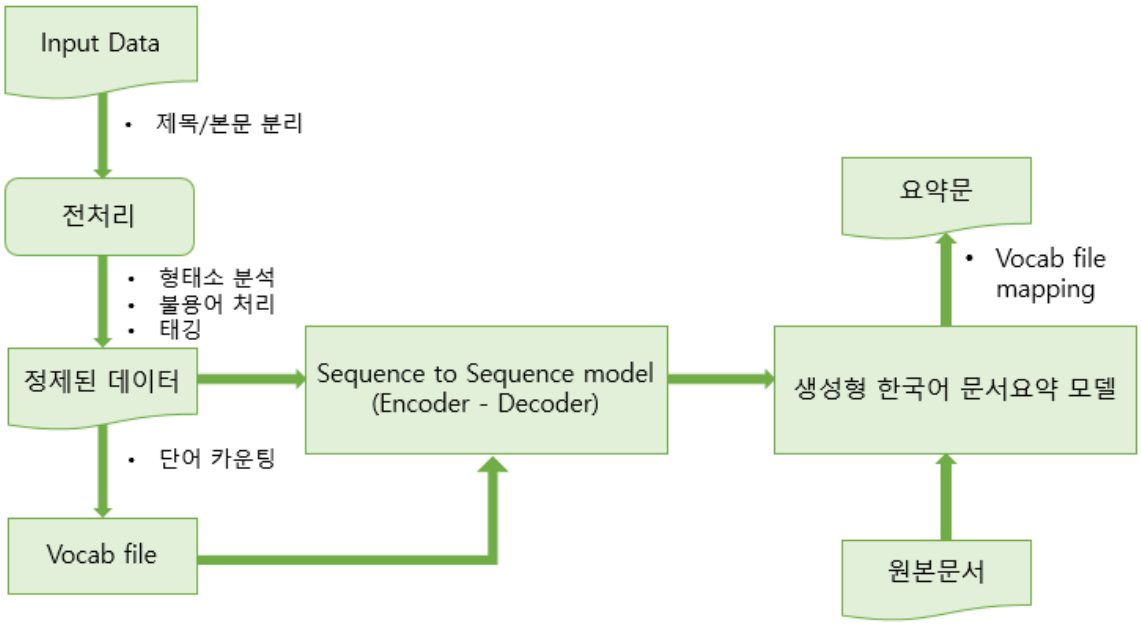


그림 17. 시스템 구성도

5.1. 데이터 수집 과정

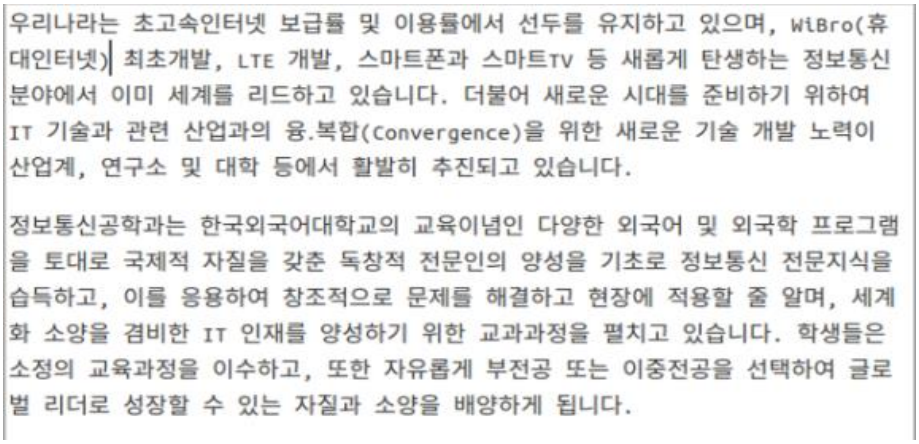
뉴스 기사를 크롤링하여 자료를 수집하는 과정에서 저작권의 문제와 양질의 데이터를 구하는데 어려움이 있었다. 따라서 Google 에서 활용한 Annotated English Gigaword 와 유사한 Dataset 을 확보하기 위하여 한국언론진흥재단에서 제공하는 뉴스 빅데이터 분석 시스템인 빅카인즈를 활용 하여 Dataset 을 확보 하였다.빅카인즈는 42 개 언론사, 약 3000 만 건의 기사 데이터를 보유하고 검색, 분석 서비스를 제공 한다. 한국어 문서요약 Model 을 생성하기 위한 Input Dataset 으로 약 7 년치의 뉴스기사 100 만건을 수집하여 훈련 데이터로 활용하였다. 그림 18 은 빅카인즈 에서 제공한 뉴스 Dataset 의 형태이다.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
뉴스 식별	일자	언론사	기고자	제목	통합 분류1	통합 분류2	통합 분류3	사건/사고	사건/사고	사건/사고	인물	위치	기관	키워드	특성추출	본문	URL	분석제외 여부				
08200101..	20180418	OBS	김용재	인천시, 1천	지역>전북	지역>충남	지역>광주					인천시	인천시	인천시,1천	1천,문화공	인천시가 :	http://www.obsnews.co.kr/news/articleView.html?idxno=1092316					
08200101..	20180306	OBS	최진광	구리시, '어	지역>경기	지역>전북	지역>광주				백경현	볼라드,구	인창초교,	구리시,통	통학로,경	구리시가 :	http://www.obsnews.co.kr/news/articleView.html?idxno=1085959					
08200101..	20180309	OBS	김창문	이태운 전	지역>대구	지역>경기	지역>광주				이태운,전	서울	대전고등법	이태운,서	이사장,서	서울고등법	http://www.obsnews.co.kr/news/articleView.html?idxno=1086408					
08200101..	20180220	OBS	유재명	경기도내 '	지역>경기	지역>울산	지역>전북					경기도내	경기도보건	경기도내,	(기준치,여	경기도내 '	http://www.obsnews.co.kr/news/articleView.html?idxno=1083936					
01500901..	20180308	울산매일	강태아 기	울산시 '노	지역>울산	지역>광주	지역>전북				김기현	남구	울산,간담	울산시,노	노동복지센	노동복지회	http://www.iusm.co.kr/news/articleView.html?idxno=790431					
01500901..	20180401	울산매일	우성만 기	벚꽃 상준	지역>울산	지역>대구	지역>광주					삼남면,울		벚꽃,상준	상준객,울	주말인 지	http://www.iusm.co.kr/news/articleView.html?idxno=793810					
08200101..	20180311	OBS	고영규	용인시, 통	지역>경기	지역>울산	지역>충남					용인시,경	용인시	용인시,통	통학버스,	경기도 용	http://www.obsnews.co.kr/news/articleView.html?idxno=1086610					

그림 18. 뉴스 Dataset

5.2. KoNLPy

뉴스 Data 를 가지고 Input Data 나 vocab file 을 만들기 위해 KoNLPy 를 사용하여 Data 를 한번 더 이번 연구에서 필요로 하는 Dataset 의 형태로 변환 하였다. 즉 형태소 분리 과정을 거쳤다. 이렇게 Dataset 을 변형시켜야 하는 이유는 아래와 같다. 그림 19 는 입력 데이터인 여러 문장으로 이루어진 문서의 예시이다. 그림 20 은 KoNLPy 를 거치지 않은 Output Data 로서 단순히 띄어쓰기 대로 문장을 분류한 결과이다.

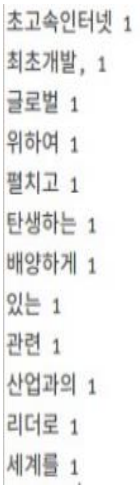


우리나라는 초고속인터넷 보급률 및 이용률에서 선두를 유지하고 있으며, WiBro(휴대인터넷) 최초개발, LTE 개발, 스마트폰과 스마트TV 등 새롭게 탄생하는 정보통신 분야에서 이미 세계를 리드하고 있습니다. 더불어 새로운 시대를 준비하기 위하여 IT 기술과 관련 산업과의 융.복합(Convergence)을 위한 새로운 기술 개발 노력이 산업계, 연구소 및 대학 등에서 활발히 추진되고 있습니다.

정보통신공학과는 한국외국어대학교의 교육이념인 다양한 외국어 및 외국학 프로그램을 토대로 국제적 자질을 갖춘 독창적 전문인의 양성을 기초로 정보통신 전문지식을 습득하고, 이를 응용하여 창조적으로 문제를 해결하고 현장에 적용할 줄 알며, 세계화 소양을 겸비한 IT 인재를 양성하기 위한 교과과정을 펼치고 있습니다. 학생들은 소정의 교육과정을 이수하고, 또한 자유롭게 부전공 또는 이중전공을 선택하여 글로벌 리더로 성장할 수 있는 자질과 소양을 배양하게 됩니다.

그림 19. 입력 데이터 예시

출처: http://ice.hufs.ac.kr/w/index.php/%EA%B5%90%EC%9C%A1%EB%AA%A9%ED%91%9C



초고속인터넷 1
최초개발, 1
글로벌 1
위하여 1
펼치고 1
탄생하는 1
배양하게 1
있는 1
관련 1
산업과의 1
리더로 1
세계를 1

그림 20. Output Data No_Konlpy

문서는 문장들로 이루어지고 문장은 단어와 그 밖의 구성요소들로 이루어진다. 단어는 형태소들로 이루어진다. 문서를 가장 작은 단위로 나누면 형태소로 구성된다. 초고속인터넷이란 단어만 보더라도 초고속과 인터넷이 분리되지 않고 합쳐져 있다. 인간은 초고속과 인터넷을 다른 두 단어가 결합한 단어임을 인지하듯이 신경망도 초고속인터넷을 초고속과 인터넷의 결합된 단어로 인식하게 하기 위하여 문장을 최소단위인 형태소로 분리하여

Vocab file 을 작성하고 이로인해 효율적인 학습과 성능향상이 있을 것으로 기대하였다. 그림 21 에서 위의 그림 19 에서 보인 문서를 형태소 분석한 결과를 보였다. 결과를 보면 초고속인터넷 뿐만 아니라 조사 등 의미를 갖는 최소단위인 형태소로 분리되었음을 확인할 수 있다. KoNLPy 를 통해 형태소 단위로 분류한 Data 에서 단어의 출현 빈도수를 Count 하여 출현 빈도수가 높은 순으로 정렬하였다.

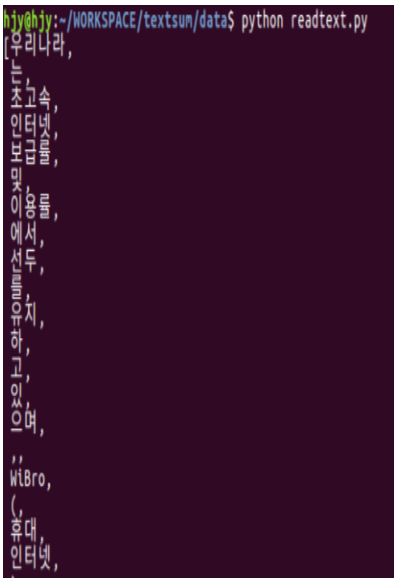


그림 21. 형태소 분석

그림 22 에서 보인 음절 단위 형식의 모델 별 문서요약 기능과 그림 23 에서 보인 형태소 단위 형식의 모델별 문서요약 기능은 참고문헌[4]에서 참조 하였다. 그림 22, 23 에서 나타내었듯이 입력데이터를 분리한 형태에 따라 성능 차이가 발생한다. 대부분의 한국어 문서요약 및 자연어처리 분야에서 다음과 같은 성능 향상을 위해 입력 데이터의 형식을 형태소 단위를 넣는 것이 우수하다는 것이 증명되어 있다. 따라서 본 논문에서 수행하는 생성형 문서요약 또한 형태소 단위로 분리하고 수행하는 것이 우수한 결과를 보일 것으로 가정하였다..

	h	ROUGE-1	ROUGE-2	ROUGE-L
GRU search	500	26.61	8.97	21.6
GRU search+ Input feeding	500	20.72	5.32	16.7
GRU search+ Input feeding + Copy	500	22.92	6.74	18.68

그림 22. 음절 단위 형식의 모델 별 문서요약 기능

출처: http://s-space.snu.ac.kr/handle/10371/141455

	h	ROUGE-1	ROUGE-2	ROUGE-L
GRU search	300	29.9	10.67	24.71
GRU search+ Input feeding	300	29.55	10.52	24.25
GRU search+ Input feeding + Copy	300	35.92	15.37	29.45

그림 23. 형태소 단위 형식의 모델 별 문서요약 기능

출처: http://s-space.snu.ac.kr/handle/10371/141455

KoNLPy 에서는 형태소 분리를 지원하는 여러 tag package 가 존재한다. 그림 24 예시는각각의 Version 에 따라서 얻어지는 결과의 예시를 보인다. 형태소 분류에 사용된 문장은 " 아버지가방에들어가신다" 이다.

Hannanum	Kkma	Komoran	Mecab	Twitter
아버지가방에 들어가 / N	아버지 / NNG	아버지가방에 들어가신다 / NNP	아버지 / NNG	아버지 / Noun
이 / J	가방 / NNG		가 / JKS	가방 / Noun
시ㄴ다 / E	에 / JKM		방 / NNG	에 / Josa
	들어가 / VV		에 / JKB	들어가신 / Verb
	시 / EPH		들어가 / VV	다 / Eomi
	ㄴ다 / EFN		신다 / EP+EC	

그림 24. KoNLPy 수행 결과 예시

한국어의 조사까지 분리해주는 Tag package 는 Kkma, Mecab Twitter 3 가지이다. 이 중 Mecab 은 처음 일본어 분석으로 나온 Model 로서 한국어를 Mapping 하는 Dictionary 가 부족하다. Kkma 의 경우에는 정확도가 매우 높으나 수행 시간이 오래 걸린다는 단점이 있다. Twitter 분류기는 성능은 Kkma 보다 떨어지나 수행 시간이 좀 더 빠르다. 백만개의 Input Data 의 처리과정을 거쳐야 되는 본 논문에서는 최종적으로 Twitter 분류기를 활용해 형태소 분리를 수행하기로 했다.

5.3. Vocab file 생성 과정

빅카인즈에서 제공하는 Dataset 은 Excel 파일 형식으로 제공되며 "뉴스 식별자, 일자, 언론사, 기고자, 제목, 통합 분류 등 여러 정보를 담고있어 빅데이터를 활용한 연구 및 자연언어처리 연구 등 다양한 분야에서 양질의 데이터를 제공한다. 본 논문에서는 기사의 제목, 본문만 필요하므로 제목과 기사 본문만 따로 정제하는 과정을 거쳐 사용하였다. KoNLPy 를 통한 텍스트의 형태소 분류는 한 번에 최대 2 만개의 기사 정도였다.따라서,백만개의 기사를 2 만개의 크기로 나누어 형태소 분류를 수행하였고 Data 의 단어의 빈도수에 따라 우선순위로

20 만개의 단어정보를 갖는 Vocab file 을 생성하였다. 20 만개의 단어로 이루어진 Vocab file 의 가장 적은 Count 는 1 의 값을 가진다.즉 사용빈도가 매우 적은 단어까지 Vocab file 에 저장 해주었으며 이를통해 단어의 다양성까지 확보하여 4.4 절의 그림 15 에서 보인 바와 같이 요약 생성시 참조할 단어를 찾지못하는 문제를 예방토록 하였다. 이를 통해 충분히 많은 단어의 다양성을 확보하였다고 가정하였다.

그림 25 는 Vocab file 을 예시를 보였다. 좌측은 빈도 수가 매우 높은 단어들이고 우측은 빈도 수가 매우 적은 단어들이다. 조사와 같은 형태소는 매우 빈도 수가 높게 계산되었고 우측에 보이는 것과 같이 값이 매우 적은 형태소들은 우리가 일상생활에서 거의 쓰지 않는 단어들이지만 만족할만한 요약문 생성을 위하여 추가 해주었다.

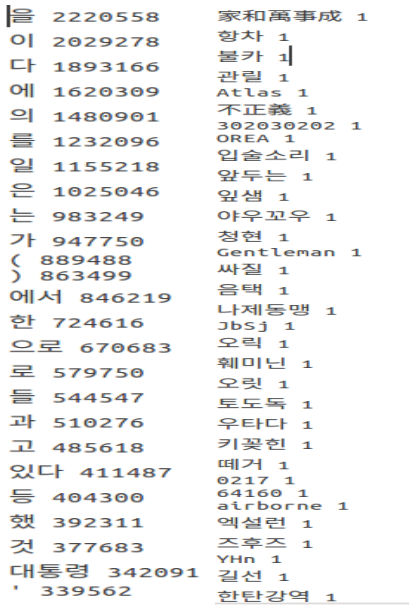


그림 25. Vocab file 단어의 일부

5.4. Input Data 형태

Input Data 는 기사의 제목과 본문으로 두 부분으로 구분하였다.본문은다시 문서, 단락, 문장별로 TAG 를 붙여 구분하였다. 표 1 에서 각 TAG 의 의미를 정의하였고, 그림 26 에서는 InputData 의 실제 수행에서 사용한 학습 data set 의 입력 형태를 나타내었다.

TAG	의미
abstract	기사의 제목
article	기사의 본문
<d> ~ </d>	문서

~	구 또는 절
<s> ~ </s>	문장

표 1. 각 TAG 의 의미

```
abstract=<d> <p> <s> 일제 징용 피해자 ' 일 기업 손해배상 ' </s> </p> </d>  article=<d> <p> <s> 일제 강제 징용 피해자 들 이 일본 미쓰비시 중공업 과 신일본제철 을 상대로 강제노동 에 대한 손해배상 과 못 받은 임금 을 청구 한 소송 에 대해 25 일 대법원 판결 이 내려진 다 . </s> <s> 일본 최고재판소 가 피해자 들 에게 패소 판결 을 내린 사건 에 한국 대법원 이 정반대 의 판결 을 내릴 경우 과거 사 문제 와 관련해 양국 에 큰 영향 을 미칠 것 으로 예상된 다 . </s> </p> </d>
```

그림 26. Input Data 형식

Input data 를 태그를 붙여 부분 별로 구분한 후 이진파일로 변환하여 Training 을 수행한다. 그림 27 에서 이진파일 변환 과정을 보였다.

```
alds@alds-ubt:~/Desktop/WORKSPACE/data$ ls
data_convert_example.py  example_txt  text
alds@alds-ubt:~/Desktop/WORKSPACE/data$ python data_convert_example.py --command text_to_binary --in_file example_txt --out_file example_bin
alds@alds-ubt:~/Desktop/WORKSPACE/data$ ls
data_convert_example.py  example_bin  example_txt  text
```

그림 27. 이진파일 변환 과정

5.5. 훈련 과정

그림 28 에서 훈련 과정을 실제 수행하며 그 과정을 그래프로 나타내었다. 그래프의 x 축은 학습 횟수, y 축은 Error 율을 의미한다. 약 10 만 번의 학습을 수행하였고 대략적인 평균 손실은 1.3 대에 이르렀다.

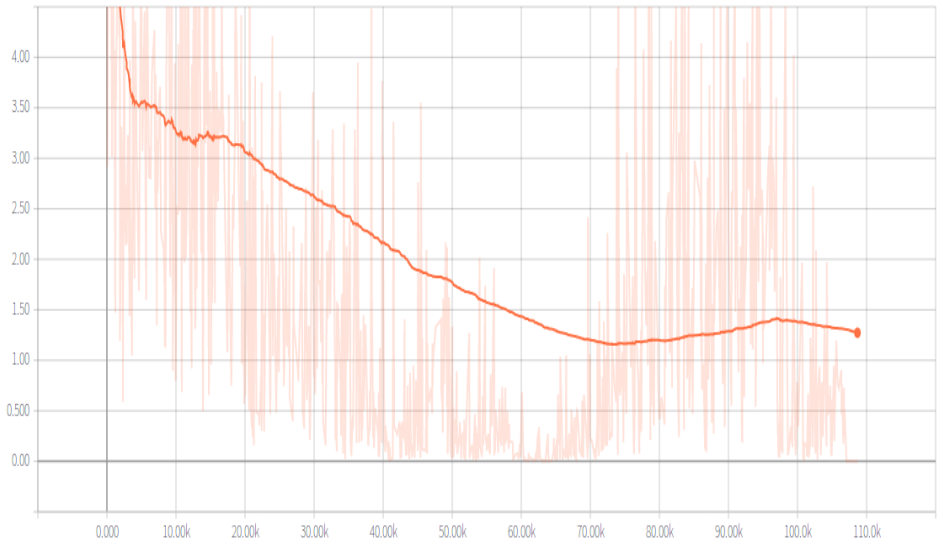


그림 28. 훈련 과정

5.6. Decode 결과

그림 29 를 보면 실제 기사 원문과 비슷하게 Model 의 결과 값이 나오는 것을 살펴 볼 수 있다. 몇 가지 특징을 살펴보면, 불용어 처리 과정에서 없앤 '.' 기호는 Vocab File 에서 찾아 볼 수 없어 <UNK>로 나오게 되었다. 또한 특정 고유명사를 가리키는 아베, 북한, 여수는 Decode 결과 값에서 나오지 않게 되었다. Vocab file 에 단어는 있지만, 낮은 Count 의 값을 가지고 있으므로 나오지않은 것으로 추측하였다. 또 한가지 확인할 수 있는 특이점은 문장 내에서 단어의 중복 사용이 많다는 것이다.

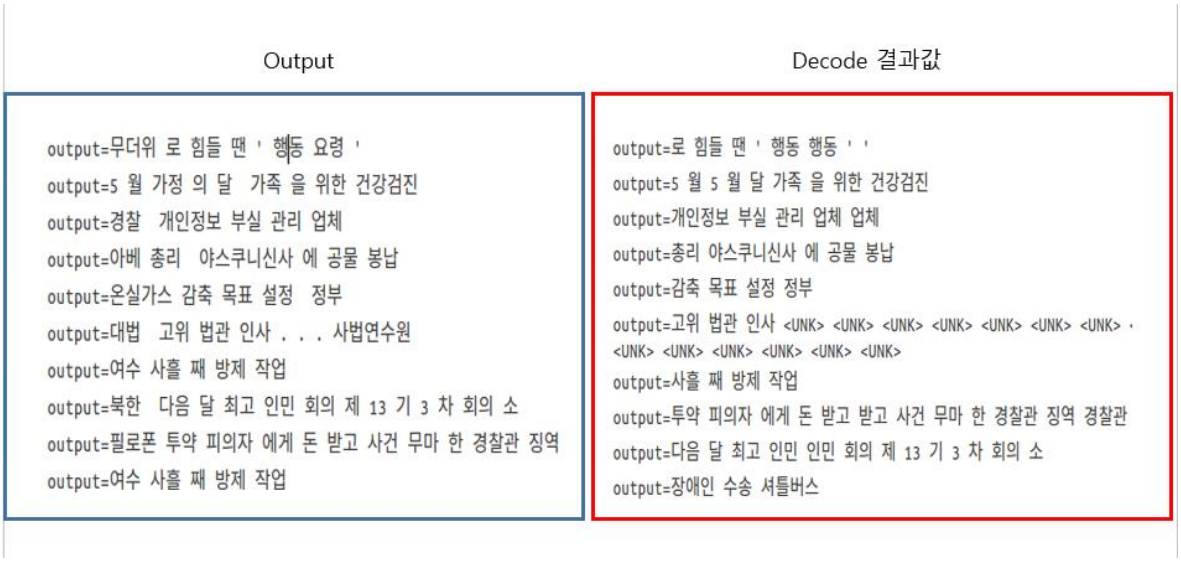


그림 29. Trainning Decode 결과값비교

6. 성능 평가 비교

6.1. Rouge-N

요약문의 성능평가는 시스템 번역문과 정답 번역문의 일치여부가 중요한 기계번역과 다르게 시스템 요약문의 정답에서의 정보누락 여부가 중요하다. 이러한 이유로 현재 문서요약 연구에서는 ROUGE(Recall-Oriented Understudy for Gisting Evaluation)라는 성능지표를 주로 사용하고 있다. ROUGE 는 정답 요약문(reference summary)과 시스템 약문(candidate summary)간의 재현율을 바탕으로 평가한다. 가장 많이 사용하는 Rouge-N 은 밑의 식 (6) 으로 나타낼 수 있다.

$$\frac{\sum_{S\in\{ReferenceSummaries\}}\sum_{gram_n\in S} Count_{matc\boxtimes}(gram_n)}{\sum_{S\in\{ReferenceSummaries\}}\sum_{gram_n\in S} Count(gram_n)} \quad (6)$$

즉 미리 정답 요약문인$ReferenceSummaries$의 $n-gram$ 들 중의 하나인 $gram_n$이 시스템 요약문으로 만들어진 $n-gram$에 포함되어 있으면 1, 그렇지 안으면 0 을 반환한다. $n-gram$이란 n 개의 문자열 크기만큼의 window 를 만들어 문자열을 왼쪽에서 오른쪽으로 한 단위씩 움직이며 추출되는 시퀀스의 집합의 출현 빈도수를 기록하는 것 이다. 그 다음으로 많이 사용하는 ROUGE-L 은 밑의 수식 (7), (8), (9)으로 나타낼 수 있다.

$$R_{lcs} = \frac{LCS(X,Y)}{m} \quad (7)$$

$$P_{lcs} = \frac{LCS(X,Y)}{n} \quad (8)$$

$$F_{lcs} = \frac{2R_{lcs}P_{lcs}}{R_{lcs}+P_{lcs}} \quad (9)$$

X 는 KoNLPy 를 거쳐서 나온 Model 의 요약문을 의미하고 Y 는 KoNLPy 를 거쳐서 나온 정답 요약문을 의미한다. LCS(X,Y)는 X 와, Y 의 최장 공통 부분 수열(연속으로 일치하는 최장의 형태소 배열)의 길이를 뜻하고, m 은 정답 요약문의 길이, n 은 Model 의 요약문의 길이를 뜻하게 된다.

6.2. 한국어 문서요약 성능 평가– 훈련 데이터

표 2 는 훈련 데이터에 대한 Rouge-N 과 Rouge-L 을 통해 나온 결과 값이다. 훈련 데이터 중 2000 개를 사용하여 성능 평가를 실시 하였다.

	Rouge-1	Rouge-2	Rouge-L
한국어 문서요약	83.56	66.26	83.56

표 2. 한국어 문서요약 성능 평가–훈련 데이터

성능 평가 결과 대략 75% 이상의 훈련 데이터에대한 성능은 잘 나온 것을 확인 할 수 있다.

6.3. 한국어 문서요약 성능 평가 – Test Data

그림 31 는 Test Decode 값 과 실제 결과를 비교한 문장 아래 Rouge 기법을 활용하여 구한 실제 값을 적어두었다.

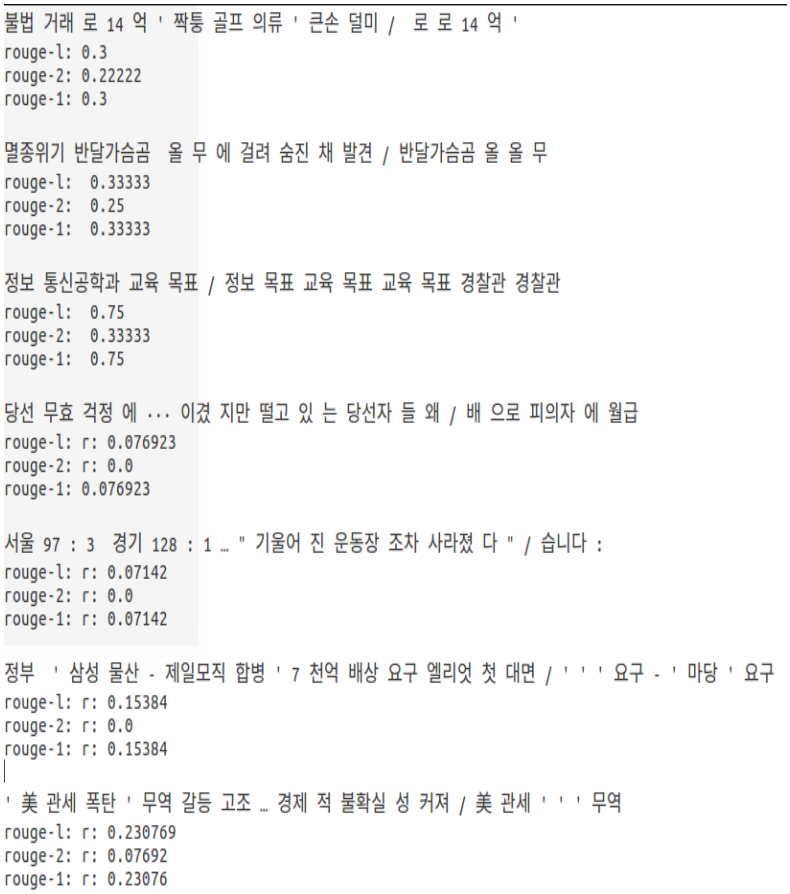


그림 30. Test Decode 결과 및 Rouge 성능 평가

표 3 에서는 Test Data 에 대한 Rouge-N 과 Rouge-L 을 통해 Test Data 중 2000 개를 평균하여 나온 결과 값이다.

	Rouge-1	Rouge-2	Rouge-L
한국어 문서요약	19.35	12.59	19.43

표 3. 한국어 문서요약 성능 평가 – 시험 데이터

훈련 데이터결과보다 시험 데이터의 품질이 확실히 떨어진다. 그 원인은 다양하겠지만 뉴스 자체의 특성 때문이다. 같은 사건을 보도하더라도 완전히 같은 내용의 기사는 없으며, 그 본문이 거의 비슷하더라도 언론사마다 제목은 크게 다른 경우가 많다.

7. 결론 및 개선 방안

7.1. 결론

본 연구에서는 Google Text Summarization 을 통한 생성형 한국어 문서요약 Model 을 만드는 것을 목표로 하였다. Model 을 구현한 결과 요약이 생성되었지만 만족할만한 수준의 요약문을 생성하였다고 보기는 어렵다. 한국어 생성형 문서요약 Model 구축에는 크게 두 가지의 한계점이 있었다.

첫번째로는 Dataset 확보의 어려움이다. 본 논문에서는 Model 은 뉴스 기사 100 만건이라는 Dataset 을 활용하여 훈련을 진행하였다. 이를위해 빅카인즈를 활용하여 Dataset 을 수집하였으나 가공되지 않은 Dataset 이었다. 따라서 Dataset 을 Model 에 적합한 훈련 데이터 형태로 바꿔주는 복잡한 과정이 필요하였고 많은 시간을 소모하였다. 인공지능을 활용한 문서요약 연구에서는 훈련 데이터에 의하여 Model 의 성능이 많이 좌우된다. 현재 영어의 경우 많은 나라에서 모국어로 사용되기 때문에 상대적으로 Dataset 의 양도 많고 양질의 데이터 또한 많이 존재한다. 앞으로의 자연어처리 분야에서 Dataset 의 문제점을 해결하기 위해서는 국가나 연구기관에서 인공지능을 활용한 자연어 처리 연구를 위한 양질의 Dataset 을 확보해 두고 이용할 수 있게 함으로써 데이터 확보에 소요되는 시간을 줄일 수 있도록 하는 것이 필요할 것이라 생각된다.

또한 Google 의 문서요약에서는 장기 의존성을 보안하기 위하여 문장의 단어 순서가 반대로 되어 들어가게 된다. 정확히 밝혀진 이유는 없지만 영어의 경우 문장의 성분이 반대로 된 경우 Model 및 Google 번역기 에서 성능이 향상된다고 Google Text summarization 에서 주장하였다. 영어의 경우 문장에서 주요 성분인 주어 와 동사가 대부분 앞에 존재하게 된다. LSTM 으로서 장기 의존성을 해결한다 하여도 Hidden Layer 의 Forget Gate 의 영향을 줄이기 위하여 제일 뒤에 주어와 동사를 위치하게 하여 주어와 동사의 의존성의 영향을 제일 적게 받게 하였다. 그러나 한국어에서는 이 과정의 필요성에 대하여 연구가 진행되어 있지 않았다. 따라서 이후의 연구에서는 문장의 순서에 따른 성능향상 여부를 확인해 보기를 기대해본다.

두 번째로는 현재 Google Text Summarization 의 문제점 이다. 결국 Vocab file 에 없는 단어는 Mapping 할 수 없어 Output 의 결과로 나올 수 없다는 점이다.이러한 문제점은 잘 사용되지 않는 단어로 이루어진 문장의 요약이나 고유명사로 이루어진 문장, 전문적인 글 등의 요약에서 한계점으로 작용할 것이다. 또한 수치 정보로 된 숫자를 요약할 수 없다는 것 이다.이러한 한계점은 수치 정보가 중요한 경제 분야의 문서요약 에서 큰 한계점으로 작용할 수 있다. 마지막으로 가장 큰 문제이자 한계점으로 문서요약에는 기계번역과 같이 정확하게 대응되는 정답이 정해져 있지 않다는 것 이다. 즉, 정확한 문서요약의 정답을 제시할 수 없는 상황이다. Google Text Summarization 은 2016 년도 말에 제시된 방법이다. 이러한 한계점을 해결 하기 위한 방안으로 2017 년도 말과 2018 년도 초에 제시된 두 논문을 참조하여 개선 방안에 대하여 기술하였다.

7.2. 개선 방안

1) 복사 방법론과 입력 추가 구조를 이용한 End-to-End 한국어 문서요약(End-to-end Korean Document Summarization using Copy Mechanism and Input-feeding)

위 논문의 핵심 방법은 Copy mechanism 이다. Copy mechanism 이란 디코딩 과정에서 문장을 생성할 때 필요한 어휘가 Vocab file 에 없어 발생하는 문제(Out-of-Vocabulary)와 고유명사의 출력 확률이 낮아지는 문제를 해결하기 위한 방법 이다.복사 방법론의 Alrgorighm 은 새로운 Copy 를 한 변수(e_{copy})를 기존 임베딩 공간에 덧붙이는 것 이다. 이를 통해 디코더는 더 넓은 크기의 단어 집합에서 요약문을 생성할 수 있다. 즉,결과는 새로운 단어 생성 확률과 복사 확률의 합으로 구성된다. 그림 31 에서 복사 방법론에 대한 과정을 설명하고 있다. [5]

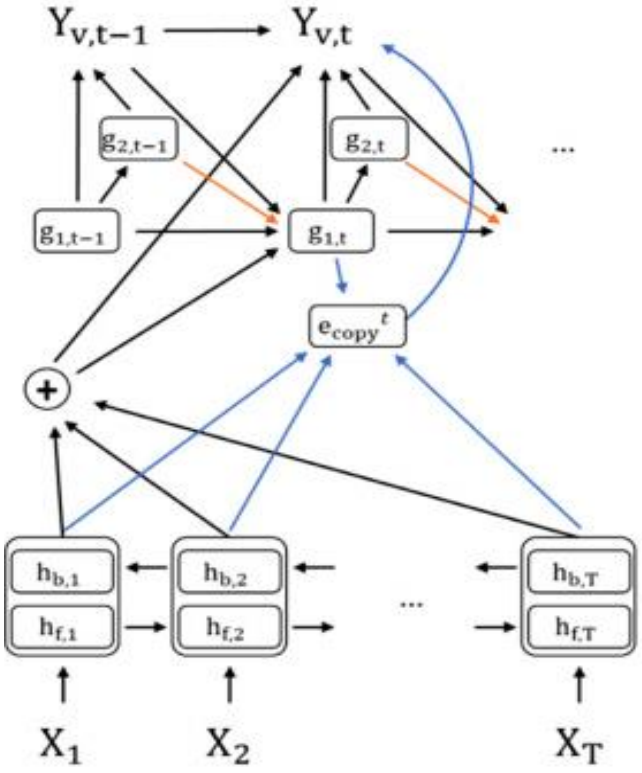


그림 31. 복사 방법론 Algorithm

출처:http://kiise.or.kr/e_journal/2017/5/JOK/pdf/08.pdf

2) 태그 정보와 복사 방법론을 활용한 수치 텍스트의 문서요약(Document Summarization for Numeric Text using Tag Information and Copy Mechanism)

본 논문에서 사용한 Google text summarization 은 숫자로 된 수치 표현을 요약 할 수 없었다.이러한 해결방법으로 제시된 것이 태그 정보와 복사 방법론을 활용한 위 논문이다. 위 논문은 태그 정보에 새로운 태그인

<Number>을 추가 하여 수치 정보가 나올 수 있다는 점에서 새로운 방법론 이였다. 그림 32 는 위 논문에서 사용한 태그의 종류와 설명에 대하여 참조하였다.[4]

태그명	설명	태그명	설명	태그명	설명
<s>	문장 시작	</s>	문장 종료	<Noun>	명사
<Verb>	동사	<Number>	수치	<Adjective>	형용사
<Punctuation>	문장 부호	<Firm>	회사명	<Determiner>	관형사
<Adverb>	부사	<Alpha>	영문자	<Conjuction>	접속사

그림 32. 태그의 종류

출처:http://s-space.snu.ac.kr/bitstream/10371/141455/1/000000149349.pdf

참고문헌

[1] 김희동, 지인영, “미디어 정보처리와 인공지능”, 담화 · 인지언어학회, 2017

[2] 김희동, 지인영, “ 심층학습을 이용한 문서요약 방법의 연구”, 국제언어인문학회, 2018

[3] 백수진, “VAE 를 이용한 의미적 연결 관계 기반 다중 문서 요약 기법”, 한국디지털정책학회, 2017

[4] 송석민, “태그 정보와 복사 방법론을 활용한 수치 텍스트의 문서 요약”, 서울대학교, 2018

[5] 최경호, 이창기, “복사 방법론과 입력 추가 구조를 이용한 End-to-End 한국어 문서 요약”, 정보과학회, 2017

[6] CY Lin, “ROUGE: A Package for Automatic Evaluation of Summaries”, Text Summarization Branches Out,2004

[7] Diederik P Kingma, Max Welling, “Auto-Encoding Variational Bayes”, ICLR, 2013

[8] IlyaSutskever, OriolVinyals, Quoc V. Le,“Sequence to Sequence Learning with Neural Networks”, NIPS, 2014

[9]MahmoodYousefi-Azar,“Text summarization using unsupervised deep learning” Expert Systems withApplications: An International Journal, 2017

[10] Mehdi Allahyari, “Text Summarization Techniques: A Brief Survey”, IJACSA, 2017

[11] Rada Mihalcea, “TextRank: Bringing Order into Texts”, Conference on Empirical Methods in Natural LanguageProcessing , 2004

[12] Sumit Chopra, Michael Auli, Alexander M. Rush, “Abstractive Summarization with Attentive RNN” , NAACL, 2016

[13] 빅카인즈
https://www.bigkinds.or.kr/

[14] Google A.I Blog
https://ai.googleblog.com/2016/08/text-summarization-with-tensorflow.html

[15] James Drulo’ s Blog
https://limseee.blogspot.com/2016/06/n-gram-tf-idf.html

[16] ratsgo's blog
https://ratsgo.github.io/from%20frequency%20to%20semantics/2017/10/06/attention/

[17] TextRank 를 이용한 문서요약
http://excelsior-cjh.tistory.com/93

[18] [S 스토리] 핵심만 콕콕! 긴 것은 NO!... '요약'에 빠진 현대인
http://www.segye.com/newsView/20180622004444

[19] Andreas Mueller, 파이썬 라이브러리를 활용한 머신러닝, 한빛미디어