박사학위논문

사용자 정의 분류체계에 따른 딥러닝 기반의 특허문서 자동분류

2022년

한 성 대 학 교 대 학 원

스마트융합컨설팅학과

스마트융합제품전공

김 성 훈

박사학위논문
지도교수 김승천

사용자 정의 분류체계에 따른 딥러닝 기반의 특허문서 자동분류

Automatic Classification of Patent Documents Based on Deep Learning According to User-defined Taxonomy

2021년 12월 일

한 성 대 학 교 대 학 원

스마트융합컨설팅학과

스마트융합제품전공

김 성 훈

박 사 학 위 논 문
지도교수 김승천

사용자 정의 분류체계에 따른 딥러닝 기반의 특허문서 자동분류

Automatic Classification of Patent Documents Based on Deep Learning According to User-defined Taxonomy

위 논문을 공학 박사학위 논문으로 제출함

2021년 12월 일

한 성 대 학 교 대 학 원

스마트융합컨설팅학과

스마트융합제품전공

김 성 훈

김성훈의 공학 박사학위 논문을 인준함

2021년 12월 일

심사위원장 이 후진 (인)

심 사 위 원 노 광현 (인)

심 사 위 원 이 문수 (인)

심 사 위 원 최 승욱 (인)

심 사 위 원 김 승천 (인)

국 문 초 록

사용자 정의 분류체계에 따른 딥러닝 기반의 특허문서 자동분류

한 성 대 학 교 대 학 원
스 마 트 융 합 컨 설 팅 학 과
스 마 트 융 합 제 품 전 공
김 성 훈

기술 혁신에 대한 권리를 보호하는 특성을 가진 특허는 대부분의 기업에서 중요한 자산으로 간주된다. 또한 특허는 기술 발전과 다양화를 대표할 수 있는 충분한 소스를 제공하므로 기술 몇 혁신 확산에 중요한 역할을 한다. 이러한 특허를 이용해서 기술발전의 추이, 경쟁사 기술 분석 등의 특허분석을 수행하기 위해서는 특허문서의 분류가 선행되어야 한다. 일반 기업의 목적에 맞는 특허분석을 위해서는 International Patent Classification(IPC), Cooperative Patent Classification(CPC) 등의 공식적인 분류체계보다는 사용자 정의 분류체계가 필요하며 사용자 분류체계에 의해 분류된 특허가 특허분석의 핵심 재료가 된다. 이러한 특허문서 분류는 전문가에 의해 대부분 수작업으로 진행되므로 많은 비용과 시간이 소요된다. 이러한 시간과 비용을 줄이기 위해 다양한 사용자 분류체계에 맞는 특허문서 분류를 자동으로 수행할 수 있도록 딥러닝을 이용한 최적의 분류모델을 찾는 것이 이 연구의 목적이다.

3가지 분류 데이터셋을 정의하고 각 데이터셋의 80%를 훈련데이터로 사용하고 20%를 테스트 데이터로 사용하였다. 키워드 기반의 분류 알고리즘 2개와 문장 기반의 분류 알고리즘 3개 총 5가지의 딥러닝 알고리즘을 선택하였다. 각 알고리즘 별로 다수의 분류모델을 만들어 각 데이터셋의 테스트데이터의 분류 정확도를 측정하였다. 또한 분류모델의 결과를 조합하여 앙상블 기법을 사용하여 각 데이터셋의 테스트데이터의 분류 정확도를 측정하였다. 3가지 데이터셋에 모두 최고의 분류정확도를 가지는 단일 분류 모델은 존재하지 않았다. 앙상블 기법은 3가지 데이터셋중 하나의 데이터셋에서는 최고의 분류정확도를 보였고 2가지 데이터셋에서는 2순위의 분류 정확도를 기록하였다.

본 연구의 목적인 다수의 사용자 분류체계에 적합한 특허분류를 위한 모델은 특정 알고리즘을 사용하는 단일 분류보다는 다수의 분류모델을 조합하는 앙상블 기법을 사용한 분류모델이 적합하다는 결과를 얻었다. 이러한 실험 결과를 바탕으로 본 연구에서는 사용자 정의 분류체계에 맞는 특허문서 자동분류를 위한 분류 아키텍처를 제안하였다. 이러한 특허문서 자동분류 아키텍처가 실제 특허분류 업무에 사용되어 특허 도메인 전문가들은 특허분석에 보다 집중할 수 있는 환경이 만들어지길 기대한다.

【주요어】 특허문서 자동분류, 사용자 정의 분류체계, 딥러닝, 앙상블

목 차

표 목 차

그 림 목 차

제 1 장 서 론

제 1 절 연구의 배경 및 필요

1) 특허분류체계와 특허분류[1)]

기술 혁신에 대한 권리를 보호하는 특성을 가진 특허는 대부분의 기업에서 중요한 자산으로 간주된다. 특허는 또한 기술 발전과 다양화를 대표할 수 있는 충분한 소스를 제공하기 때문에 기술 및 혁신 확산에 중요한 역할을 한다.

특허분류체계는 특허의 기술 분야에 따라 특허를 구분하는 기준이 되는 것으로, 특허문서의 수집, 정리 및 검색의 수단으로 사용된다. 특허분류체계의 종류로는 국제특허분류체계인 IPC(International Patent Classification)가 대표적이며, 일본 특허청의 FI(File Index), F-Term(File forming Term), 미국 특허청의 UPC(United States Patent Classification), 유럽 특허청의 ECLA(European Classification) 그리고 유럽 특허청과 미국 특허청에서 공동관리하는 CPC(Cooperative Patent Classification) 분류체계가 있다.(한국전자정보통신산업진흥회, 2009)

IPC 분류체계의 성립은 미국, 일본, 유럽 등 각국마다 다른 분류체계를 사용하여 왔으나, 국제적으로 통일된 특허 분류체계가 필요함에 따라 1968년에 도입되었다. IPC의 목적은 첫째 특허문서를 체계적으로 정리해서, 특허문서에 포함되어 있는 기술 및 권리정보에 용이하게 접근할 수 있게 하기 위함이며, 둘째 특허정보의 모든 이용자에게 정보를 선택적으로 보급하기 위함이며, 셋째 주어진 기술 분야에서 공지기술을 조사하기 위함이며, 넷째 여러 영역에서의 기술발전을 평가하는 공업 소유권 통계를 내기 위함이다.(한국특허청홈페이지)

CPC 분류체계는 IPC 보다 세분화된 특허분류 체계로서 IPC(7만여 개소)

1) 분류 : 분류체계에 따라 특허를 나누는 행위.

보다 많은 26만여 개의 특허분류 개소를 갖고 있다. CPC는 효율적인 선행기술조사를 위해 미국특허청과 유럽특허청의 주도로 2012년 개발되어, 2021년 현재 전 세계 중 30개 국가가 특허문헌을 CPC로 분류하고 있으며, 특허문헌과 비특허문헌을 포함하여 전 세계 6000만 건 이상의 문헌이 CPC로 분류되어 있다. 우리나라는 2015년 1월 이후 신규출원 특허문헌에 CPC, IPC를 함께 부여하고 있다.(한국특허청홈페이지)

특허분류는 특허문헌을 효율적으로 검색, 분석, 구조화하기 위해 특허문헌을 미리 정의된 특허분류체계에 할당하는 작업이다.

IPC, CPC 등의 공식적인 특허분류체계에 대한 분류는 특허문헌이 공개될 때 각국 특허청에 의해 수행된다. 대만민국 특허청에서는 2015년 1월 이후 신규출원 특허문헌에 CPC, IPC를 함께 부여하고 있다.

특허분류는 출원된 발명이 속하는 기술 분야를 명확히 하여 심사를 위한 선행기술조사를 용이하게 하며, 우리나라의 경우 특허출원이 되면, 한국특허정보원에서 임시 분류를 부여한 뒤, 특허청에서 기술 분야에 따라 담당 분야의 심사관을 배정하여 전문적인 심사를 가능하게 한다.

2) 특허분류 체계의 종류

가) International Patent Classification (IPC)

1971년 스트라스부르 협정에 의해 설립된 국제 특허 분류(IPC)는 특허 및 실용신안이 속하는 다양한 기술 영역에 따라 분류하기 위한 언어 독립 기호의 계층적 시스템을 제공한다. IPC의 새 버전은 매년 1월 1일에 발효된다.

IPC는 기술을 약 70,000개의 세분화된 8개 섹션으로 나눈다. 각 세분화에는 아라비아숫자와 알파벳 문자로 구성된 기호가 있다.

각 특허 문서에는 적절한 IPC 기호가 표시되어 있으며, 그 중 지난 10년 동안 매년 1,000,000개 이상이 발행되었다. IPC 기호는 특허 문서를 발행하는 국가 또는 지역 산업 재산권 사무소에서 할당한다. PCT 문서의 경우 IPC 기호는 ISA(International Search Authority)에서 할당한다.

WIPO(World Intellectual Property Organization)에 따르면 “분류는 선행

기술 검색에서 특허문헌 검색에 필수적이며, 이러한 검색은 특허 발급기관(특허청), 잠재적인 발명가 연구 및 개발부서 및 기술의 적용 또는 개발과 관련된 사람에게 꼭 필요하다.” 라고 한다.

[표 1-1] 국제특허분류체계(IPC) 분류체계표

IPC코드류	내용(한글)	내용(영문)
A 섹션	생활필수품	HUMAN NECESSITIES
B 섹션	처리조작; 운수	PERFORMING OPERATIONS; TRANSPORTING
C 섹션	화학; 야금	CHEMISTRY; METALLURGY
D 섹션	섬유; 지류	TEXTILES; PAPER
E 섹션	고정구조물	FIXED CONSTRUCTIONS
F 섹션	기계공학; 조명; 가열; 무기; 폭파	MECHANICAL ENGINEERING; LIGHTING; HEATING; WEAPONS; BLASTING
G 섹션	물리학	PHYSICS
H 섹션	전기	ELECTRICITY

나) Cooperative Patent Classification (CPC)

CPC는 IPC보다 세분화된 특허분류체계로서 IPC 보다 많은 26만여 개의 특허분류 개소를 가지고 있다. 효율적인 선행기술 조사를 위해 미국특허청과 유럽특허청의 주도로 2012년 개발되어, 2021년 현재 전 세계 중 30개 국가가 특허문헌을 CPC로 분류하고 있으며, 특허문헌과 비특허문헌을 포함하여 전 세계 6,000만 건 이상의 문헌이 CPC로 분류되어 있다.

EPO(유럽특허청)에 따르면 “CPC는 USPTO(미국특허청)와 EPO 간의 공동 파트너십으로 시작되었으며, 각 특허청은 기존 분류체계 (각각 ECLA 및 USPC)을 조화시키고 공통 분류 체계로 마이그레이션 하기로 동의했다. 이것은 두 특허청의 전략적 결정이며 현재 IP5의 분류 작업그룹 1을 통해 수행되고 있는 노력을 진전시키는 중요한 단계이다. CPC로의 마이그레이션은 세계지적재산권기구(WIPO)에서 관리하는 IPC 표준을 준수하도록 수정된 기존 유럽분류체계인 ECLA을 기반으로 개발 되었다” 라고 설명한다.

[표 1-2] 선진특허분류(CPC) 분류체계표

CPC코드	내용(한글)	내용(영문)
A 섹션	생활필수품 농업	HUMAN NECESSITIES AGRICULTURE
B 섹션	처리조작; 운수	PERFORMING OPERATIONS; TRANSPORTING
C 섹션	화학; 야금	CHEMISTRY; METALLURGY
D 섹션	섬유; 지류	TEXTILES; PAPER
E 섹션	고정구조물	FIXED CONSTRUCTIONS
F 섹션	기계공학; 조명; 가열; 무기; 폭파	MECHANICAL ENGINEERING; LIGHTING; HEATING; WEAPONS; BLASTING
G 섹션	물리학	PHYSICS
H 섹션	전기	ELECTRICITY
Y 섹션	새로운 기술 발전의 일반적 구분표; IPC의 여러 섹션에 걸친 다양한 섹션을 넘나드는 기술의 일반적 구분표; 이전의 USPC 상호참조 기술 수집 및 개요에 의해 포함	GENERAL TAGGING OF NEW TECHNOLOGICAL DEVELOPMENTS; GENERAL TAGGING OF CROSS-SECTIONAL TECHNOLOGIES SPANNING OVER SEVERAL SECTIONS OF
Z 섹션	4차 산업혁명 분류	THE 4TH INDUSTRIAL REVOLUTION

다) 사용자 정의 분류체계

사용자 정의 분류체계는 공식적인 분류체계와는 다른 관점에서의 분류체계가 필요할 경우 목적에 따라 기업, 공공기관 및 연구소의 특허전문가 및 도메인 전문가에 의해서 만들어진다. 이러한 분류체계는 기관의 목적에 따라 제품에 관련된 특허 분류체계, 목적에 관련된 특허 분류체계 및 경쟁사 기술과 관련된 특허 분류체계 등 다양한 사용자 정의 분류체계가 존재한다.

[표 1-3]은 의료기기 유형에 대한 사용자 정의 분류체계 이다.

[표 1-3] 의료기기 유형별 특허분류 체계 별 IPC 매칭현황(1,199개 특허)

구분	중분류	소분류
1	영상진단 (62)	①X선/CT(47), ②MRI/PET(2), ③초음파(13)
2	생체계측 (108)	①심박측정(8), ②청진기(3), ③심전계(19), ④뇌파검사(5), ⑤근전도검사(2), ⑥체온측정(2), ⑦호흡검사(9), ⑧청력검사(1), ⑨검안장치(24), ⑩혈류계측(15), ⑪생체진단(4), ⑫생체계측 기타(16)
3	체외진단 (77)	①혈액측정(8), ②화학분석(69)
4	진료장치 (58)	①환자용 침대(28), ②위생기구(2), ③인큐베이터(1), ④소독살균기(22), ⑤진료장치 기타(5)
5	마취호흡 (14)	①마취기(3), ②호흡보조(11)
6	수술치료 (134)	①미세수술(2), ②안과수술(8), ③방사선치료(9), ④충격파치료(5), ⑤산소챔버(3), ⑥비기계적수술(15), ⑦매체도입기(32), ⑧봉합장치(19), ⑨수술치료 기타(41)
7	치료보조 (73)	①흡입분무기(10), ②심장충격기(9), ③인공소생기(2), ④자기치료기(7), ⑤전기자극기(20), ⑥온열저온기(7), ⑦마사지장치(18)
8	정형용품 (72)	①인공관절(28), ②스텐트(25), ③뼈접합기구(19)
9	기능대체 (29)	①인공대체물(9), ②순환장치(6), ③여과장치(14)
10	의료경 (29)	①내시경(20), ②구강경(3), ③기관지경(1), ④식도경(1), ⑤복강경(1), ⑥관절경(1), ⑦이경/비경(2)
11	의료용품 (296)	①주사침(2), ②수액장치(33), ③채혈기구(6), ④의안/렌즈(2), ⑤외과용품(25), ⑥봉합사(7), ⑦피임용구(13), ⑧체액유도(15), ⑨흡수용품(91), ⑩조성물(70), ⑪의료용기(21), ⑫의료용품 기타(11)
12	치과기기 (160)	①임플란트(2), ②치과보철(44), ③교정기(24), ④절삭기(29), ⑤치과재료(14), ⑥충전기구(10), ⑦작업대(10), ⑧치과기기 기타(27)
13	재활보조 (85)	①정형기구(34), ②청각기구(10), ③시각기구(1), ④운반기구(26), ⑤재활보조 기타(14)
14	정보기기 (2)	①원격진료(1), ②기록관리(1)

* 괄호 안의 숫자는 각 분류별로 매칭된 IPC의 수

3) 특허분류의 필요성 및 활용

IPC 및 CPC 등의 공식적인 특허분류체계를 통한 특허분류의 가장 주된

목적은 선행기술 검색을 용이하게 하기 위함이다. 선행기술 검색은 특허를 심사하는 심사관에게는 새롭게 출원된 발명에 대해 특허권을 부여할 것인가의 심사에 필수적인 사항이다. 또한 발명을 출원하고자 하는 발명자에게 본인의 발명과 유사한 특허가 이미 출원 또는 등록되었는지 확인하는 과정이다.

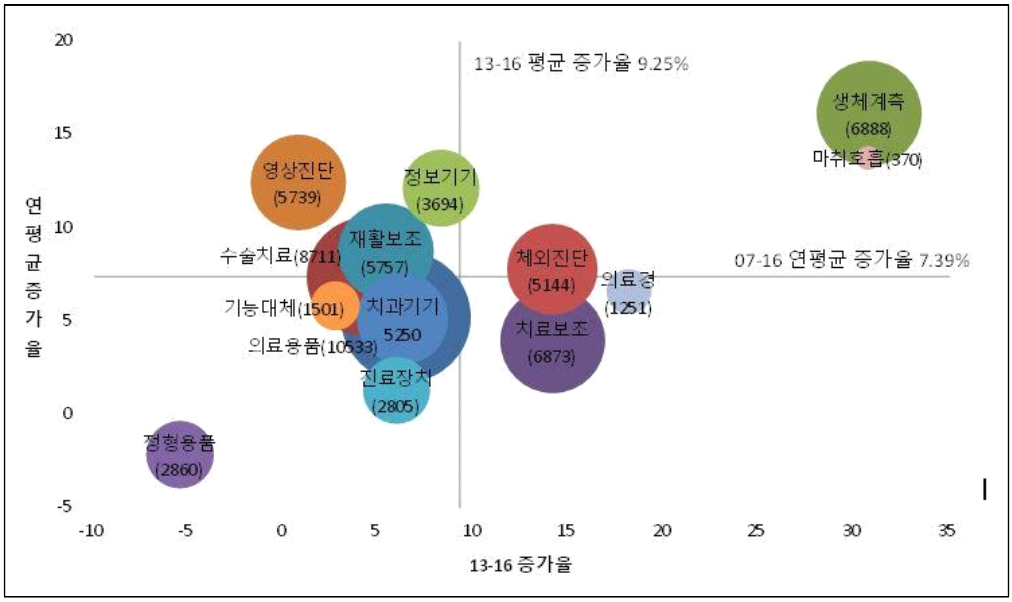
이러한 선행기술을 단순히 키워드로만 검색할 경우 다양한 기술 분야에 걸쳐 검색되기 때문에 해당 발명과 관련 없는 분야의 노이즈 특허가 검색될 가능성이 높다. 이러한 노이즈로 인해 선행기술 검색이 어려워지고 많은 비용이 들게 된다. 이러한 어려움을 해결하기 위해 각국 특허청에서는 특허분류 체계에 따른 특허분류를 통해 관심 분야에 대한 특허의 검색이 용이하도록 하였다.

새로운 분야의 기술이 만들어지고, 세분화된 분류체계를 위해 IPC 와 CPC 는 매년 특허분류 체계의 개정판을 만들고 있다. 또한 CPC 에는 Z섹션이 만들어져 4차 산업혁명 분류 섹션이 추가되었다.

특허분류는 특허동향 조사, 경쟁사 특허분석 및 산업동향 분석 등의 재료로 사용된다. 특허동향조사는 기술 동향, 주요 연구개발 주체의 연구개발 동향 등을 파악하거나, 연구개발 방향 등을 도출하기 위하여 과거부터 현재까지의 특허정보를 조사하여 분석하는 활동이다(한국특허전략개발원, 2017). 경쟁사 특허분석은 경쟁사가 보유한 특허와 새롭게 출원하는 발명에 대해 기술의 추이 및 권리범위를 분석하여 경쟁사 특허에 대한 무효 및 권리 침해에 대한 분석을 수행하는 활동이다. 산업동향 분석은 특정 산업군에 관련된 특허와 새롭게 출원된 발명을 통해 해당 산업군의 생명주기와 특정 산업군에 포함된 기업들과 기업의 특허활동을 파악하는 활동이다.

앞에서 언급한 것처럼 특허분류체계와 이에 기준한 특허분류는 특허검색과 특허분석에 사용된다. IPC 와 CPC 등의 공식적인 분류체계는 주로 특허검색에 사용되고 일반적인 특허동향 조사에 적합하다. 반면 사용자 정의 분류체계는 경쟁사 특허분석 등의 보다 세밀한 분석에 사용되는 것으로 적합하다.

[그림 1-1]은 의료기 유형에 대한 사용자 분류체계를 기반으로 분류된 특허를 이용하여 의료기기 유형별 특허 증가율을 분석한 예시이다.



[그림 1-1] 의료기기 유형별 특허 증가율 분석(오송 첨단의료 산업진흥재단)

제 2 절 연구의 목적 및 방법

사용자 정의 분류체계는 공식 분류체계와는 다른 관점의 분류체계가 필요할 경우 사용목적에 따라 도메인 전문가들에 의해서 분류체계가 정의되고 관심 특허가 분류되어 진다. 이러한 사용자 분류체계를 통해 분류되어진 특허를 통해 경쟁사 특허분석, 기술 동향 분석 등이 수행된다.

사용자 정의 분류체계에 따른 분류 작업은 특허 사무소 및 기업의 특허부서 등의 특허분류 전문가에 의해 수작업으로 진행되는 것이 대부분이다. 또한 계속적으로 출원되는 특허에 대해 분류작업은 주기적으로 지속적으로 이루어져야 한다. 이러한 수작업 분류와 주기적인 분류는 많은 시간과 비용이 소요된다. 이러한 시간과 비용을 절감하기 위해, 초기에 분류전문가에 의해 분류된 특허를 학습데이터로 사용하여 생성된 지도학습 기반의 특허자동 분류기는 좋은 해결책이 될 수 있다(김성훈, 2021).

본 연구의 목적은 사용자 분류체계에 따라 특허문헌을 자동으로 분류하는 분류 모델 및 분류기 아키텍처를 설계하는 것이다.

1) 지도학습 기반의 특허문서 분류모델 생성

사용자 정의 분류체계에 적합한 분류모델을 찾기 위해 다양한 딥러닝 모델을 생성한다. 전문가에 의해 특허분류가 수행될 때는 특허를 구성하고 있는 키워드 와 문장이 주요한 분류 기준이 된다. 사람에 의한 분류를 모사하기 위해 딥러닝 모델 구성에 있어서도 키워드 기반 모델과 문장기반 모델을 생성한다. 키워드 빈도 기반의 특징을 사용하는 Multi Layer Perceptron, 인접한 단어의 특징을 사용하는 Multi Channel CNN, 문장을 구성하는 단어의 순서를 특징으로 하는 LSTM, LSTM에 비해 긴 문장을 처리하기 위한 어텐션(Attention) 그리고 어텐션만을 사용해서 속도와 성능을 높이고자 하는 트랜스포머(Transformer)등 자연어를 처리하는 5개 딥러닝 알고리즘을 사용하여 특허분류에 맞게 변형한다.

다양한 사용자 분류체계에 적합한 모델의 분류 성능을 측정하기 위해 CPC의 특정 분류체계의 특허를 이용하여 3가지 데이터 셋을 생성한다. 다양한 관점으로 만들어진 사용자 분류체계에 대해 적합한 모델을 찾기 위해 다른 관점에서 만들어진 3가지 데이터셋 모두에서 높은 분류성능을 기록하는 분류모델을 선정하고자 한다.

각 데이터셋은 훈련데이터, 검증데이터 그리고 테스트 데이터로 분리한다. 정의된 데이터셋을 이용해서 5가지 알고리즘을 기반으로 만들어진 100여개 이상의 분류 모델에 대해서 훈련데이터로 훈련시키고 테스트 데이터로 정확도를 측정한다. 측정된 분류 정확도를 기반으로 3가지 데이터셋에서 최고의 분류 정확도를 기록하는 분류모델이 동일한 분류모델인지 데이터셋별로 각각 다른 분류모델인지 확인한다.

2) 앙상블기법 기반의 분류모델 구성

5가지 딥러닝 알고리즘을 기반으로 생성된 분류모델들이 각 데이터셋별로 개별로 나타내는 성능과 비교하기 위해 다수의 분류모델의 분류 결과를 결합하여 결과를 예측하는 앙상블기법을 사용하는 분류모델을 정의한다. 앙상블 기법은 개별 분류모델의 예측 확률값을 단순 합산하는 summation, 각 분류모델이 선택한 분류체계의 다수결을 이용하는 voting, 각 분류모델의 검증 데이터 정확도를 예측 확률값에 곱해서 합산하는 weighting 방식을 사용한다.

3가지 데이터셋에 대해서 각 분류 알고리즘에서 최고의 분류 성능을 가진 개별 문류모델 5개, 그리고 앙상블 기법 3가지 방식을 적용한 앙상블 기법 기반 분류모델 5개를 선정하여 총 8개 분류 모델의 정확도를 비교하여 앙상블 기반의 분류모델이 개별 분류모델에 비해 성능 개선이 이루어지는지 확인한다. 이를 통해 다양한 사용자 정의 분류체계에 따른 분류에 앙상블 기반 분류 모델이 적용가능하지 여부를 판단하고자 한다.

3) 다양한 분류체계의 특허분류를 위한 모델 설계

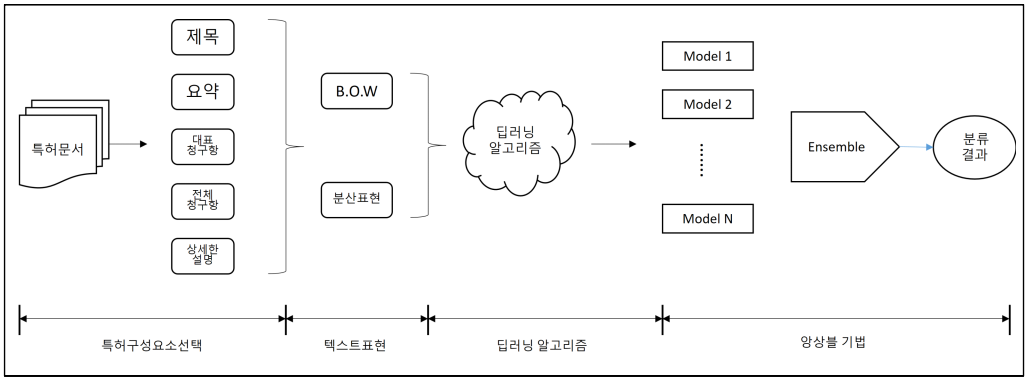
사용자 정의 분류체계는 기업별 목적, 산업분야 등에 따라 다양하게 정의되고 있다. 이러한 다양한 분류체계에 따른 특허 분류를 수행하는 제품수준의 분류아키텍처를 설계하고자 한다. 특허문서 분류기가 제품화 되고자 하면 특허 분류체계의 종류와 상관없이 일정 수준 이상의 분류 정확도가 담보되어야 하고, 사용자 편의성을 위해 모델의 학습, 훈련된 모델의 선택 및 앙상블 기법을 통한 추론이 모두 자동으로 이루어져야 한다.

본 연구에서는 앞서 언급한 다양한 실험과 그 결과를 바탕으로 제품수준의 특허문서 분류 아키텍처를 설계하고자 한다.

제 2 장 관련기술 및 선행연구 분석

기계학습 및 딥러닝을 이용한 문서 분류 및 특허문서 분류에 관한 선행 연구들은 다음과 같은 흐름을 가진다.

특허문서의 어떠한 구성 요소를 분류모델의 입력으로 사용할 것인가에 대한 연구, 텍스트의 벡터화 및 분산표현에 관한 연구, 특허분류에 적용하는 다양한 딥러닝 알고리즘에 관한 연구, 분류모델의 정확도 개선을 위한 앙상블 기법에 관한 연구가 이루어지고 있다.



[그림 2-1] 특허문서 분류에 관련된 관련기술 및 선행연구의 분야

[그림 2-1]은 특허문서 분류 연구의 분야를 문서 분류 흐름과 연동시켜 도식화한 그림이다.

근래 가장 활발하게 진행되고 있는 딥러닝 알고리즘에 대한 상세한 설명은 본연구의 3장에서 개별 알고리즘을 이용한 특허문서 분류모델에서 보다 자세한 설명을 하도록 하겠다.

제 1 절 문서 분류를 위한 기계학습/신경망 학습에 관한 연구

문서분류를 위한 기계학습에 관한 연구는 텍스트 문서에서 분류에 적합한 특징을 선택하는 방법과 문서분류에 적용 가능한 다양한 알고리즘 연구 등이 있다.

Yang et al.(1997) 텍스트 분류의 통계적 학습에서 특징 선택 방법에 대한 비교 연구를 수행하였다. 텍스트에 대해 5가지 방법의 차원 축소 기법을 사용하여 KNN(K-Nearest Neighbor) 과 LLSF(Linear Least Squares Fit mapping) 분류 기법에 적용하여 결과를 비교하였다. 연구에 사용된 5가지 차원 축소 방법은 DF(Document Frequency), IG(Information Gain), MI(Mutual Information), CHI(Chi Square Statistics) 및 TS(Term Strength) 이며 가장 높은 정확도를 내는 축소기법은 IG 와 CHI 기법이라고 보고하였다. 해당 연구에서는 Reuters 코퍼스 데이터와 분류기로는 K-최근접 이웃 분류기를 사용하였다. IG 임계값을 사용하여 고유 임계값을 최대 98% 제거하면 분류 정확도가 향상되었다고 한다. DF 임계값도 유사한 결과 패턴을 보였다. 저자는 용어의 DF, IG 및 CHI 값 사이에 강한 상관관계를 발견하였다고 한다. 이를 통해 대규모 데이터일 경우 비용이 가장 저렴한 DF 임계값을 사용하면 IG 또는 CHI 에 비해 안정적으로 사용할 수 있다고 보고하였다. 해당 연구의 특징은 단어의 여러 차원 축소 기법 중 문서분류에 적합한 축소 기법을 제시한 것이라 할 수 있다.

Hochreiter et al.(1997)은 순환신경망의 한 종류인 LSTM 모델을 발표하였다. 기존에 널리 사용된 순환신경망인 Vanilla RNN은 이전의 정보와 현재 정보를 결합하여 아웃풋을 만들어내고 이 과정이 순환되는 구조이다. 이 과정에서 입력 시퀀스가 길어지면 초기에 입력된 데이터가 뒤로 갈수로 데이터의 특징이 소멸되어 간다. 이런 문제 때문에 긴 시퀀스에 대한 처리에 좋지 못한 성능을 내는 문제가 있다. LSTM 모델은 기존 순환 신경망(Vanilla RNN)의 문제인 긴 시퀀스 입력에서 발생하는 기울기 소멸 문제를 방지하기 위해 셀, 입력 게이트, 출력 게이트, 망각게이트를 적용한 모델이다. 기본 개념은 순환

신경망에서 직전 스텝의 데이터뿐만 아니라 여러 스텝이전의 과거 데이터를 고려하여 예측한다는 개념을 도입하였다. 그러기 위해서 오래 기억해야할 데이터와 기억하지 말아야 할 데이터를 자동으로 학습하고 이를 예측에 사용하는 것이다. LSTM을 구성하는 요소인 입력게이트는 현재 정보를 기억하기 위한 게이트 이며, 삭제 게이트는 이전의 정보들 중 삭제를 위한 게이트이며, 셀은 장기 기억을 저장하는 역할을 담당한다고 한다. LSTM은 Vanilla RNN과 비교하여 긴 시퀀스의 입력을 처리하는데 높은 성능을 보였고 텍스트 문서 역시 긴 시퀀스로 이루어진 데이터 이므로 많은 연구에서 LSTM의 높은 성능이 확인되었다. 해당 연구는 자연어 처리문제에서 기존의 Vanilla RNN의 문제였던 긴 시퀀스를 가진 문장을 처리하는데 있어서 떨어지는 성능을 문장의 시퀀스중 기억 해야할 것 과 망각해야 하는 시퀀스를 학습함으로써 문제를 해결했다는 것이 연구의 성과라고 할 수 있다.

Kim(2014)은 문장 수준의 분류작업을 위해 사전 훈련된 워드 벡터를 기반으로 CNN(Convolutional Neural Networks) 모델을 제안하였다. 하이퍼파라미터의 조정이 거의 없고 간단한 CNN이라고 하더라도 사전 훈련된 워드 벡터를 적용하면 여러 벤치마크 테스트에서 우수한 결과를 달성함을 보였다. 1개의 레이어만을 가진 간단한 CNN 모델로도 우수한 성능을 보일 수 있다고 보였고 성능 향상에 가장 크게 기여한 것은 사전 훈련된 단어벡터라고 하였다. 결국 자연어 처리에서 사전 훈련된 단어벡터의 중요성을 확인 할 수 있다고 하였다. 해당 모델은 감정 분석 및 질문 분류를 포함하는 7개 작업중 4개의 작업에서 기존의 연구결과를 능가하는 성능을 기록하였다고 보고하였다. 본 연구는 컴퓨터 비전 분야에서 주목할 만한 결과를 보인 CNN 모델을 자연어 처리에 사용하여 주목할 만한 결과를 얻었다는 것이며 이에 사용된 사전 훈련된 단어 벡터의 중요성을 인식시킨 것이 해당 연구의 큰 의의라 할 수 있다.

Bahdanau et al.(2014)는 기계번역을 위한 sequence to sequence 모델에서 기존의 모델이 고정 길이 벡터로 소스 문장을 인코딩 하는 방법이 문장이 길어지면 인코더 디코더 아키텍처의 성능 향상에 병목 현상을 일으킨다고 가정하였다. 즉, Vanilla RNN 또는 LSTM의 디코더로 전달되는 인코더의 최종

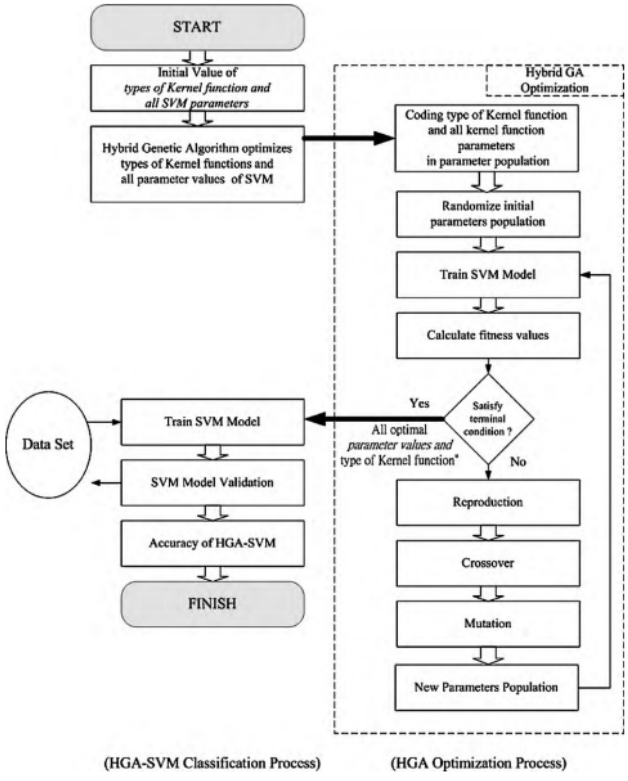
출력층인 문장 임베딩이 고정길이의 벡터이므로 번역을 위한 입력 문장 개개의 단어의 정보가 디코더로 입력되지 않는다고 저자는 판단하였다. 이 경우 짧은 문장에서는 큰 문제가 없을 수도 있지만 문장이 길어질수록 더 많은 정보가 전달되고 이를 고정된 크기의 문장 임베딩으로 표현해야 하므로 정보의 손실이 발생할 수 있다고 판단된다. 저자는 이러한 문제를 해결하고자 입력문장의 개별단어의 정보를 디코더로 전달하기 위해 인코더 디코더 모델이 학습하고 이를 활용해서 번역하도록 하고 있다. 이때 정렬은 소프트하게 구성되며 디코딩 시 매 시점마다 해당 시점과 가장 관련이 있는 번역할 단어의 위치를 찾게 된다. 이런 방법을 사용함으로써 모델은 대상 문장을 고정된 크기의 벡터로 변환할 필요가 없게 된다. 영어-불어 번역작업에서 기존 최고 성능의 구문 기반 번역 시스템과 필적할 만한 결과를 낸다고 보고하였다. 해당 논문은 기존의 순환신경망의 고정벡터로 이루어진 문장 임베딩을 탈피하여 각 스텝에서 입력되는 개별단어의 정보를 이용하여 예측할 수 있게 하는 모델을 제시한 주요한 연구라고 할 수 있다.

Vaswani et al.(2017)은 순환신경망과 CNN 모델 없이 어텐션 만을 사용하여 설계된 트랜스포머(Transformer)라는 새로운 구조를 제안하였다. 트랜스포머의 구조는 크게 인코더와 디코더로 나눌 수 있다. 트랜스포머의 인코더는 N개의 인코더 블럭이 쌓인 형태이고 인코더 블럭의 결과값은 다음 인코더 블럭의 입력값으로 사용된다. 하나의 인코더 블럭는 멀티 헤드 어텐션과 피드 포워드 네트워크로 구성되어 있으며 이 블록이 N개 쌓여 인코더가 구성된다. 각각의 블록에서 는 단어의 위치를 판단할 수 없으므로 입력문장을 임베딩으로 변환 후 위치 인코딩의 값과 더해주게 된다. 이를 통해 문장속에서 각 단어의 위치를 파악할 수 있도록 하였다. 트랜스포머의 디코더는 인코더와 마찬가지로 N개의 디코더 블록이 쌓인 형태이고 인코더의 출력값을 입력값으로 사용한다. 각 디코더 블록은 마스크된 멀티 헤드 어텐션, 멀티헤드 어텐션 그리고 피드 포워드 네트워크로 구성되어 있으며 이 블록이 N개 쌓여 디코더가 구성된다. 트랜스포머 에서는 단어의 관계를 이해하기 위해 셀프 어텐션을 사용하였으며 이를 계산하기 위해 쿼리, 키, 밸류 행렬이라는 세가지 행렬을 사용하였다. 트랜스포머 모델은 기계번역 작업에 대한 실험결과에서 병렬처리

가 가능하고 학습시간이 대폭 감소하였으며 동시에 품질이 우수함을 보고 하였다. 멀티헤드 어텐션, 위치 인코딩 그리고 셀프 어텐션 등의 기법을 사용하여 성능에 비해 빠른 속도로 우수한 BLEU score(Bilingual Evaluation Understudy score)를 기록하였다. 해당 논문에서 제안된 트랜스포머 구조는 자연어 처리 문제에서 순환신경망과 LSTM을 빠르게 대체하였고 최신의 모델인 BERT, GPT 등의 자연어 처리 모델에 적용되는 기본 모델이 되었다.

제 2 절 딥러닝을 이용한 특허문서 분류 연구

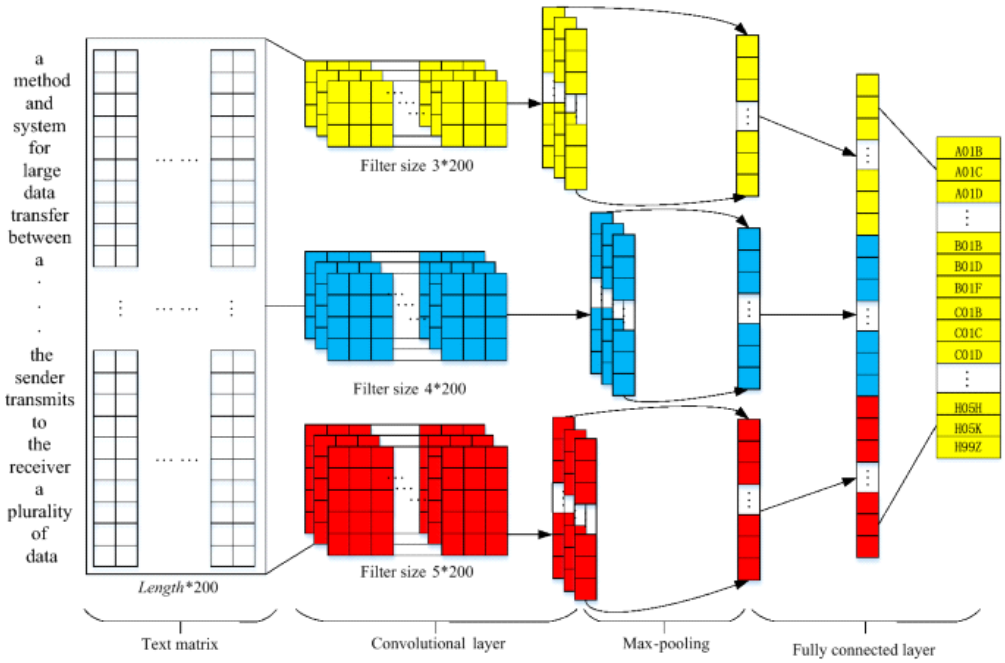
딥러닝을 이용한 특허문서 분류연구는 다양한 딥러닝 알고리즘을 특허분류에 적용한 연구가 다양하게 수행되었다.



[그림 2-2] Optimization process of HGA-SVM (Wu et al., 2010)

Wu et al.(2010) 은 기존의 특허분류 연구의 대부분이 특허의 실제 사례가 아닌 IPC 위주의 분류에 집중되어 있다고 문제 제기를 하였다. 이 연구에서는 HGA-SVM(유전자 기반 지원 벡터머신) 을 이용하여 기존의 SVM에 비해 높은 성능을 기록함을 보고하였다. 또한 이 연구에서는 기존의 IPC 분류가 아닌 반도체 장비 부품에 대한 234개의 특허문서를 포함하는 전문가의 사용자 정의 분류체계에 대한 분류를 수행하여 높은 성능을 유도하였다. 이런 결과를 바탕으로 특허 분류 및 검색 시스템의 개선을 위한 새로운 의사 결정 프로세스를 제안하였다. 제안된 의사 결정 프로세스는 1단계에서는 특허 전문가에 의해 주요 특허가 검색되고 키워드가 추출되고 2단계에서는 제안된 HGA-SVM 모델을 추출된 키워드로 훈련하여 특허를 분류하는 단계를 가지는 프로세스이다. 본 연구는 IPC가 아닌 실제 사례의 사용자 분류체계를 사용한 특허데이터로 연구를 진행하였으며, 분류 성능을 높이기 위해 특허전문가와 분류모델을 함께 적용하는 방법을 제안한 점이 특징이라 할 수 있다.

Grew et al.(2017)은 IPC 서브클래스 레벨의 분류를 위해 텍스트 분산표현과 LSTM(Long Short Term Memory network)를 적용하는 연구를 진행하였다. 텍스트가 가지는 의미론적 순차적 특성을 반영하기 위해 LSTM의 특징인 순환 처리 특성으로 인해 채택하였으며, Word2Vec은 텍스트 데이터를 LSTM 네트워크에 대한 입력으로 제공되는 의미 있는 표현으로 변환하는데 사용하였다. Word2vec은 입력으로 큰 텍스트 말뭉치를 사용하고 일반적으로 수백차원의 벡터 공간을 생성하며 말뭉치의 각 고유 단어에는 이 공간의 해당 벡터가 할당된다. 단어 벡터는 벡터 공간에 위치하므로 말뭉치에서 공통 컨텍스트를 공유하는 단어는 동일한 공간에서 서로 근접하게 위치한다. Word2Vec은 특허 텍스트를 LSTM이 분류 학습에 사용할 표현으로 변환한다. 이 변환은 문서의 컨텍스트에서 주어진 단어 의미를 나타낸다. 해당 연구의 분류 정확도는 서브클래스 레벨에서 63%를 기록하였다. 해당 연구는 단어의 분산표현인 Word2vec과 LSTM을 연결하여 특허문서 분류를 수행했다는 데 의미가 있다.

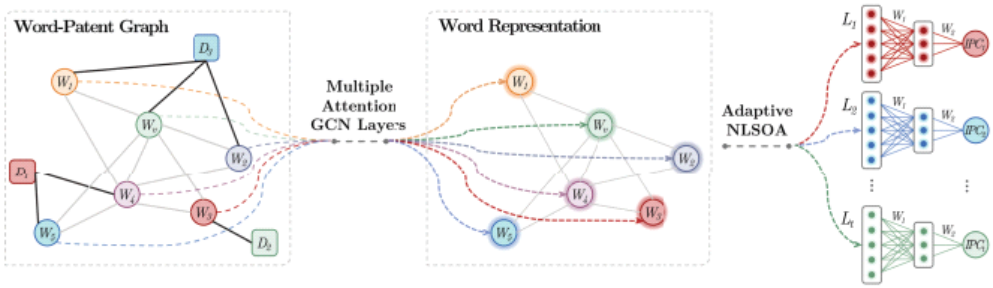


[그림 2-3] The architecture of the CNN model in DeepPatent algorithm(Li et al., 2018)

Li et al.(2018)은 IPC 서브클래스 레벨에 대한 특허분류를 위해 CNN 과 word vector embedding 에 기반한 딥러닝 알고리즘을 제안하였고 이를 DeepPatent 라고 명명하였다. [그림 2-2]에서 보이는 것처럼 DeepPatent의 구조는 3개 채널의 Convolution layer를 가지는 구조이다. 각 채널은 필터 크기가 각각 3, 4, 5를 가진 레이어를 표현하다. 이 연구에서는 표준 특허 분류 벤치마크 데이터 세트 CLEF-IP에서 알고리즘을 평가하고 다른 알고리즘과 비교하였다. DeepPatent는 제목과 요약 정보만 사용하여83.98%의 분류 정밀도를 달성하여 동일한 정보를 학습에 사용하는 기존 알고리즘을 모두 능가하였다고 보고 하였다. 또한 연구자들이 참여한 새로운 벤치마크 데이터 세트인 USPTO-2M (637개의 분류체계)에서 추가 테스트 하여 73.88%의 분류 정밀도를 달성하였다. 해당 연구는 단어벡터 임베딩과 CNN 모델을 사용하여 특허 분류에 있어서 최고 성능을 기록했다는 것에 의의가 있다.

Lee et al.(2020) 은 사전 훈련된 BERT(Bidirectional Encoder Representations from Transformers) 모델을 fine-tuning(미세조정) 하여 이

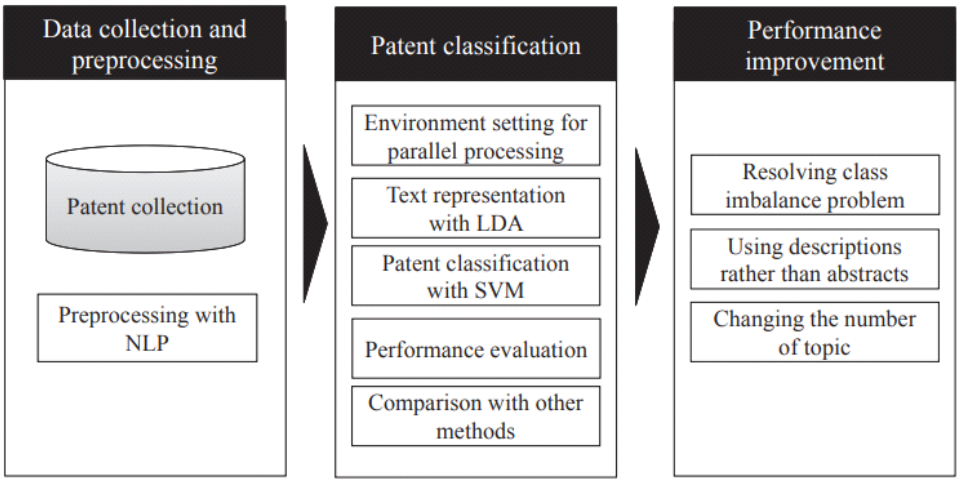
를 특허분류에 적용하였다. 이 연구에서는 CNN과 word vector embedding에 기반한 DeepPatent 보다 높은 성능을 보였다고 보고하였다. 연구자들은 해당 논문을 통해 사전 훈련된 BERT모델 및 특허 분류를 위한 미세조정으로 DeepPatent를 능가하는 분류 성능을 기록하였고, 미래의 연구자들이 사용할 수 있는 SQL문을 포함하는 CPC 하위 클래스 수준의 대규모 데이터세트인 USPTO-3M 데이터셋을 구축하였으며, 기존의 통념과 달리 청구항만으로도 특허 분류에 충분함을 보였다고 보고 하였다. 해당 연구는 최신의 언어모델인 BERT 모델을 미세조정 하여 특허분류에 사용했다는 점과 특허 청구항만을 이용해서 높은 분류 성능을 달성했다는 것이 특징이라 할 수 있다.



[그림 2-4] Overall Architecture of Non-local Attention-based Graph Convolutional Network(Tang et al., 2020)

Tang et al.(2020)은 다중 레이블 특허 문서 분류를 위해 문서-단어 연관 및 단어-단어 동시 발생을 동시에 학습하여 문서의 풍부한 의미론적 임베딩을 생성하였고, 그래프 컨볼루션 네트워크를 기반으로 하는 레이블 attention 모델을 제안하였다. [그림 2-4]은 해당 연구에서 제안한 모델로서 AGCN(Multiple Attention Graph Convolutional Network) 과 A-NLSOA(Adaptive Non-Local Second-Order Attention)이 결합된 아키텍처를 나타내고 있다. AGCN은 텍스트 그래프를 통해 문서-단어 연관 및 단어 동시 발생을 인식하여 단어의 유익한 표현을 학습 할 수 있도록 충분한 표현력을 가지고 있다고 한다. A-NLSOA는 GCN에서 생성된 텍스트 표현에서 비국소적이고 세분화된 의미론적 관계를 효과적으로 인식하는데 사용할 수 있다고 하였다. 해당연구를 통해 대규모 CIRCA(거의 대강의 의미) 특허

데이터베이스에서 7개의 baseline 기준과 비교하여 해당 모델이 모든 7개의 baseline 모델보다 높은 성능을 보인다고 보고하였다. 해당 연구는 문장속 단더의 관계를 Graph로 표현하여 문서-단어, 단어-단어의 관계를 AGCN을 통해 학습하고 이를 A-NLSOA 라는 모델을 이용해서 분류하는 모델이다. 단어의 표현벡터 뿐만 아니라 문서-단어 관계의 표현도 함께 사용해서 모델의 입력으로 사용하는 특징을 가진 연구라고 할 수 있다.



[그림 2-5] Yun et al.(2020)의 전체 프로세스(Yun et al., 2020)

Yun et al.(2020)은 기존의 특허분류 연구에서 분류체계에 어떤 특허가 포함되어 있는지, 특허 내용을 구조화된 형태로 효과적으로 표현하여 분류에 사용하는 방법에 관해 충분히 연구되지 못했다고 판단하였다. 연구자는 문서를 표현하는 벡터 공간 모델인 TF-IDF는 기계학습 입력으로 사용하기에는 텍스트를 정확히 표현하지 못할 수 있다고 판단하였다. 저자는 주제 확률 벡터가 특허 문서의 주제를 효과적으로 나타낼 수 있다고 가정하였고 특허 문서를 주제 모델링을 통한 확률 벡터로 표현하였고 이를 기계학습의 입력으로 사용하는 방법을 제시하였다. 요약하면 해당 연구에서는 텍스트 표현을 위해 주제 모델링 기법과 LDA(Latent Dirichlet allocation)를 사용하였다. LDA의 결과를 SVM 모델의 입력으로 사용하여 특허를 분류하는 방법을 사용하였다. 이를 통해 특허 내용의 구조화된 형태를 이용한 특허 분류를 수행한다고 보고

하였다. 해당 연구는 특허분류체계의 계층적 구조 특성을 이용하여 주제 확률 벡터로 변환하고 이를 분류기의 입력으로 사용하는 특징을 가진다.

제 3 절 분류문제에 있어서 앙상블 적용에 관한 연구

분류문제의 성능을 높이기 위해 개별 모델의 결과를 조합하는 앙상블 기법에 대한 연구가 다양하게 진행되었다. 앙상블기법은 훈련 데이터를 다양하게 하는 기법(Varying Training Data), 모델 알고리즘을 다양하게 하는 기법(Varying Models) 그리고 결과의 조합 방법(Varying Combinations) 방법 등이 연구되고 있다.

Hansen et al.(1990)은 분류를 위한 신경망의 성능과 훈련을 개선하기 위한 방법을 제안하였다. CrossValidation(교차 검증)은 통계적 방법으로 데이터셋에 대한 매개변수의 선택을 결정하는 표준 도구이다. 저자는 데이터셋의 오류를 최소화하는 것 보다는 향후 입력된 데이터셋의 오류를 최소화 하는 것이 목표이기 때문에 교차검증은 분류의 성능의 일반화를 위한 유용한 도구라고 하였다. 또한 저자는 기존의 방식인 데이터셋에 적합한 하나의 신경망을 통해 예측하는 것보다 전체 신경망 셋을 유지하고 적절한 집합적 의사결정 전략(앙상블)으로 모두 실행하는 것이 더 나은 예측을 할 수 있을 것이라고 하였다. 앙상블은 가중치의 선택에 따라 많은 로컬 최소값에서 최적화 된다는 사실 때문에 바람직한 방법이라 주장하였다. 요약하면 해당 연구는 신경망의 매개변수와 아키텍처를 최적화하기 위한 도구로 교차 검증을 제안하였고, 유사한 신경망의 앙상블을 사용하여 일반화 오류를 줄일 수 있음을 보고하였다. 해당 연구에서는 기존의 표준화된 단일 네트워크의 최적화된 파라메타를 찾는 것이 아니라 교차 검증을 통해 다양한 파라메타 셋을 집합화하고 그로 인해 만들어지는 다수의 네트워크의 결과를 조합하는 앙상블 방법을 제안한 것이 특징이다.

Breiman(1996)은 여러 가지 버전의 predictor(추론기)를 생성하고 이를 사용하여 집계된 추론결과를 생성하는 Begging predictor를 제안하였다. 집계된 수치 결과를 예측할 때 버전에 대한 평균을 내고 분류를 예측할 때 다수결을 수행하였다. 학습 데이터셋의 부트스트랩의 복제본을 만들고 이를 새로운 학습세트로 사용하여 여러 버전의 predictor를 생성하였다. 이 연구에서 실제

및 시뮬레이션 데이터셋에 대한 분류 테스트 및 선형 회귀의 회귀 트리 및 하위 집합 선택에서 Begging은 정확도에서 상당한 이득을 줄 수 있다고 보고하였다. 저자는 Begging의 중요한 요소는 예측의 불안정성이라 판단하였으며 학습데이터를 교란하여 구성된 예측 변수에 상당한 변화를 일으킬 수 있는 경우 Begging은 정확도를 향상시킬 수 있다고 보고하였다. 해당 연구에서는 다양한 분류기를 생성한 후 이를 사용하여 Begging 방식의 앙상블을 이용해 성능을 개선한 것이 특징이라고 할 수 있다.

Xie et al.(2013)은 신경망의 분류 성능을 향상하기 위해 Horizontal Voting Vertical Voting 과 Horizontal Stacked 앙상블 방법을 제안하였다.

Huang et al.(2017)은 앙상블을 위한 여러 신경망을 학습시키는 비용을 줄이기 위해 1번의 훈련으로 여러 신경망을 앙상블하는 방법인 Snapshot 앙상블을 제안하였다. Snapshot 앙상블은 단일 신경망을 훈련하고 최적화 경로를 따라 여러 로컬 최소값으로 수렴한 모델 매개변수를 저장하는 방식이다. 일련의 실험에서 다양한 네트워크 아키텍처 및 학습작업과 호환됨을 보였고, 추가 훈련비용 없이 최신 단일 모델보다 일관되게 낮은 오류율을 기록하였다고 보고하였다. CIFAR-10 및 CIFAR-100에서 신경망 Snapshot 앙상블은 각각 3.7%, 17.4%의 오류율을 기록하였다고 보고하였다.

Benities et al.(2018)은 Support Vector Machine(SVM)을 기반으로 캐릭터와 단어의 조합을 피쳐로 사용하여 10-fold cross validation 방법으로 학습데이터를 훈련시켜 모델을 생성하였고 앙상블을 통하여 특허문서 분류를 진행하였다. ALTA 2018 이라는 특허문서 분류 경진대회에서 0.778 의 F1-Score를 기록하였으며 14개의 참가 팀에서 1위의 결과를 얻었다고 보고하였다. 해당 연구는 SVM 단일 알고리즘에 피쳐와 데이터의 변주를 통해 앙상블을 진행한 사례이다.

앞서 소개한 기존의 선행연구와 본 연구는 다음과 같은 차이점이 있다. 첫째, 기존의 연구들은 하나의 데이터셋을 이용해서 테스트 및 결과를 도출하였다면 본 연구에서는 다수의 데이터셋을 이용하여 적합한 분류모델 및 방법을 찾고자 하였다. 둘째, 기존의 연구들은 특허의 입력셋을 하나로 고정했다면 본 연구에서는 다수의 입력셋을 사용하였다. 셋째, 기존의 선행연구들은 단일

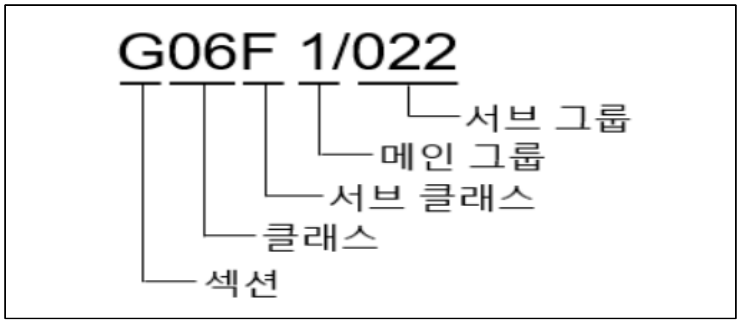
알고리즘 기반의 앙상블 방법을 사용했다면 본 연구에서는 키워드 기반 분류 알고리즘과 문장 기반 분류 알고리즘의 서로 다른 알고리즘에서 만들어진 분류모델의 결과를 앙상블 기법을 사용하여 조합한 것이 큰 차이점이라 할 수 있다.

제 3 장 딥러닝 모델을 이용한 특허문서 분류

제 1 절 실험 데이터 정의

본 연구에서는 총 3가지 종류의 데이터 셋을 사용하였다. 분류의 신빙성을 확보하기 위해 CPC 특허분류 체계를 사용한 미국에서 등록된 특허를 채택하였다.

화학분야의 분류체계인 C 섹션의 서브클래스 분류체계와 전기전자 분야의 분류체계인 H 섹션의 서브클래스 분류체계의 데이터를 사용하였다.



[그림 3-1] CPC의 구조 및 표기 예

분류체계에 포함된 특허데이터는 각 서브클래스 분류별 최신 등록특허 200건을 각각 선택하였다. 200건 중 훈련데이터와 테스트 데이터의 비율은 8:2로 선택하였으며 과거의 특허를 학습하여 미래의 특허를 분류한다는 개념을 적용하기 위하여, 테스트 데이터는 훈련데이터보다 최근에 발행된 특허로 구성하였다.

이렇게 다수의 데이터셋을 구성한 이유는 사용자 분류체계는 다양한 구성을 가지므로 다수의 데이터셋 모두에서 높은 분류성능을 가지는 분류모델을

찾고자 함이다. 다양한 딥러닝 알고리즘과 매개변수를 이용해서 만들어진 분류모델중 3가지 데이터셋 모두에서 최고의 분류성능을 가진 분류모델을 찾는 것이 목적이라 할 수 있다.

1) 데이터 셋 #1

[표 3-1] CPC C01계열 분류표(2021년 8월 버전)

CPC 코드	내용(영어)	내용(한글)
C	CHEMISTRY; METALLURGY	화학; 야금
C01	INORGANIC CHEMISTRY	무기화학
C01B	NON-METALLIC ELEMENTS; COMPOUNDS THEREOF; {METALLOIDS OR COMPOUNDS THEREOF NOT COVERED BY SUBCLASS C01C}	비금속 원소; 그 화합물; {서브클래스 C01C에 포함되지 않는 준금속 또는 그 화합물}
C01C	AMMONIA; CYANOGEN; COMPOUNDS THEREOF ({metal hydrides, monoborane, diborane or addition complexes thereof C01B6/00}; salts of oxyacids of halogens C01B11/00; peroxides, salts of peroxyacids C01B15/00; thiosulfates, dithionites, polythionates C01B17/64; compounds containing selenium or tellurium C01B19/00; azides C01B21/08; {compounds other than ammonia or cyanogen, containing nitrogen, non-metals and optionally metals C01B21/082}; metal imides or amides C01B21/092; nitrites C01B21/50; {compounds of noble gases C01B23/0005}; phosphides	암모니아; 시안; 화합물 ({금속 수소화물, 모노 보란, 디보 란 또는 이의 부가 착물 C01B6/00}; 할로겐의 옥시 산염의 염 C01B11/00; 퍼옥 시드, 과산의 염 C01B15/00; 티오 술 페이트, 디티 오 나이트, 폴리 티온 산염 C01B17/64을 함유하는 화합물) 텔 루륨 C01B19/00; 아 지드 C01B21/08; {질소, 비금속 및 선택적으로 금속을 함유하는 암모니아 또는 시아 노 이외의 화합물 C01B21/082}; 금속 이미 드 또는 아미드 C01B21/092; 아질산염 C01B21/50; {고귀한 가스의 화합물 C01B23/0005}; 포스 파이드 C01B25/08; 인 C01B25/16의 옥시 산 염; 실리콘 C01B33/00 함유 화합물; 붕소

	C01B25/08; salts of oxyacids of phosphorus C01B25/16; compounds containing silicon C01B33/00; compounds containing boron C01B35/00)	C01B35/00 함유 화합물)
C01D	COMPOUNDS OF ALKALI METALS, i.e. LITHIUM, SODIUM, POTASSIUM, RUBIDIUM, CAESIUM, OR FRANCIUM (metal hydrides {monoborane, diborane or addition complexes thereof}C01B6/00; salts of oxyacids of halogens C01B11/00; peroxides, salts of peroxyacids C01B15/00; sulfides C01B17/22; thiosulfates, dithionites, polythionates C01B17/64; compounds containing selenium or tellurium C01B19/00; binary compounds of nitrogen with metals C01B21/06; azides C01B21/08; {compounds other than ammonia and cyanogen, containing nitrogen and other non-metals C01B21/082}; metal amides C01B21/092; nitrites C01B21/50; phosphides C01B21/50; {compounds of noble gases C01B23/0005}; phosphides C01B25/08; salts of oxyacids of phosphorus C01B25/16; carbides C01B32/90; compounds containing silicon C01B33/00; compounds containing boron C01B35/00; cyanides C01C3/08; salts of cyanic acid C01C3/14; salts of	알칼리 금속의 화합물, 즉. 리튬, 나트륨, 칼륨, 루비듐, 세슘, 또는 프란슘 (금속 수소화물 {단일보란, 디보란 또는 그의 복합물 첨가} C01B6/00; 할로겐의 옥시산염 C01B11/00; 과산화물, 페르옥소산염 C01B15/00; 황화물 C01B17/22; 티오황산염, 아디티온산염, 폴리티온산염 C01B17/64; 셀레늄 또는 텔루륨을 포함하는 화합물 C01B19/00; 금속을 가지는 질소의 2원 화합물 C01B21/06; 아지드화물 C01B21/08; {암모니아 및 시아노겐, 다른 비금속 및 질소를 포함하는 것 이외에 화합물 C01B21/082}; 금속 아미드 C01B21/092; 아질산염 C01B21/50; 인화물 C01B21/50; {비활성 가스의 화합물 C01B23/0005}; 인화물 C01B25/08; 인의 옥시산염 C01B25/16; 탄소화물 C01B32/90; 실리콘을 포함하는 화합물 C01B33/00; 붕소를 포함하는 화합물 C01B35/00; 시안화물 C01C3/08; 시안을 함유한 산염 C01C3/14; 시안아미드의 염 C01C3/16; 티오시안산 C01C3/20)

	cyanamide C01C3/16; thiocyanates C01C3/20)	
C01F	COMPOUNDSOFTHEMETALSBERYLLIUM,MAGNESIUM,ALUMINIUM,CALCIUM,STRONTIUM,BARIUM,RADIUM,THORIUM,OROFTHERARE-EARTHMETALS(metalhydrides{monoborane,diboraneoradditioncomplexesthereof}C01B6/00;saltsofoxyacidsofhalogensC01B11/00;peroxides,saltsofperoxyacidsC01B15/00;sulfidesorpolysulfidesofmagnesium,calcium,strontium,orbariumC01B17/42;thiosulfates,dithionites,polythionatesC01B17/64;compoundscontainingseleniumortelluriumC01B19/00;binarycompoundsofnitrogenwithmetalsC01B21/06;azidesC01B21/08;{compoundsotherthanammoniaorcyanogencontainingnitrogenandnon-metalsandoptionallymetalsC01B21/082;amidesorimidesofsiliconC01B21/087};metal{imidesor}amidesC01B21/092,{C01B21/0923};nitritesC01B21/50;{compoundsofnoblegasesC01B23/0005};phosphidesC01B25/08;saltsofoxyacidsofphosphorusC01B25/16;carbidesC01B32/90;compoundscontainingsiliconC01B33/00;compoundscontainingboronC01B35/00;compoundshavingmolecularsievepropertiesbutnothavingbase-exchangepropertiesC01B37/00;compoundshavingmolecularsieveandbase-exchangeproperties,e.g.crystallinezeolites,C01B39/00;cyanidesC01C3/08;saltsofcy	금속 베릴륨, 마그네슘, 알루미늄, 칼슘, 스트론튬, 바륨, 라듐, 토륨, 또는 희토류 금속의 화합물 (금속 수소화물 {단일 보란, 디보란 또는 그의 추가 복합체} C01B6/00; 할로겐의 옥시산염 C01B11/00; 과산화물, 퍼옥시산염 C01B15/00; 마그네슘, 칼슘, 스트론튬 또는 바륨의 다황화물 또는 황화물 C01B17/42; 티오황산염, 아디티온산염, 폴리티온산염 C01B17/64; 셀레늄 또는 텔루륨을 포함하는 화합물 C01B19/00; 금속을 가지는 질소의 2원 화합물 C01B21/06; 아지드화물 C01B21/08; {암모니아 또는 질소를 포함하는 시아노겐 및 비금속 및 선택적 금속 이외에 화합물 C01B21/082; 실리콘의 이미드 또는 아미드 C01B21/087}; 금속 {이미드 또는} 아미드 C01B21/092, {C01B21/0923}; 아질산염 C01B21/50; {비활성 가스의 화합물 C01B23/0005}; 인화물 C01B25/08; 인의 옥시산염 C01B25/16; 탄소화물 C01B32/90; 실리콘을 포함하는 화합물 C01B33/00; 붕소를 포함하는 화합물 C01B35/00; 분자체 속성을 가지지만 염기 교환 속성을 가지지 않는 화합물 C01B37/00; 분자체 및 염기 교환 속성을 가지는 화합물, 예. 결정체 제올라이트, C01B39/00; 시

	anicacidC01C3/14;saltsofcyanamide C01C3/16;thiocyanatesC01C3/20;{ doublesulfatesofmagnesiumwithsodi umorpotassiumC01D5/12;withother alkalimetalsC01D15/00,C01D17/00 })	안화물 C01C3/08; 시안산의 염 C01C3/14; 시안아미드의 염 C01C3/16; 티오시안산염 C01C3/20; {나트륨 또는 칼륨을 가지는 마그네슘의 이중 황산염 C01D5/12; 다른 알칼리 금속을 가지는 것 C01D15/00, C01D17/00})
C01G	COMPOUNDS CONTAINING METALS NOT COVERED BY SUBCLASSES C01D OR C01F (metal hydrides {monoborane, diborane or addition complexes thereof}C01B6/00; salts of oxyacids of halogens C01B11/00; peroxides, salts or peroxyacids C01B15/00; thiosulfates, dithionites, polythionates C01B17/64; compounds containing selenium, or tellurium C01B19/00; binary compounds of nitrogen with metals C01B21/06; azides C01B21/08; {compounds containing nitrogen, other non-metals and metal C01B21/082}; metal amides C01B21/092; nitrites C01B21/50; {compounds of noble gases C01B23/0005}; phosphides C01B25/08; salts of oxyacids of phosphorus C01B25/16; carbides C01B32/90; compounds containing silicon C01B33/00; compounds containing boron C01B35/00; compounds having molecular sieve properties but not having	서브클래스 C01D 또는 C01F에 의해 포함되지 않는 금속을 포함하는 화합물 (금속 수소화물 {단일 보란, 디아보란 또는 그의 추가 복합제} C01B6/00; 할로겐의 옥시산염 C01B11/00; 과산화물, 염 또는 퍼옥시산 C01B15/00; 티오황산염, 아디티온산염, 폴리티오네이트 C01B17/64; 셀레늄, 또는 텔루륨을 포함하는 화합물 C01B19/00; 금속을 가지는 질소의 이원 화합물 C01B21/06; 아지드화물 C01B21/08; {질소, 다른 비금속 및 금속을 포함하는 화합물 C01B21/082}; 금속 아미드 C01B21/092; 아질산염 C01B21/50; {비활성 가스의 화합물 C01B23/0005}; 인화물 C01B25/08; 인의 옥시산염 C01B25/16; 탄소화물 C01B32/90; 실리콘을 포함하는 화합물 C01B33/00; 붕소를 포함하는 화합물 C01B35/00; 분자체 속성을 가지지만 염기 교환 속성은 가지지 않는 화합물 C01B37/00; 분자체 및 염기 교환 속성을 가지는 화합물, 예. 결정체 제올라이트 C01B39/00; 시

	base-exchange properties C01B37/00; compounds having molecular sieve and base-exchange properties, e.g. crystalline zeolites, C01B39/00; cyanides C01C3/08; salts of cyanamide C01C3/16; thiocyanates C01C3/20)	안화물 C01C3/08; 시안아미드의 염 C01C3/16; 티오시안산염 C01C3/20)

데이터셋 #1은 CPC C01 계열의 서브클래스(C01B, C01C, C01D, C01F, C01G) 분류체계로 분류된 특허데이터를 사용한다. 총 5개의 분류체계로 구성되었으며 분류체계에 포함된 미국 등록특허를 최신 출원일 순으로 각 200개씩 총 1,000개의 특허로 구성하였다. 훈련데이터와 테스트 데이터는 8:2로 분리하였다. 훈련데이터와 테스트 데이터의 분리기준도 출원일순으로 분리하였다. 이러한 분리기준은 과거의 특허문서를 훈련해서 미래의 특허문서를 분류하는 것을 모사하기 위해서이다.

[표 3-2] 데이터셋 #1의 구성

CPC 코드	훈련 데이터	검증 데이터	테스트 데이터	합계
C01B	144	16	40	200
C01C	144	16	40	200
C01D	144	16	40	200
C01F	144	16	40	200
C01G	144	16	40	200
합계	720	80	200	1,000

훈련데이터는 딥러닝 분류모델을 훈련하는데 사용되며 검증데이터는 모델의 훈련 진행상황을 검증하는데 사용된다. 훈련이 끝난 후 모델의 분류성능을 측정하기 위해 테스트데이터를 사용한다.

2) 데이터 셋 #2

[표 3-3] CPC H01계열 분류표(2021년 8월 버전)

CPC 코드	내용(영어)	내용(한글)

H	ELECTRICITY	전기
H01	BASIC ELECTRIC ELEMENTS	기본적 전기소자
H01B	CABLES; CONDUCTORS; INSULATORS; SELECTION OF MATERIALS FOR THEIR CONDUCTIVE, INSULATING OR DIELECTRIC PROPERTIES (selection for magnetic properties H01F1/00; waveguides H01P{; printed circuits H05K})	케이블; 지휘자; 절연체; 전도성, 절연 또는 유전체 특성을위한 재료 선택 (자기 특성 H01F1/00; 도파관 H01P {; 인쇄 회로 H05K} 선택)
H01C	RESISTORS	저항기
H01F	MAGNETS; INDUCTANCES; TRANSFORMERS; SELECTION OF MATERIALS FOR THEIR MAGNETIC PROPERTIES (ceramics based on ferrites C04B35/26; alloys C22C; {construction of loading coils H01B} ; thermomagnetic devices H01L37/00; loudspeakers, microphones, gramophone pick-ups or like acoustic electromechanical transducers H04R)	자석; 인덕턴스(inductance); 변압기; 자기 특성을 위한 재료의 선택 (페라이트에 기반한 세라믹 C04B35/26; 합금 C22C; {장하 코일의 구성 H01B}; 열자기 장치 H01L37/00; 스피커, 마이크, 축음기 선별 또는 그와 유사한 음향 전자기계 변환기 H04R)
H01G	CAPACITORS; CAPACITORS, RECTIFIERS, DETECTORS, SWITCHING DEVICES OR LIGHT-SENSITIVE DEVICES, OF THE ELECTROLYTIC TYPE (selection of specified materials as dielectric H01B3/00; capacitors with potential-jump or surface barrier H01L29/00)	콘덴서; 전해용 콘덴서, 정류기, 검파기, 개폐장치 또는 감광장치 (유전체로서의 특정재료의 선택 H01B3/00; 전위 점프 또는 표면 장벽이 있는 콘덴서 H01L 29/00)
H01H	ELECTRIC SWITCHES; RELAYS; SELECTORS; EMERGENCY PROTECTIVE DEVICES (contact cables H01B7/10; electrolytic	전기적스위치; 계전기; 셀렉터 (selector); 비상보호장치(접촉케이블 H01B7/10; 전해자기차단장치 H01G9/18; 비상보호회로장치

	self-interrupters H01G9/18; emergency protective circuit arrangements H02H; switching by electronic means without contact-making H03K17/00)	H02H; 무접점 전자장치에 의한 스위칭 H03K17/00)
H01J	ELECTRIC DISCHARGE TUBES OR DISCHARGE LAMPS (spark-gaps H01T; arc lamps with consumable electrodes H05B; particle accelerators H05H)	방전관 또는 방전램프(스파크-갭 H01T; 소모하는 전극을 갖는 아크 램프 H05B; 입자가속기 H05H)
H01K	ELECTRIC INCANDESCENT LAMPS (details, apparatus or processes for manufacture applicable to both discharge devices and incandescent lamps H01J; light sources using a combination of incandescent and other types of light generation H01J61/96, H05B35/00)	전기 백열등 (방사장치 및 백열등 모두에 적용 가능한 제조를 위한 부품 또는 장치 또는 과정 H01J; 백열 및 다른 형태의 빛 발생 장치의 결합을 이용한 광원 H01J61/96, H05B35/00)
H01L	SEMICONDUCTOR DEVICES; ELECTRIC SOLID STATE DEVICES NOT OTHERWISE PROVIDED FOR (use of semiconductor devices for measuring G01; resistors in general H01C; magnets, inductors, transformers H01F; capacitors in general H01G; electrolytic devices H01G9/00; batteries, accumulators H01M; waveguides, resonators, or lines of the waveguide type H01P; line connectors, current collectors H01R; stimulated-emission devices H01S; electromechanical resonators H03H; loudspeakers, microphones, gramophone	반도체 장치; 달리 제공되지 않는 전기 고체 장치 (G01 측정을 위한 반도체 장치의 사용, 일반 H01C 저항, 자석, 인덕터, 변압기 H01F, 커패시터 일반 H01G, 전해 장치 H01G9/00, 배터리, 축전지 H01M, 도파관, 공진기, 도파관 유형 H01P의 라인; 라인 커넥터, 집전체 H01R; 유도 방출 장치 H01S; 전기 기계 공진기 H03H; 확성기, 마이크, 축음기 픽업 또는 이와 유사한 음향 전기 기계 변환기 H04R; 일반 전기 광원 H05B; 인쇄 회로, 하이브리드 전기 장치의 회로, 케이싱 또는 구조적 세부사항, 전기 부품 집합의 제조 H05K, 특정 용도를 갖는 회로내

	pick-ups or like acoustic electromechanical transducers H04R; electric light sources in general H05B; printed circuits, hybrid circuits, casings or constructional details of electrical apparatus, manufacture of assemblages of electrical components H05K; use of semiconductor devices in circuits having a particular application, see the subclass for the application)	반도체 장치의 사용, 해당 용도의 하위그룹 참조)
H01M	PROCESSES OR MEANS, e.g. BATTERIES, FOR THE DIRECT CONVERSION OF CHEMICAL ENERGY INTO ELECTRICAL ENERGY	방법 또는 수단, 예. 전지. 화학적 에너지의 전기 에너지로의 직접 변환을 위한 것
H01P	WAVEGUIDES; RESONATORS, LINES, OR OTHER DEVICES OF THE WAVEGUIDE TYPE (operating at optical frequencies G02B)	도파관; 공진기, 선로 또는 도파관 형의 다른 장치 (광 주파수에서 작동하는 G02B)
H01Q	ANTENNAS, i.e. RADIO AERIALS (radiators or antennas for microwave heating H05B6/72)	안테나, 즉. 무선 공중선 (마이크로파 가열용 라디에이터 또는 안테나 H05B 6/72)
H01R	ELECTRICALLY-CONDUCTIVE CONNECTIONS; STRUCTURAL ASSOCIATIONS OF A PLURALITY OF MUTUALLY-INSULATED ELECTRICAL CONNECTING ELEMENTS; COUPLING DEVICES; CURRENT COLLECTORS	전기적으로 전도성있는 연결; 상호 절연된 전기적 연결 소자들의 구조적 결합; 커플링 장치; 전류 수집기
H01S	DEVICES USING THE PROCESS OF LIGHT AMPLIFICATION BY STIMULATED EMISSION OF	빛을 증폭시키거나 발생시키기 위해 방사 [레이저]의 유도된 방출에 의한 빛의 진행 과정을 이용하는

	RADIATION [LASER] TO AMPLIFY OR GENERATE LIGHT; DEVICES USING STIMULATED EMISSION OF ELECTROMAGNETIC RADIATION IN WAVE RANGES OTHER THAN OPTICAL	장치; 광파 이외의 파동 범위에서 전자파 방사의 유도된 방사를 사용하는 기기

데이터셋 #2는 CPC H01 계열의 서브클래스(H01B, H01C, H01F, H01G, H01G, H01H, H01J, H01K, H01L, H01M, H01P, H01Q, H01R, H01S) 분류체계로 분류된 특허데이터를 사용한다. 총 14개의 분류체계로 구성되었으며 분류체계에 포함된 미국 등록특허를 출원일 순으로 각 200개씩 총 2,800개의 특허로 구성하였다. 훈련데이터와 테스트 데이터는 8:2로 분리하였다. 분리기준은 최신 출원일 순으로 분리하였으며 데이터셋 #1 과 동일한 기준을 적용하였다.

[표 3-4] 데이터셋 #2의 구성

CPC 코드	훈련 데이터	검증 데이터	테스트 데이터	합계
H01B	144	16	40	200
H01C	144	16	40	200
H01F	144	16	40	200
H01G	144	16	40	200
H01H	144	16	40	200
H01J	144	16	40	200
H01K	144	16	40	200
H01L	144	16	40	200
H01M	144	16	40	200
H01P	144	16	40	200
H01Q	144	16	40	200
H01R	144	16	40	200
H01S	144	16	40	200
H01T	144	16	40	200
합계	2,016	224	560	2,800

훈련데이터는 딥러닝 분류모델을 훈련하는데 사용되며 검증데이터는 모델의 훈련 진행상황을 검증하는데 사용된다. 훈련이 끝난 후 모델의 성능을 측정하기 위해 테스트데이터를 사용한다.

3) 데이터 셋 #3

데이터셋 #3은 CPC C01 계열과 CPC H01 계열 데이터를 결합하여 총 19개의 분류체계로 구성된다.

분류체계별로 200건의 미국등록 특허가 포함하여 총 3,800건의 데이터로 구성되어 있다. 훈련데이터와 테스트 데이터는 8:2로 분리하여 각각 3,040건 760건으로 구성하였다.

4) 데이터 셋 요약

본 연구에서는 특허의 필드 중 제목, 요약, 대표 청구항, 전체 청구항을 사용한다. 제목 + 요약, 제목 + 요약 + 대표청구항, 제목 + 요약 + 전체청구항의 3가지 조합을 각 모델의 입력으로 사용한다. 이를 각각 입력 #1(제목 , 요약), 입력 #2(제목, 요약, 대표청구항) 그리고 입력 #3(제목, 요약, 전체정구항)으로 명명하였다.

최종적으로 CPC 분류 체계의 미국 등록특허를 이용하여 3가지 데이터 셋을 구성하였다. 각 데이터셋은 각 딥러닝 모델을 훈련하고 테스트 하여 모델의 성능을 측정하는데 사용한다.

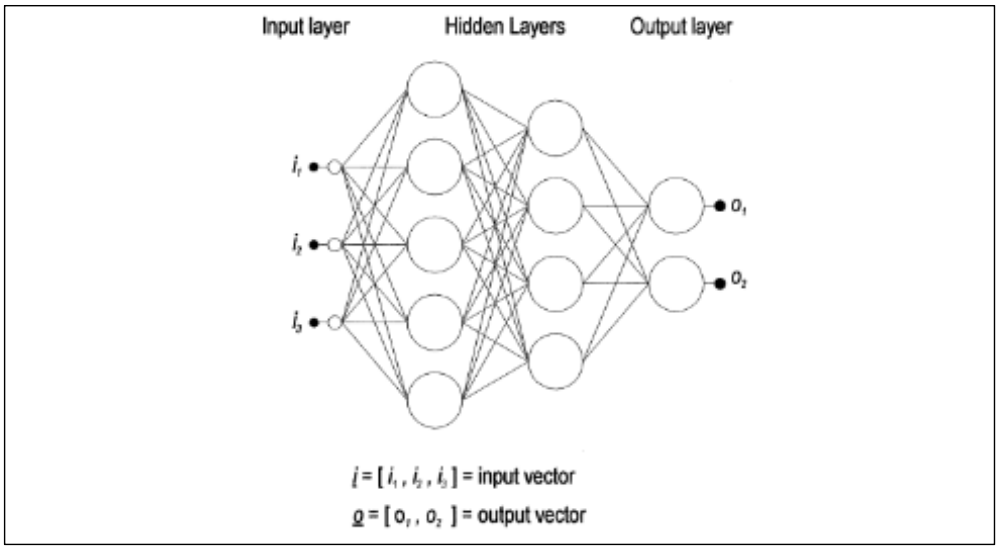
[표 3-5] 데이터셋 요약

	분류체계	훈련데이터	검증데이터	테스트데이터	합계
데이터셋 #1	5	720	80	200	1,000
데이터셋 #2	14	2,016	224	560	2,800
데이터셋 #3	19	2,736	304	760	3,800

제 2 절 Multi Layer Perceptron(MLP) 알고리즘을 사용한 특허분류

퍼셉트론(Perceptron)은 인공신경망의 한 종류로서, 1957년에 코넬 항공연구소(Cornell Aeronautical Lab)의 Frank Rosenblatt에 의해 고안되었다. 퍼셉트론은 가장 간단한 형태의 순방향(Feedforward) 네트워크의 선형분류기라고 할 수 있다. 퍼셉트론의 동작방식은 각 노드의 가중치와 입력치를 곱한 것을 모두 합한 값이 활성화 함수에 의해 판단되는데, 그 값이 임계치보다 크면 뉴런이 활성화 되고 결과 값 1을 출력한다. 뉴런이 활성화 되지 않으면 결과 값 -1을 출력한다. (Van Der Malsburg C., 1962)

퍼셉트론의 기본 구조는 입력층(Input Layer), 은닉층(Hidden Layer), 출력층(Output Layer)으로 구성되어 있다. 이때 은닉층의 개수가 2개 이상인 경우를 MLP(Multi Layer Perceptron) 라고 한다. 은닉층의 출력결과가 다음 은닉층의 입력으로 사용되며 이 과정이 은닉층의 개수 만큼 반복된다.



[그림 3-2] 2개의 은닉층을 가진 Multi Layer Perceptron(Gardner et al., 1998)

MLP에서 입력층은 vector의 형태로 입력되며 마찬가지로 출력층도 vector의 형태로 출력된다. 출력층의 벡터 길이는 분류문제에서는 분류체계의 개수와 동일하며 입력층의 입력은 텍스트를 다양한 방식으로 벡터로 변환한 것이다.

요약하자면 MLP 알고리즘를 이용해서 분류문제를 해결하는 과정은 분류체계와 출력층을 적용시키고, 데이터와 입력층을 적용하고 은닉층의 개수 및 다양한 변수들을 변동시키며 데이터에 적응시키는 과정이라고 할 수 있다.

본 연구에서 MLP 알고리즘을 사용하는 이유는 사람에 의해 수행되는 특허분류에 있어 문장 속에 존재하는 단어의 존재를 판단하여 분류를 판단하는 방법을 묘사하기 위해서이다.

1) 특허분류를 위한 MLP 알고리즘 분류모델의 구성

가) 데이터 전처리

MLP 의 입력층은 벡터로 이루어진 데이터를 받아들인다. 특허문헌을 입력층에서 받아들일 수 있는 형태로 변환하기 위해 각 데이터셋에 포함된 텍스트를 벡터로 변환해야 한다. 본 연구에서는 특허문헌의 여러 구성 요소중 제목, 요약, 대표 청구항, 전체 청구항을 사용한다. 모두 텍스트로 이루어진 요소이며 특허문서의 분류에 적합한 구성요소로 판단된다.

이러한 텍스트를 벡터로 변환하기 위해서는 B.O.W.(Bag of Words) 방식을 사용한다. B.O.W.는 텍스트에 포함된 단어의 순서는 고려되지 않으며, 단어들에 임의의 아이디를 부여하고 이를 이용해 텍스트를 수치화하는 표현 방법이다.

특허문서 분류용 BOW를 만들기 위해서는 전체 훈련데이터에 포함된 단어들을 모두 추출해서 각 단어에 고유한 정수 인덱스를 부여한다, 각 특허문헌에 포함된 단어의 등장 횟수를 기록하는 벡터를 만든다. 이러한 과정을 거쳐서 특허 문헌 한건은 하나의 벡터로 변환하게 된다.

해당 연구에서 사용되는 세 가지 데이터셋의 BOW는 각각 만들어지며 각

각의 데이터셋에 포함된 BOW의 크기는 입력층의 크기가 된다.

[표 3-6] 데이터셋 별 입력층과 출력층의 벡터크기

	특허문헌 수	입력층 크기	출력층 크기(분류체계 수)
데이터셋 #1	1,000	1,573	5
데이터셋 #2	2,800	7,449	9
데이터셋 #3	3,800	9,549	14

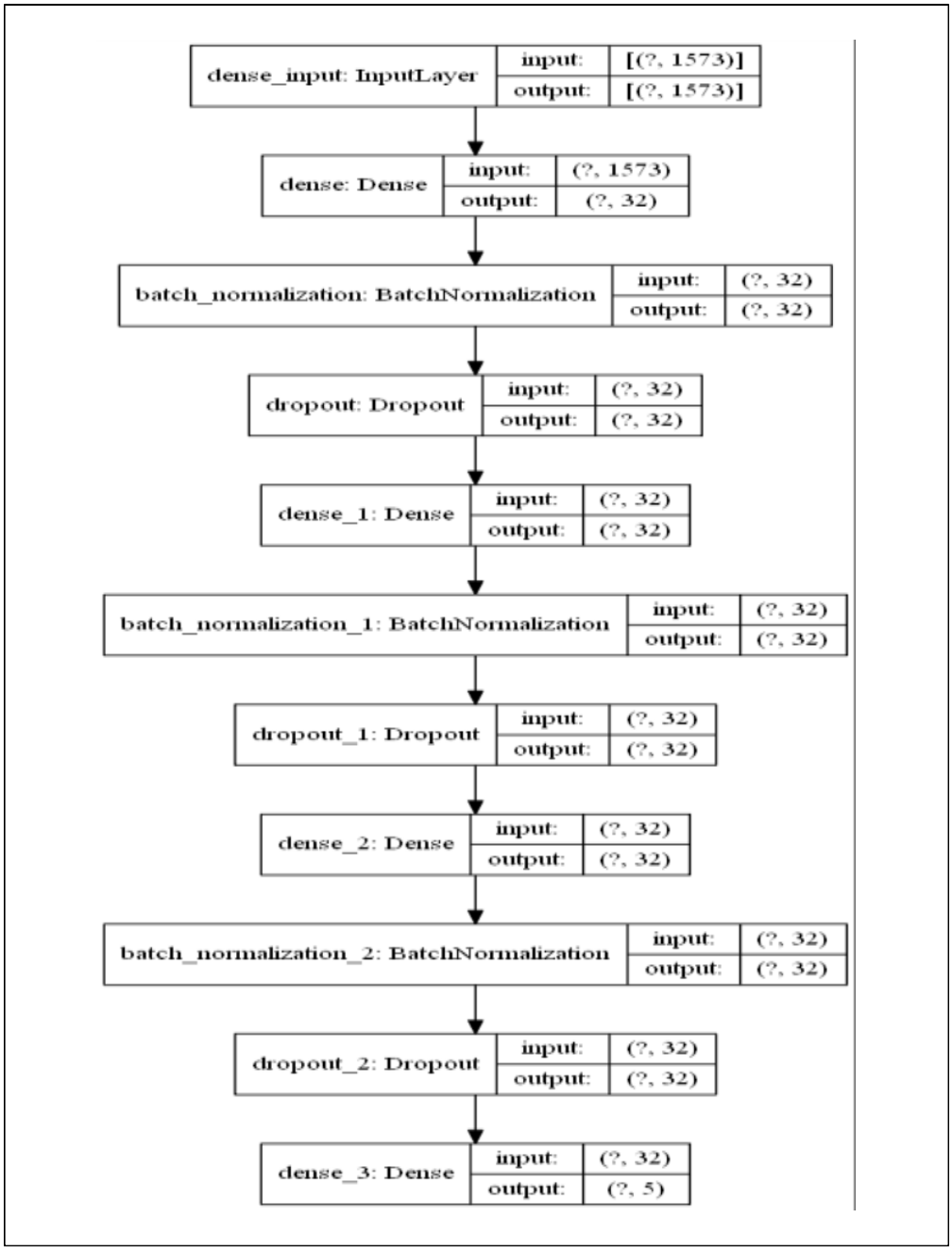
나) 특허분류용 MLP 기본 분류모델

본 연구에서 사용한 MLP의 기본 분류모델은 데이터셋에 따라 입력층과 출력층의 노드 개수가 고정된 값을 가진다. 은닉층은 층의 개수와 각 층의 노드 개수를 변화시켜 여러개의 조합에 대해서 분류모델을 생성한다. 은닉층의 출력을 다음 은닉층의 출력으로 입력하기 전에 배치 정규화(Batch Normalization) 층을 통과시킨 후 활성화 함수를 적용하여 다음 은닉층의 입력값으로 사용한다. 과적합을 줄이기 위해 각 층에 드롭아웃(Dropout) 층을 삽입하여 사용하였다.

[그림 3-3]은 데이터셋 #1 에 대한 MLP 분류모델의 예시를 보여주고 있다. 입력층의 노드 개수는 동일한 1,573 이며, 출력층의 노드 개수는 분류체계의 개수와 동일한 5임을 확인할 수 있다. 또한 은닉층은 총 3개의 층이 있으며 각 층마다 각각 32개의 노드로 구성되어 있음을 확인 할 수 있다.

본 분류모델에 사용된 배치 정규화는 활성화 함수를 통과한 값 또는 층을 통과한 출력값을 정규화 하는 작업이다 (Ioffe et al., 2015).

배치정규화는 학습속도의 개선, 가중치 초깃값 선택의 의존성이 적어지며, 과적합의 위험을 줄여줄 수 있다고 알려져 있다.



[그림 3-3] 데이터셋 #1 에 적용된 MLP 모델의 구조 예시 (은닉층의 개수 : 3, 은닉층의 노드 수 : 32)

2) 매개변수의 구성

MLP 분류모델를 훈련시키는데 주요한 매개변수에는 은닉층의 개수, 은닉층의 노드 수, 학습률, 배치사이즈 등이 있다.

본 연구에서는 [표 3-7]에서 보이는 것처럼 각 매개변수의 종류에 따라 여러 값을 적용하여 총 144가지의 조합으로 MLP 분류모델을 훈련하였다.

[표 3-7] MLP 분류모델의 매개변수 종류 및 적용값

매개변수 종류	적용값 집합
입력셋	입력#1, 입력#2, 입력#3
은닉층 수	1, 2, 3
은닉층의 노드 수	32, 64
학습률	$1e^{-3}$, $1e^{-4}$
배치사이즈	8, 16
드롭아웃 비율	0.2, 0.3

3) 실험 조건

본 연구에서는 Python 3.8 버전, Tensorflow 2.3.0 버전을 사용하였고, Keras 2.4 버전을 이용하여 프로그램을 작성하였다.

각 분류모델은 총 150 에폭씩 훈련하며 조기종료 기법을 적용하였다. 조기종료의 조건은 매 에폭마다 검증데이터의 손실값(loss)를 측정하여 10 에폭이 지나도 손실값이 감소하지 않으면 종료시키도록 하였다.

하나의 데이터 셋에 사용되는 분류모델은 입력셋(3), 하이퍼파라미터 조합(48) 적용하여 총 144개 분류모델에 대해서 훈련시켰으며 이 분류모델 중 검증데이터의 손실 값이 가장 낮은 분류모델, 검증 데이터의 정확도가 가장 높은 분류모델 그리고 훈련데이터의 손실값과 검증데이터의 손실값의 차가 가장 낮은 분류모델을 각각 선택하였다. 즉, 각 데이터셋 당 3개의 최적 MLP 분류모델을 선정하여 테스트 데이터에 대한 정확도를 측정하였다.

4) 실험 결과

가) 데이터셋 #1 에 MLP 분류모델을 적용한 분류 결과

데이터셋 #1 에 대해서 입력데이터와 매개변수의 조합으로 총 144개의 MLP 분류모델을 훈련시켰다.

총 720건의 특허문서를 훈련데이터로 사용하였고 80건의 특허문서를 검증데이터로 사용하였다. 총 144건의 모델의 훈련결과 중 검증 정확도가 가장 높은 모델, 검증 손실값이 가장 낮은 모델 그리고 훈련 손실값과 검증 손실값의 차가 가장 낮은 3개의 모델을 선정하였다.

이 모델을 각각 MLP#1-1, MLP#1-2, MLP#1-3으로 명명하였으며 이 모델의 하이퍼파라미터 값 및 검증데이터의 정확도와 손실값은 [표 3-8]에 나타내었다.

MLP#1-1 분류모델의 검증 정확도 0.850이라는 값은 총 80건의 검증데이터 중 68개의 특허문서에 대해 정답분류체계와 동일한 분류체계를 예측하였고 12건의 특허문서에 대해서는 정답분류체계와 다른 분류체계를 예측했다는 것을 나타낸다.

[표 3-8] 데이터셋 #1 의 검증 데이터에 최적인 MLP 분류모델

	입력	은닉층	노드	학습률	배치	드롭아웃	에폭	검증 정확도	검증 손실값
MLP#1-1	입력#1	1	64	1e-4	8	0.2	43	0.850	0.5774
MLP#1-2	입력#1	2	32	1e-4	16	0.3	70	0.863	0.6121
MLP#1-3	입력#2	1	32	1e-3	16	0.2	18	0.863	0.6750

각 분류모델에 대한 데이터셋 #1 의 테스트 데이터 200건에 대한 분류결과는 5개의 분류체계의 각 항목에 대해서 정밀도(precision), 재현율(recall)

그리고 f1-score를 측정하였으며, 전체 데이터에 대한 정확도(accuracy), 정밀도 재현율 그리고 f1-score를 측정하였다.

데이터셋 #1의 테스트데이터에 대한 정확도를 비교하면 MLP#1-2 분류모델의 정확도가 79.50% 였으며 MLP#1-1 과 MLP#1-3 에 비해 각각 2%, 2.5% 높은 정확도를 나타냈다.

[표 3-9] MLP#1-1 모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.6579	0.6250	0.6410	40
C01C	0.8810	0.9250	0.9024	40
C01D	0.7500	0.8250	0.7857	40
C01F	0.8205	0.8000	0.8101	40
C01G	0.7568	0.7000	0.7273	40
accuracy			0.7750	200
macro avg	0.7732	0.7750	0.7733	200

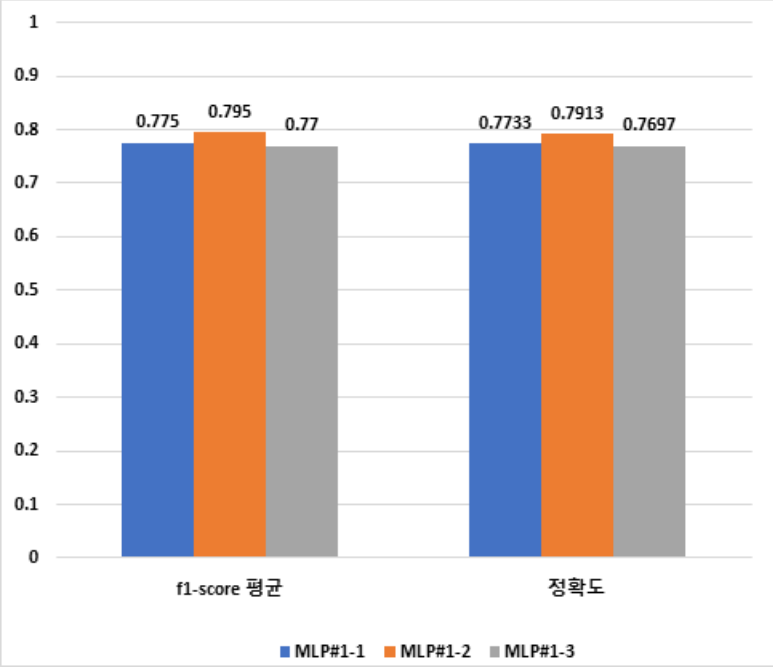
[표 3-10] MLP#1-2 모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.7742	0.6000	0.6761	40
C01C	0.9268	0.9500	0.9383	40
C01D	0.7708	0.9250	0.8409	40
C01F	0.8205	0.8000	0.8101	40
C01G	0.6829	0.7000	0.6914	40
accuracy			0.7950	200
macro avg	0.7951	0.7950	0.7913	200

[표 3-11] MLP#1-3 모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.6667	0.6000	0.6316	40
C01C	0.8649	0.8000	0.8312	40
C01D	0.8182	0.9000	0.8571	40
C01F	0.8684	0.8250	0.8462	40

C01G	0.6444	0.7250	0.6824	40
accuracy			0.7700	200
macro avg	0.7725	0.7700	0.7697	200



[그림 3-4] 데이터셋 #1 테스트데이터 MLP 분류모델 f1-score와 정확도

나) 데이터셋 #2 에 MLP 모델을 적용한 분류 결과

데이터셋 #2 에 대해서 입력데이터와 매개변수 조합으로 총 144개의 MLP 분류모델을 훈련시켰다.

총 2,016건의 특허문서를 훈련데이터로 사용하였고 224건의 특허문서를 검증데이터로 사용하였다. 총 144건의 모델의 훈련결과 중 검증 정확도가 가장 높은 모델, 검증 손실값이 가장 낮은 모델 그리고 훈련 손실값과 검증 손

실값의 차가 가장 낮은 3개의 모델을 선정하였다.

이 모델을 각각 MLP#2-1, MLP#2-2, MLP#2-3으로 명명하였으며 이 모델에 적용된 매개변수 값은 [표 3-12]에 나타내었다.

[표 3-12] 데이터셋 #2 의 검증 데이터에 최적인 MLP 분류모델

	입력	은닉층	노드	학습률	배치	드롭아웃	에폭	검증 정확도	검증 손실값
MLP#2-1	입력#3	1	64	$1e^{-4}$	16	0.3	41	0.857	0.4861
MLP#2-2	입력#3	1	32	$1e^{-4}$	16	0.2	50	0.862	0.5012
MLP#2-3	입력#3	1	32	$1e^{-3}$	8	0.2	6	0.871	0.4852

데이터셋 #2의 테스트데이터에 대한 정확도를 비교하면 MLP#2-1 분류모델의 정확도가 83.39% 였으며 MLP#2-3 과 MLP#2-2 에 비해 각각 0.35%, 0.71% 높은 정확도를 나타냈다.

[표 3-13] MLP#2-1 분류모델의 테스트 데이터 분류 결과

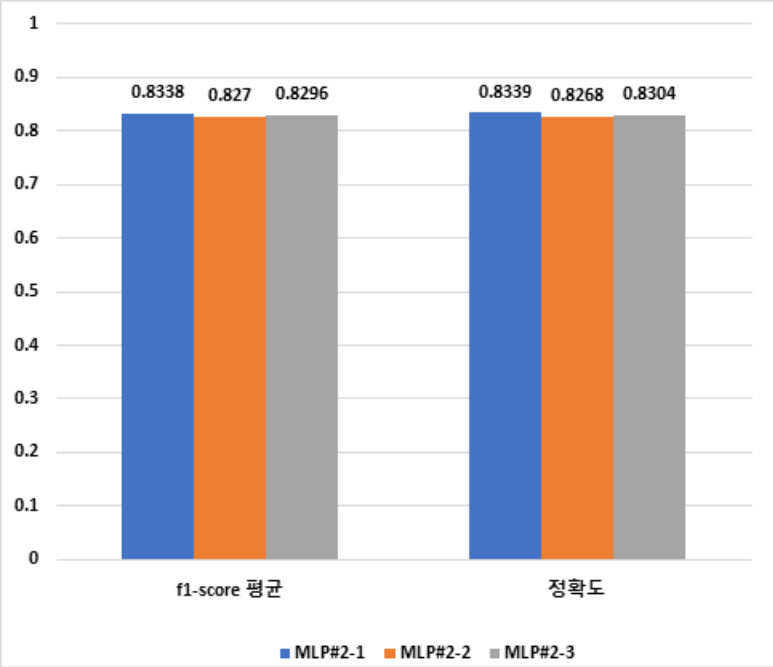
	precision	recall	f1-score	support
H01B	0.8158	0.7750	0.7949	40
H01C	0.7727	0.8500	0.8095	40
H01F	0.9091	0.7500	0.8219	40
H01G	0.7619	0.8000	0.7805	40
H01H	0.8043	0.9250	0.8605	40
H01J	0.7917	0.9500	0.8636	40
H01K	0.9730	0.9000	0.9351	40
H01L	0.7174	0.8250	0.7674	40
H01M	0.8421	0.8000	0.8205	40
H01P	0.8667	0.6500	0.7429	40
H01Q	0.8140	0.8750	0.8434	40
H01R	0.7750	0.7750	0.7750	40
H01S	0.9474	0.9000	0.9231	40
H01T	0.9730	0.9000	0.9351	40
accuracy			0.8339	560
macro avg	0.8403	0.8339	0.8338	560

[표 3-14] MLP#2-2 모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.7692	0.7500	0.7595	40
H01C	0.7727	0.8500	0.8095	40
H01F	0.9394	0.7750	0.8493	40
H01G	0.7619	0.8000	0.7805	40
H01H	0.7600	0.9500	0.8444	40
H01J	0.7551	0.9250	0.8315	40
H01K	0.9730	0.9000	0.9351	40
H01L	0.7111	0.8000	0.7529	40
H01M	0.8611	0.7750	0.8158	40
H01P	0.8966	0.6500	0.7536	40
H01Q	0.8182	0.9000	0.8571	40
H01R	0.7692	0.7500	0.7595	40
H01S	0.9444	0.8500	0.8947	40
H01T	0.9730	0.9000	0.9351	40
accuracy			0.8268	560
macro avg	0.8361	0.8268	0.8270	560

[표 3-15] MLP#2-3 모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.7632	0.7250	0.7436	40
H01C	0.7660	0.9000	0.8276	40
H01F	0.8611	0.7750	0.8158	40
H01G	0.8049	0.8250	0.8148	40
H01H	0.7872	0.9250	0.8506	40
H01J	0.8095	0.8500	0.8293	40
H01K	0.9474	0.9000	0.9231	40
H01L	0.7500	0.8250	0.7857	40
H01M	0.8718	0.8500	0.8608	40
H01P	0.9259	0.6250	0.7463	40
H01Q	0.7955	0.8750	0.8333	40
H01R	0.8378	0.7750	0.8052	40
H01S	0.8140	0.8750	0.8434	40
H01T	0.9730	0.9000	0.9351	40
accuracy			0.8304	560
macro avg	0.8362	0.8304	0.8296	560



[그림 3-5] 데이터셋 #2 테스트데이터 MLP 분류모델 f1-score와 정확도

다) 데이터셋 #3 에 MLP 분류모델을 적용한 분류 결과

데이터셋 #3 에 대해서 입력데이터와 매개변수의 조합으로 총 144개의 MLP 분류모델을 훈련시켰다.

총 3,040건의 특허문서를 훈련데이터로 사용하였고 760건의 특허문서를 검증데이터로 사용하였다. 총 144개의 모델의 훈련결과 중 검증 정확도가 가장 높은 모델, 검증 손실값이 가장 낮은 모델 그리고 훈련 손실값과 검증 손실값의 차가 가장 낮은 3개의 모델을 선정하였다.

이 모델을 각각 MLP#3-1, MLP#3-2, MLP#3-3으로 명명하였으며 이 모델에 적용된 매개변수 값은 [표 3-16]에 나타내었다.

[표 3-16] 데이터셋 #3 의 검증 데이터에 최적인 MLP 분류모델

	입력	은닉층	노드	학습률	배치	드롭아웃	에폭	검증 정확도	검증 손실값
MLP#3-1	입력#1	1	32	$1e^{-4}$	8	0.3	39	0.836	0.7151
MLP#3-2	입력#1	1	64	$1e^{-4}$	16	0.3	47	0.839	0.6635
MLP#3-3	입력#1	1	64	$1e^{-4}$	16	0.3	36	0.836	0.6551

데이터셋 #3의 테스트데이터에 대한 정확도를 비교하면 MLP#3-2 모델의 정확도가 81.18% 였으며 MLP#3-3 과 MLP#3-2 에 비해 각각 0.52%, 0.92% 높은 정확도를 나타냈다.

[표 3-17] MLP#3-1 모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.6944	0.6250	0.6579	40
C01C	0.8864	0.9750	0.9286	40
C01D	0.7347	0.9000	0.8090	40
C01F	0.8889	0.8000	0.8421	40
C01G	0.6744	0.7250	0.6988	40
H01B	0.7368	0.7000	0.7179	40
H01C	0.8095	0.8500	0.8293	40
H01F	0.8919	0.8250	0.8571	40
H01G	0.7436	0.7250	0.7342	40
H01H	0.7609	0.8750	0.8140	40
H01J	0.8095	0.8500	0.8293	40
H01K	0.9706	0.8250	0.8919	40
H01L	0.7234	0.8500	0.7816	40
H01M	0.8125	0.6500	0.7222	40
H01P	0.7812	0.6250	0.6944	40
H01Q	0.8085	0.9500	0.8736	40
H01R	0.8205	0.8000	0.8101	40
H01S	0.9167	0.8250	0.8684	40
H01T	0.9268	0.9500	0.9383	40
accuracy			0.8066	760
macro avg	0.8101	0.8066	0.8052	760

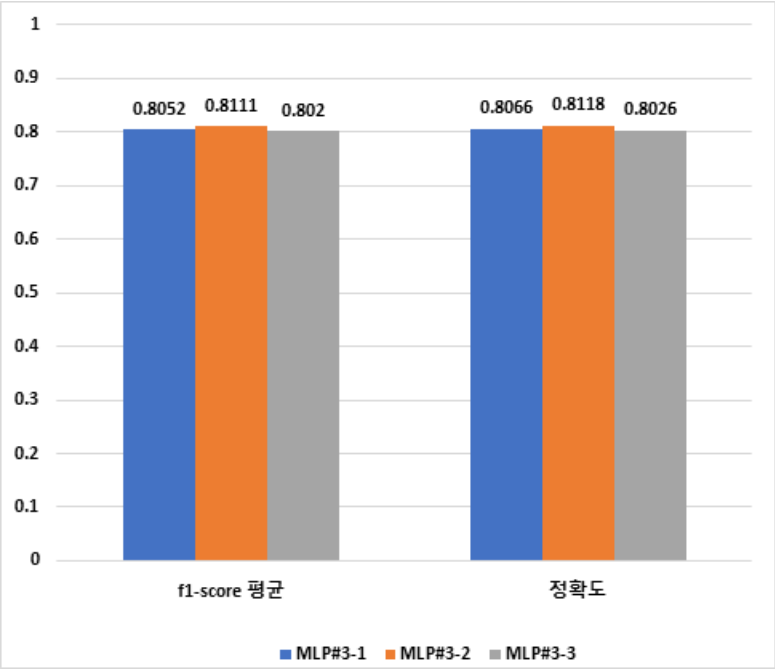
[표 3-18] MLP#3-2 모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.7027	0.6500	0.6753	40

C01C	0.9268	0.9500	0.9383	40
C01D	0.7500	0.9000	0.8182	40
C01F	0.7561	0.7750	0.7654	40
C01G	0.6923	0.6750	0.6835	40
H01B	0.7805	0.8000	0.7901	40
H01C	0.8537	0.8750	0.8642	40
H01F	0.9118	0.7750	0.8378	40
H01G	0.7949	0.7750	0.7848	40
H01H	0.7551	0.9250	0.8315	40
H01J	0.8293	0.8500	0.8395	40
H01K	0.9167	0.8250	0.8684	40
H01L	0.7021	0.8250	0.7586	40
H01M	0.7500	0.6750	0.7105	40
H01P	0.7941	0.6750	0.7297	40
H01Q	0.8222	0.9250	0.8706	40
H01R	0.8857	0.7750	0.8267	40
H01S	0.9429	0.8250	0.8800	40
H01T	0.9268	0.9500	0.9383	40
accuracy			0.8118	760
macro avg	0.8155	0.8118	0.8111	760

[표 3-19] MLP#3-3 모델의 테스트 데이터 분류 결과

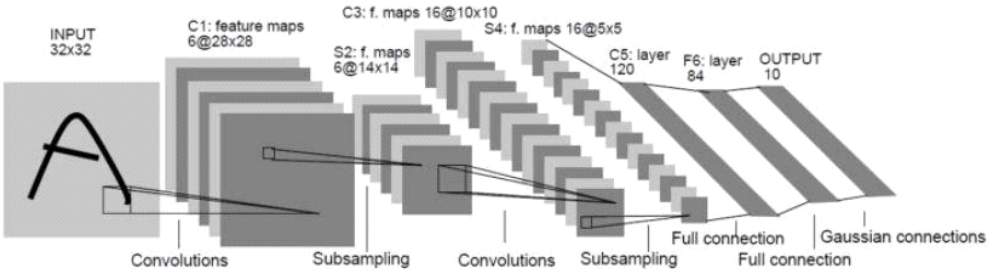
	precision	recall	f1-score	support
C01B	0.6667	0.6500	0.6582	40
C01C	0.9268	0.9500	0.9383	40
C01D	0.7347	0.9000	0.8090	40
C01F	0.7561	0.7750	0.7654	40
C01G	0.6579	0.6250	0.6410	40
H01B	0.7561	0.7750	0.7654	40
H01C	0.8537	0.8750	0.8642	40
H01F	0.9118	0.7750	0.8378	40
H01G	0.7632	0.7250	0.7436	40
H01H	0.7500	0.9000	0.8182	40
H01J	0.8293	0.8500	0.8395	40
H01K	0.9429	0.8250	0.8800	40
H01L	0.6875	0.8250	0.7500	40
H01M	0.7429	0.6500	0.6933	40
H01P	0.8182	0.6750	0.7397	40
H01Q	0.8222	0.9250	0.8706	40
H01R	0.8378	0.7750	0.8052	40
H01S	0.9429	0.8250	0.8800	40
H01T	0.9268	0.9500	0.9383	40
accuracy			0.8026	760
macro avg	0.8067	0.8026	0.8020	760



[그림 3-6] 데이터셋 #3 테스트데이터 MLP 분류모델 f1-score와 정확도

제 3 절 Convolutional Neural Network 알고리즘을 사용한 특허분류

시각 피질의 구조에 대한 연구(Hubel, 1959)를 기반으로 고수준 뉴런이 이웃한 저수준 뉴런의 출력에 기반 한다는 아이디어가 나왔다. 이러한 아이디어는 합성곱 신경망으로 점점 진화하였고, 이는 수표에 쓰인 손글씨 숫자를 인식하는데 널리 사용된 LeNet-5 구조에 사용되었으며 오늘날의 CNN 구조의 기초를 확립하였다.(LeCun et al., 1998)



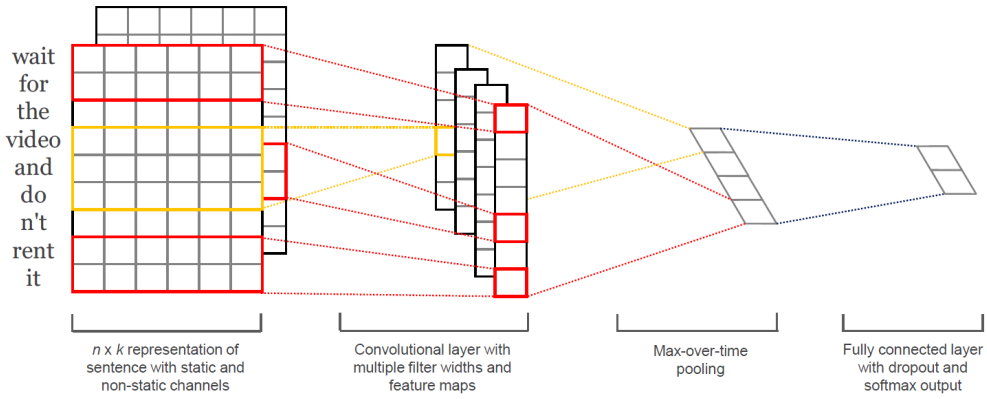
[그림 3-7] Architecture of LeNet-5, a convolutional NN, here used for digits recognition. Each plane is a feature map, i.e., a set of units whose weights are constrained to be identical (LeCun et al., 1998)

일반적으로 CNN은 구조의 특성상 이미지 인식에 효과를 발휘한다고 인식되었으나 문장의 인식과 분류에도 사용할 수 있으며 효과적인 성능을 낸다는 연구가 발표되었다. (Kim, 2014)

해당 연구에서는 단어를 벡터로 변환하기 위해 Word2Vec 이나 Glove 등의 분산표현(distributed representation)을 사용하였다. 단어들로 이루어진 문장은 행렬로 표시될 수 있으며 이는 흑백 이미지와 동일한 차원을 가진다. 행렬로 변환된 문장을 합성곱 층을 통과시키고 서브샘플링 방법을 사용하여 최종적으로 완전 연결층을 통과시켜 문장을 분류하는 구조이다.

본 연구의 CNN을 이용한 특허문서 분류에서는 한 번에 인식하는 단어의

개수를 3, 4, 5개씩 개별적으로 인식하는 합성곱층을 만들고 이를 각 채널로 인식하여 3개의 채널을 사용하는 CNN 구조로 특허문서 분류를 수행하였다.



[그림 3-8] Model architecture with two channels for an example sentence(Kim, 2014)

본 연구에서 특허분류를 위해 CNN 모델을 사용하는 이유는 사람에 의해 수행되는 특허분류에 있어 문장 속에 존재하는 연속된 단어의 구성으로 분류를 판단하는 방법을 묘사하기 위해서이다.

1) 특허분류를 위한 CNN 분류모델의 구성

가) 데이터 전처리

CNN의 입력층은 행렬 또는 텐서로 이루어진 데이터를 받아들인다. 문장에 포함된 단어를 벡터로 변환하여 이를 행렬로 변환하므로 본 연구의 특허문장은 행렬로 변환된다.

단어를 벡터로 변환하기 위한 분산표현으로 Word2Vec을 사용하였으며 벡터의 사이즈는 100으로 정의하였다. 만약 20개의 단어로 이루어진 문장이라고 하면 20 X 100 의 구조를 가지는 행렬로 변환된다.

문장에서 특수기호와 구두점 및 숫자는 제외하고 영문으로 이루어진 단어

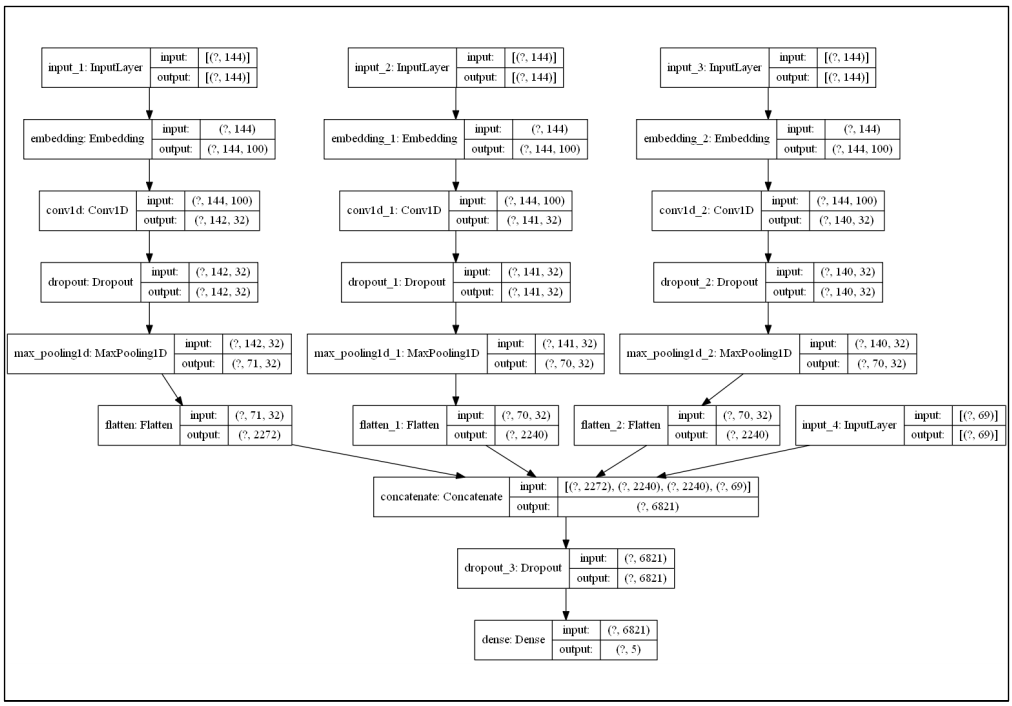
만으로 전처리 후 이를 행렬로 변환하는 방법을 사용하였다.

나) 특허분류용 CNN 분류모델

본 연구에서 설계한 CNN 분류모델은 [그림 3-9] 과 같은 구조를 가진다. 3개의 채널로 구성된 멀티채널 CNN이며 각 채널은 커널의 크기가 각각 3, 4, 5 인 합성곱 층으로 구성된다.

각 합성곱 층은 드롭아웃과 맥스 풀링층을 거친 후 행렬의 각 단위를 이어 붙이는 플래튼(Flatten) 층을 거치게 된다. 이를 통해 벡터로 만들어진 각 층의 출력값을 모두 연결하게 된다. 그리고 이를 완전 연결층으로 변환하여 최종 출력층으로 전달하게 된다.

이러한 기본 구조를 바탕으로 필터의 개수와 드롭아웃 비율 등을 변형시켜서 다양한 구조의 모델을 생성하였다.



[그림 3-9] 데이터셋 #1 에 적용된 CNN 분류모델의 구조 예시

2) 매개변수의 구성

[표 3-20] CNN 분류 모델의 매개변수 종류 및 적용값

매개변수 종류	적용값
입력셋	입력#1, 입력#2, 입력#3
필터수	32, 64
학습률	1e-3, 1e-4
배치사이즈	8, 16
드롭아웃 비율	0.2, 0.3, 0.5

3) 실험조건

본 연구에서는 Python 3.8 버전, Tensorflow 2.3.0 버전을 사용하였고,

Keras 2.4 버전을 이용하여 프로그램을 작성하였다.

각 모델은 총 150 에폭씩 훈련하며 조기종료 기법을 적용하였다. 조기종료의 조건은 매 에폭마다 검증데이터의 손실값(loss)를 측정하여 10 에폭이 지나도 손실값이 감소하지 않으면 종료시키도록 하였다.

하나의 데이터 셋에 사용되는 모델은 입력셋(3), 하이퍼파라미터 조합(24) 적용하여 총 72개 모델에 대해서 훈련시켰으며 이 모델 중 검증데이터의 손실 값이 가장 낮은 모들, 검증 데이터의 정확도가 가장 높은 모델 그리고 훈련데이터의 손실값과 검증데이터의 손실값의 차가 가장 낮은 모델을 각각 선택하였다. 즉, 각 데이터셋 당 3개의 최적 CNN 분류모델을 선정하여 테스트 데이터에 대한 정확도를 측정하였다.

4) 실험결과

가) 데이터셋 #1에 CNN 분류모델을 적용한 분류결과

[표 3-21] 데이터셋 #1에 최적인 CNN 분류모델

	입력	필터	학습률	배치	드롭아웃	에폭	검증 정확도	검증 손실값
CNN#1-1	입력#1	32	1e-3	8	0.3	23	0.812	0.7724
CNN#1-2	입력#1	32	1e-3	8	0.3	22	0.812	0.9980
CNN#1-3	입력#1	32	1e-3	8	0.3	17	0.825	1.0054

데이터셋 #1의 테스트데이터에 대한 정확도를 비교하면 CNN#1-1 분류모델의 정확도가 75.00% 이며 CNN#1-3 과 CNN#1-2 에 비해 각각 2.5%, 5.23% 높은 정확도를 나타냈다.

[표 3-22] CNN#1-1 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.7568	0.7000	0.7273	40
C01C	0.7907	0.8500	0.8193	40
C01D	0.7805	0.8000	0.7901	40
C01F	0.8056	0.7250	0.7632	40

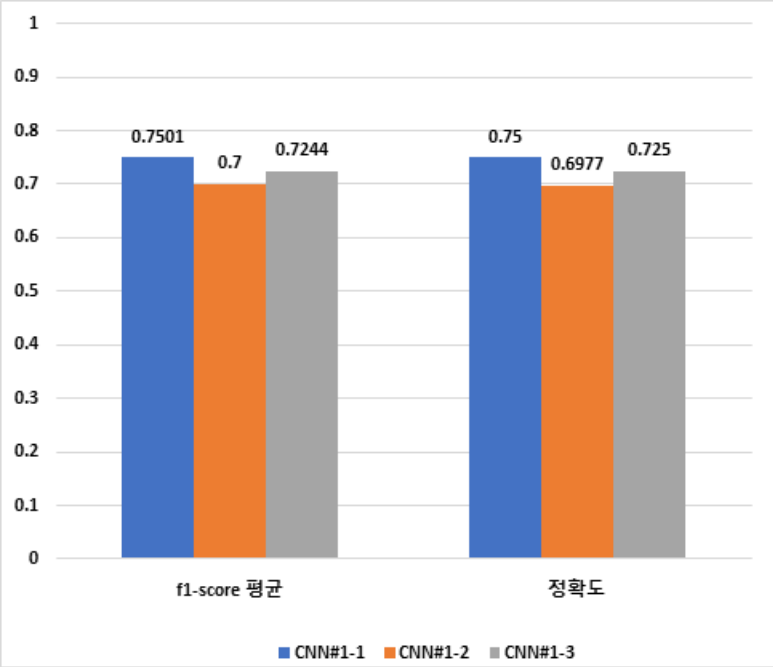
C01G	0.6279	0.6750	0.6506	40
accuracy			0.7500	200
macro avg	0.7523	0.7500	0.7501	200

[표 3-23] CNN#1-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.7333	0.5500	0.6286	40
C01C	0.7907	0.8500	0.8193	40
C01D	0.7111	0.8000	0.7529	40
C01F	0.7105	0.6750	0.6923	40
C01G	0.5682	0.6250	0.5952	40
accuracy			0.7000	200
macro avg	0.7028	0.7000	0.6977	200

[표 3-24] CNN#1-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.6522	0.7500	0.6977	40
C01C	0.8421	0.8000	0.8205	40
C01D	0.6939	0.8500	0.7640	40
C01F	1.0000	0.5250	0.6885	40
C01G	0.6087	0.7000	0.6512	40
accuracy			0.7250	200
macro avg	0.7594	0.7250	0.7244	200



[그림 3-10] 데이터셋#1 테스트데이터 CNN 분류모델 f1-score와 정확도

나) 데이터셋 #2에 CNN 분류모델을 적용한 분류결과

[표 3-25] 데이터셋 #2에 최적인 CNN 분류모델

	입력	필터	학습률	배치	드롭아웃	에폭	검증 정확도	검증 손실값
CNN#2-1	입력#1	32	$1e^{-4}$	8	0.2	61	0.875	0.5733
CNN#2-2	입력#1	32	$1e^{-4}$	8	0.2	46	0.871	0.5342
CNN#2-3	입력#1	32	$1e^{-3}$	8	0.3	13	0.888	0.7634

데이터셋 #2의 테스트데이터에 대한 정확도를 비교하면 CNN#2-3 모델의 정확도가 88.39% 이며 CNN#2-2 과 CNN#2-1 에 비해 각각 2.68%, 2.7% 높은 정확도를 나타냈다.

[표 3-26] CNN#2-1 분류모델의 테스트 데이터 분류 결과

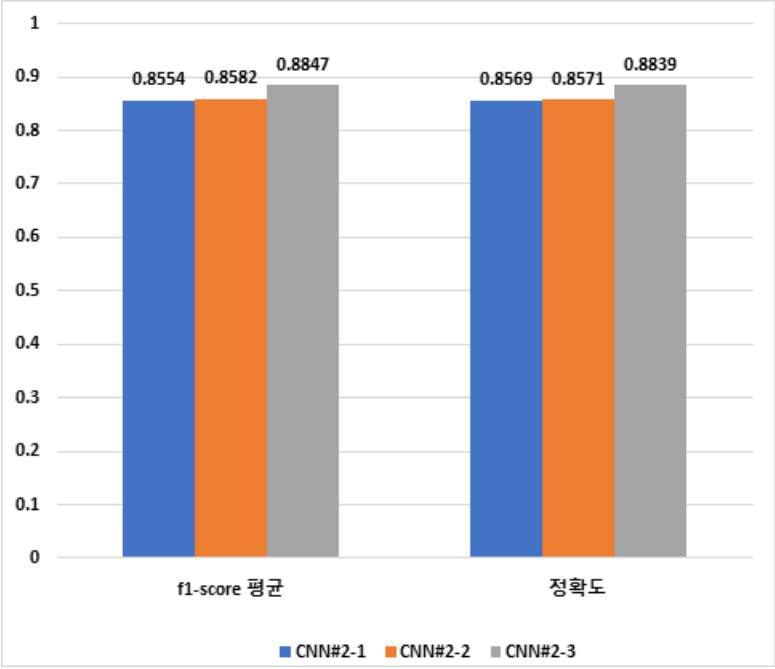
	precision	recall	f1-score	support
H01B	0.6739	0.7750	0.7209	40
H01C	0.9722	0.8750	0.9211	40
H01F	0.8974	0.8750	0.8861	40
H01G	0.7674	0.8250	0.7952	40
H01H	0.8571	0.9000	0.8780	40
H01J	0.8718	0.8500	0.8608	40
H01K	1.0000	0.8250	0.9041	40
H01L	0.7660	0.9000	0.8276	40
H01M	0.7826	0.9000	0.8372	40
H01P	0.8286	0.7250	0.7733	40
H01Q	0.8444	0.9500	0.8941	40
H01R	0.8378	0.7750	0.8052	40
H01S	1.0000	0.8500	0.9189	40
H01T	1.0000	0.9500	0.9744	40
accuracy			0.8554	560
macro avg	0.8642	0.8554	0.8569	560

[표 3-27] CNN#2-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.6818	0.7500	0.7143	40
H01C	0.9000	0.9000	0.9000	40
H01F	0.8919	0.8250	0.8571	40
H01G	0.7955	0.8750	0.8333	40
H01H	0.8333	0.8750	0.8537	40
H01J	0.9474	0.9000	0.9231	40
H01K	0.9714	0.8500	0.9067	40
H01L	0.8222	0.9250	0.8706	40
H01M	0.7292	0.8750	0.7955	40
H01P	0.8529	0.7250	0.7838	40
H01Q	0.8571	0.9000	0.8780	40
H01R	0.8611	0.7750	0.8158	40
H01S	0.9459	0.8750	0.9091	40
H01T	1.0000	0.9500	0.9744	40
accuracy			0.8571	560
macro avg	0.8636	0.8571	0.8582	560

[표 3-28] CNN#2-3 분류 모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.9375	0.7500	0.8333	40
H01C	0.9211	0.8750	0.8974	40
H01F	0.9231	0.9000	0.9114	40
H01G	0.7308	0.9500	0.8261	40
H01H	0.9286	0.9750	0.9512	40
H01J	0.9444	0.8500	0.8947	40
H01K	0.9730	0.9000	0.9351	40
H01L	0.7708	0.9250	0.8409	40
H01M	0.8718	0.8500	0.8608	40
H01P	0.9355	0.7250	0.8169	40
H01Q	0.7451	0.9500	0.8352	40
H01R	0.8974	0.8750	0.8861	40
H01S	0.9730	0.9000	0.9351	40
H01T	0.9744	0.9500	0.9620	40
accuracy			0.8839	560
macro avg	0.8947	0.8839	0.8847	560



[그림 3-11] 데이터셋#2 테스트데이터 CNN 분류모델 f1-score와 정확도

다) 데이터셋 #3에 CNN 분류모델을 적용한 분류결과

[표 3-29] 데이터셋 #3에 최적인 CNN 분류모델

	입력	필터	학습률	배치	드롭아웃	에폭	검증 정확도	검증 손실값
CNN#3-1	입력#2	32	$1e^{-4}$	16	0.3	39	0.740	1.1094
CNN#3-2	입력#2	32	$1e^{-3}$	8	0.2	23	0.743	5.9760
CNN#3-3	입력#2	16	$1e^{-3}$	16	0.3	16	0.737	4.1769

데이터셋 #3의 테스트데이터에 대한 정확도를 비교하면 CNN#3-1 분류모델의 정확도가 76.58% 이며 CNN#3-2 과 CNN#3-1 에 비해 각각 3.29%, 4.87% 높은 정확도를 나타냈다.

[표 3-30] CNN#3-1 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.6000	0.4500	0.5143	40
C01C	0.6032	0.9500	0.7379	40
C01D	0.6875	0.8250	0.7500	40
C01F	0.7647	0.6500	0.7027	40
C01G	0.6429	0.6750	0.6585	40
H01B	0.6829	0.7000	0.6914	40
H01C	0.6939	0.8500	0.7640	40
H01F	0.8378	0.7750	0.8052	40
H01G	0.7667	0.5750	0.6571	40
H01H	0.8333	0.7500	0.7895	40
H01J	0.8250	0.8250	0.8250	40
H01K	0.8780	0.9000	0.8889	40
H01L	0.6800	0.8500	0.7556	40
H01M	0.8182	0.6750	0.7397	40
H01P	0.8684	0.8250	0.8462	40
H01Q	0.9000	0.9000	0.9000	40
H01R	0.8571	0.7500	0.8000	40
H01S	0.8824	0.7500	0.8108	40
H01T	0.8974	0.8750	0.8861	40
accuracy			0.7658	760
macro avg	0.7747	0.7658	0.7644	760

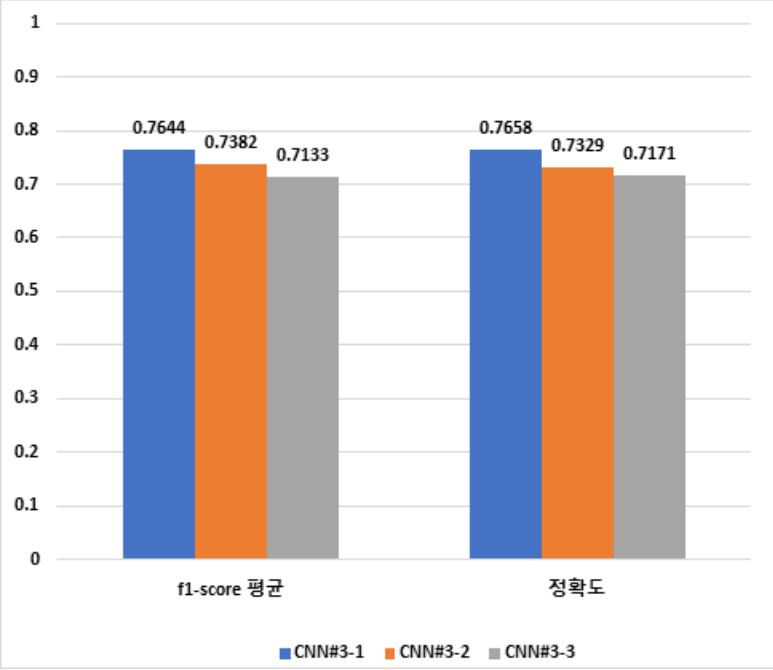
[표 3-31] CNN#3-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.6842	0.3250	0.4407	40
C01C	0.8293	0.8500	0.8395	40
C01D	0.3016	0.9500	0.4578	40
C01F	0.8846	0.5750	0.6970	40
C01G	0.6667	0.5000	0.5714	40
H01B	0.7368	0.7000	0.7179	40
H01C	0.8537	0.8750	0.8642	40
H01F	1.0000	0.7750	0.8732	40
H01G	0.9091	0.5000	0.6452	40
H01H	0.8261	0.9500	0.8837	40
H01J	0.7949	0.7750	0.7848	40
H01K	0.9231	0.9000	0.9114	40
H01L	0.8889	0.4000	0.5517	40
H01M	0.7941	0.6750	0.7297	40
H01P	0.6809	0.8000	0.7356	40
H01Q	0.7872	0.9250	0.8506	40
H01R	0.9600	0.6000	0.7385	40
H01S	0.7708	0.9250	0.8409	40
H01T	0.8605	0.9250	0.8916	40
accuracy			0.7329	760
macro avg	0.7975	0.7329	0.7382	760

[표 3-32] CNN#3-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.6786	0.4750	0.5588	40
C01C	0.5286	0.9250	0.6727	40
C01D	0.6818	0.7500	0.7143	40
C01F	0.7073	0.7250	0.7160	40
C01G	0.6757	0.6250	0.6494	40
H01B	0.7647	0.6500	0.7027	40
H01C	0.9167	0.8250	0.8684	40
H01F	0.8462	0.8250	0.8354	40
H01G	0.6429	0.9000	0.7500	40
H01H	0.5789	0.8250	0.6804	40
H01J	0.9286	0.3250	0.4815	40
H01K	0.9310	0.6750	0.7826	40
H01L	0.7857	0.5500	0.6471	40
H01M	0.9545	0.5250	0.6774	40
H01P	0.6800	0.8500	0.7556	40
H01Q	0.9600	0.6000	0.7385	40
H01R	0.5303	0.8750	0.6604	40
H01S	0.9143	0.8000	0.8533	40
H01T	0.7347	0.9000	0.8090	40

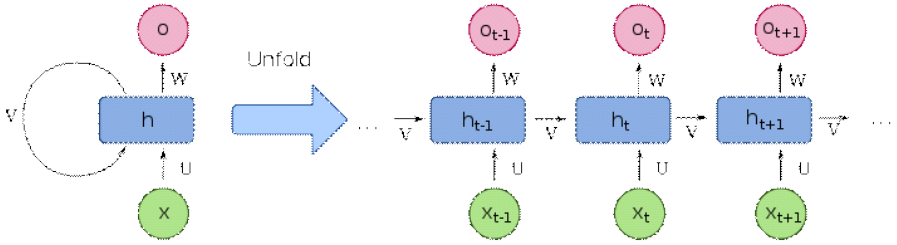
accuracy			0.7171	760
macro avg	0.7600	0.7171	0.7133	760



[그림 3-12] 데이터셋#3 테스트데이터 CNN 분류모델 f1-score와 정확도

제 4 절 Long Short-Term Memory(LSTM) 모델을 사용한 특허분류

Recurrent Neural Networks(RNN)은 노드가 방향을 가진 엣지로 연결돼 순환구조를 이루는 인공신경망의 한 종류이다. 이러한 구조는 음성, 문자 등 순차적으로 등장하는 데이터 처리에 적합한 모델로 알려져 있다.



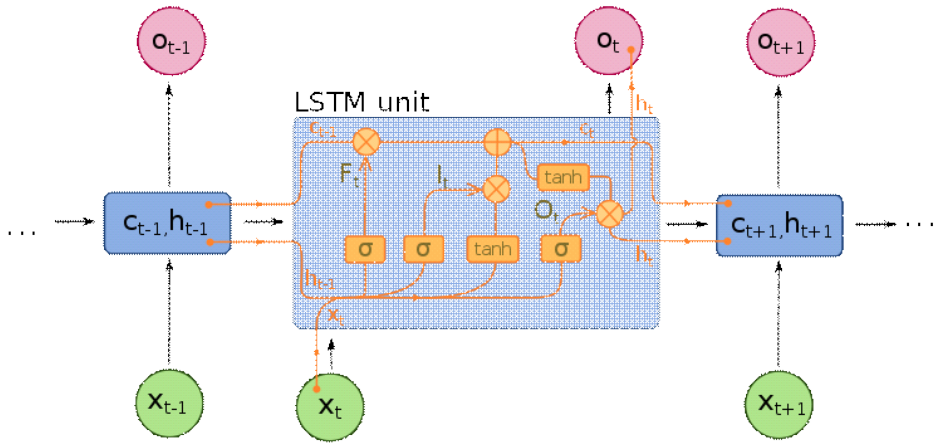
[그림 3-13] RNN의 기본 구조

Long Short-Term Memory(LSTM)은 RNN에서 발생하는 기울기 소실 문제를 해결하기 위해 고안된 딥러닝 구조이다(Hochreiter, 1997).

기울기 소실 문제는 출력층의 출력값이 0이나 1로 수렴하여 역전파로 전달될 기울기 값이 0에 수렴하여 입력층 방향으로 전달되는 기울기가 소실되어 입력층 방향 노드의 가중치가 변경되지 않는 현상이다.

RNN 에서는 훈련이 진행될수록 데이터가 변환되므로 일부 정보는 매 훈련 스텝 후 사라지게 된다. 어느 정도 스텝이 진행되면 RNN의 상태는 초기 입력값이 사라지게 된다. 즉 긴 시퀀스 길이를 가진 데이터를 RNN 모델을 사용하게 되면 기울기 소실문제가 발생하게 되며 초기 입력의 영향이 줄어들게 된다. LSTM의 핵심 아이디어는 네트워크가 장기 상태에 저장할 것, 버릴 것 그리고 읽을 것을 학습하는 것이다. RNN에 비해 삭제 게이트(forget gate)가 추가되었으며 이 삭제 게이트를 통해 역전파시 기울기 값이 급격히 사라지거나 증가하는 문제를 방지할 수 있다.

중요한 입력을 인식하고(입력 게이트의 역할), 장기 상태에 저장하고, 필요한 기간 동안 이를 보존하고(삭제 게이트의 역할), 필요할 때마다 이를 추출하기 위해 학습한다.



[그림 3-14] LSTM 기본 구조

본 연구에서 특허분류를 위해 LSTM 모델을 사용하는 이유는 사람에 의해 수행되는 특허분류에 있어 문장의 내용으로 분류를 판단하는 방법을 묘사하기 위해서이다.

1) 특허분류를 위한 LSTM 분류모델의 구성

가) 데이터 전처리

LSTM의 입력층은 행렬 또는 텐서로 이루어진 데이터를 받아들인다. 문장에 포함된 단어를 벡터로 변환하여 이를 행렬로 변환하므로 본 연구의 특허문장은 행렬로 변환된다.

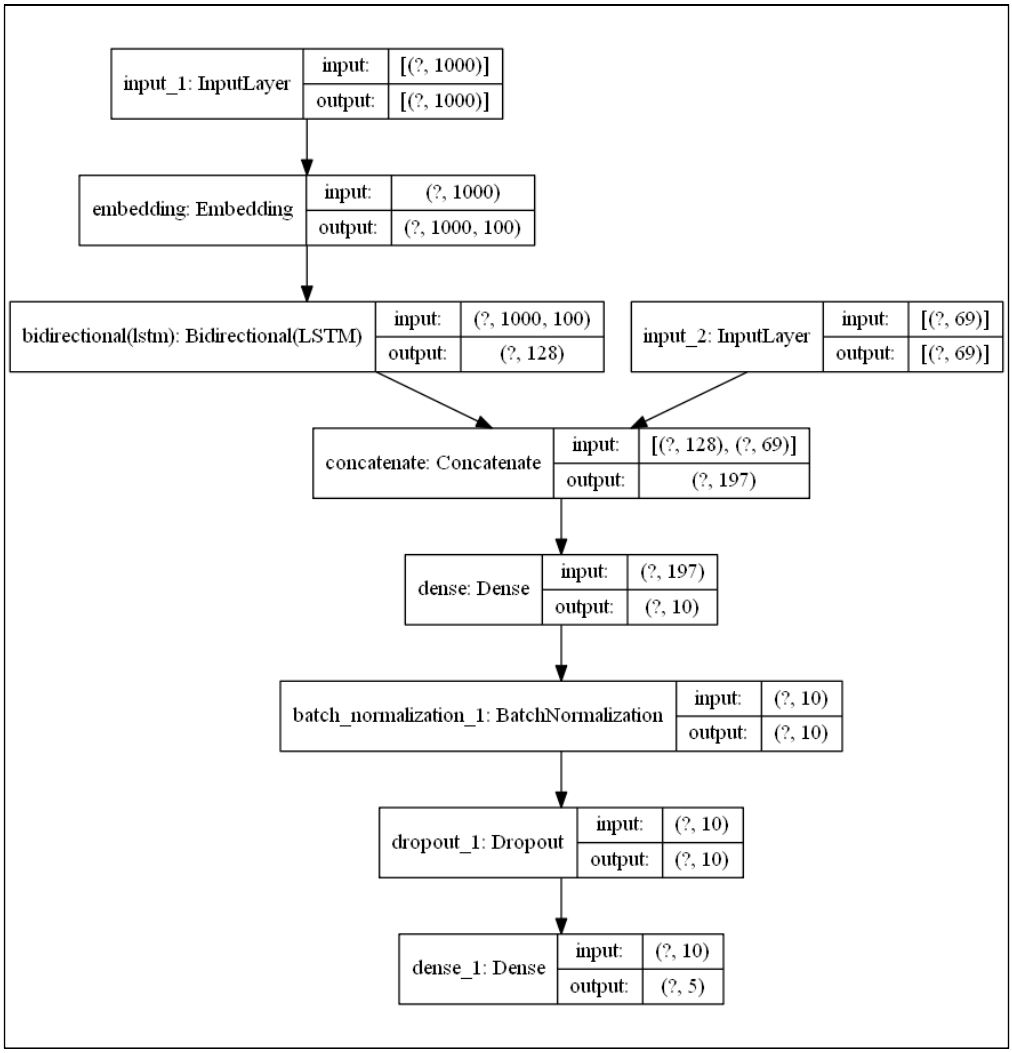
단어를 벡터로 변환하기 위한 분산표현으로 Word2Vec을 사용하였으며 벡터의 사이즈는 100으로 정의하였다. 만약 20개의 단어로 이루어진 문장이라고 하면 20 × 100 의 구조를 가지는 행렬로 변환된다.

문장에서 특수기호와 구두점 및 숫자는 제외하고 영문으로 이루어진 단어만으로 전처리 후 이를 행렬로 변환하는 방법을 사용하였다.

나) 특허분류용 LSTM 분류모델

본 연구에서 설계한 LSTM모델은 [그림 3-11] 과 같은 구조를 가진다. 임베딩층을 통과하면 문장이 행렬로 변환된다. 이를 양방향(Bi-directional) LSTM 층을 통과하도록 설계하였다. 양방향 순환 신경망은 길이가 정해진 데이터를 정방향과 역방향으로 각각 통과시켜 훈련하는 방식이다. 이후 완전 연결망으로 구성하여 출력층과 연결하여 분류를 수행하는 구조로 설계하였다.

이러한 기본 구조를 바탕으로 LSTM Cell 개수와 드롭아웃 비율 등을 변형시켜서 다양한 구조의 분류모델을 생성하였다.



[그림 3-15] 데이터셋 #1 에 적용된 LSTM 분류모델의 구조 예시

2) 매개변수의 구성

[표 3-33] LSTM의 매개변수 종류 및 적용값

매개변수 종류	적용값
입력셋	입력#1, 입력#2, 입력#3
LSTM Cell 갯수	32, 64
학습률	1e-3, 1e-4
배치사이즈	8, 16

드롭아웃 비율	0.2, 0.3, 0.5

3) 실험조건

본 연구에서는 Python 3.8 버전, Tensorflow 2.3.0 버전을 사용하였고, Keras 2.4 버전을 이용하여 프로그램을 작성하였다.

각 모델은 총 150 에폭씩 훈련하며 조기종료 기법을 적용하였다. 조기종료의 조건은 매 에폭마다 검증데이터의 손실값(loss)를 측정하여 10 에폭이 지나도 손실값이 감소하지 않으면 종료시키도록 하였다.

하나의 데이터 셋에 사용되는 모델은 입력셋(3), 하이퍼파라미터 조합(24) 적용하여 총 72개 분류모델에 대해서 훈련시켰으며 이 분류모델 중 검증데이터의 손실 값이 가장 낮은 분류모델, 검증 데이터의 정확도가 가장 높은 분류모델 그리고 훈련데이터의 손실값과 검증데이터의 손실값의 차가 가장 적은 분류모델을 각각 선택하였다. 즉, 각 데이터셋 당 3개의 최적 LSTM 분류모델을 선정하여 테스트 데이터에 대한 정확도를 측정하였다.

4) 실험결과

가) 데이터셋 #1에 LSTM 분류모델을 적용한 분류 결과

[표 3-34] 데이터셋 #1에 최적인 LSTM 분류모델

	입력	LSTM Cell	학습률	배치	에폭	검증 정확도	검증 손실값
LSTM#1-1	입력#3	64	$1e^{-3}$	8	11	0.900	0.3593
LSTM#1-2	입력#3	32	$1e^{-3}$	8	12	0.925	0.3826
LSTM#1-3	입력#2	64	$1e^{-3}$	16	16	0.900	0.3674

데이터셋 #1의 테스트데이터에 대한 정확도를 비교하면 LSTM#1-3 분류모델의 정확도가 85.50% 이며 LSTM#1-2 과 LSTM#1-1 에 비해 각각

2.0%, 4.0% 높은 정확도를 나타냈다.

[표 3-35] LSTM#1-1 분류모델의 테스트 데이터 분류 결과

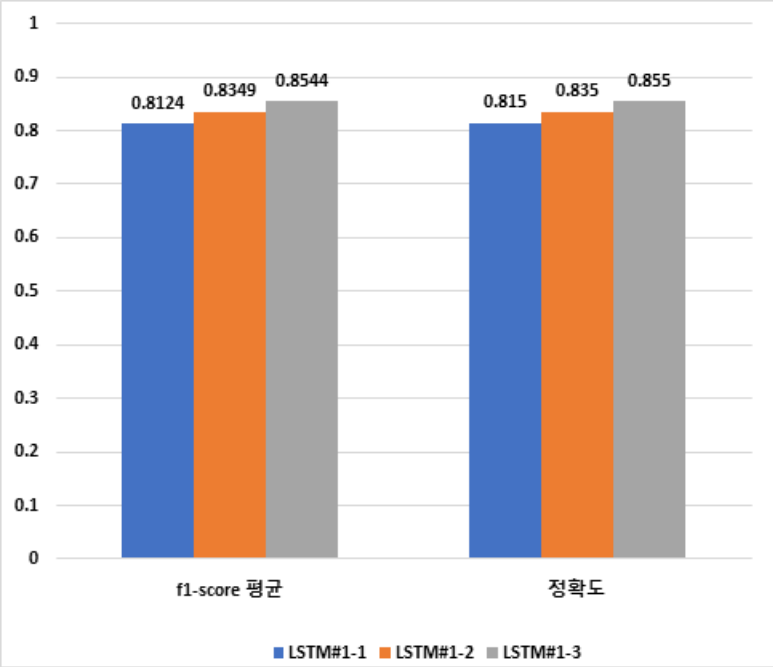
	precision	recall	f1-score	support
C01B	0.7879	0.6500	0.7123	40
C01C	0.9048	0.9500	0.9268	40
C01D	0.8919	0.8250	0.8571	40
C01F	0.7708	0.9250	0.8409	40
C01G	0.7250	0.7250	0.7250	40
accuracy			0.8150	200
macro avg	0.8161	0.8150	0.8124	200

[표 3-36] LSTM#1-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8158	0.7750	0.7949	40
C01C	0.9024	0.9250	0.9136	40
C01D	0.8205	0.8000	0.8101	40
C01F	0.7727	0.8500	0.8095	40
C01G	0.8684	0.8250	0.8462	40
accuracy			0.8350	200
macro avg	0.8360	0.8350	0.8349	200

[표 3-37] LSTM#1-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8537	0.8750	0.8642	40
C01C	0.8571	0.9000	0.8780	40
C01D	0.8947	0.8500	0.8718	40
C01F	0.9355	0.7250	0.8169	40
C01G	0.7708	0.9250	0.8409	40
accuracy			0.8550	200
macro avg	0.8624	0.8550	0.8544	200



[그림 3-16] 데이터셋#1 테스트데이터 LSTM 분류모델 f1-score와 정확도

나) 데이터셋 #2에 LSTM 분류모델을 적용한 분류 결과

데이터셋 #2의 테스트데이터에 대한 정확도를 비교하면 LSTM#2-2 분류모델의 정확도가 94.64% 이며 LSTM#2-1 과 LSTM#2-3 에 비해 각각 0.18%, 1.96% 높은 정확도를 나타냈다.

[표 3-38] 데이터셋 #2에 최적인 LSTM 모델

	입력	LSTM Cell	학습률	배치	에폭	검증 정확도	검증 손실값
LSTM#2-1	입력#2	64	$1e^{-3}$	8	5	0.960	0.1721
LSTM#2-2	입력#1	32	$1e^{-3}$	8	12	0.964	0.1925
LSTM#2-3	입력#3	32	$1e^{-3}$	16	24	0.964	0.1706

[표 3-39] LSTM#2-1 분류모델의 테스트 데이터 분류 결과

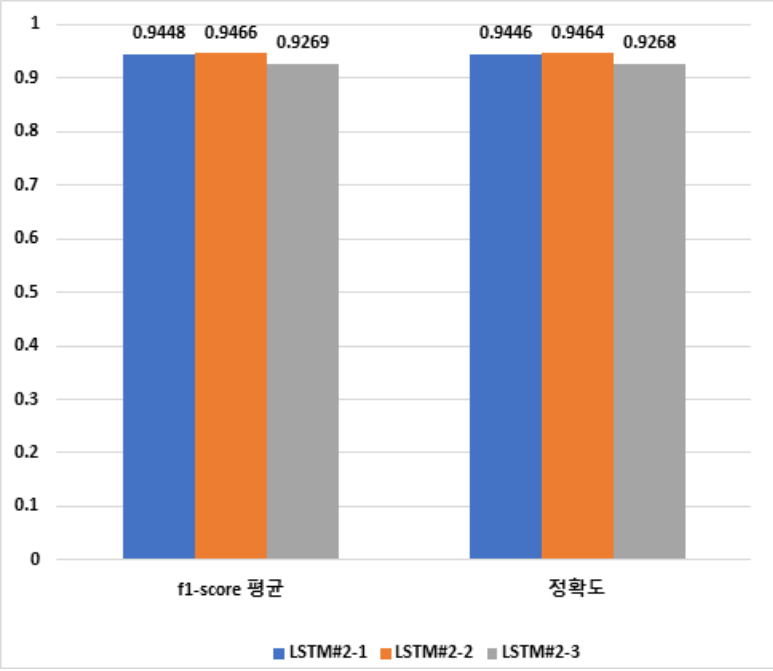
	precision	recall	f1-score	support
H01B	0.9048	0.9500	0.9268	40
H01C	0.9744	0.9500	0.9620	40
H01F	0.9730	0.9000	0.9351	40
H01G	0.8444	0.9500	0.8941	40
H01H	0.9750	0.9750	0.9750	40
H01J	0.9091	1.0000	0.9524	40
H01K	0.9512	0.9750	0.9630	40
H01L	0.9487	0.9250	0.9367	40
H01M	0.9737	0.9250	0.9487	40
H01P	0.9487	0.9250	0.9367	40
H01Q	0.8864	0.9750	0.9286	40
H01R	0.9714	0.8500	0.9067	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	0.9500	0.9744	40
accuracy			0.9446	560
macro avg	0.9472	0.9446	0.9448	560

[표 3-40] LSTM#2-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.9048	0.9500	0.9268	40
H01C	0.9512	0.9750	0.9630	40
H01F	1.0000	0.9750	0.9873	40
H01G	0.8837	0.9500	0.9157	40
H01H	0.9512	0.9750	0.9630	40
H01J	0.9512	0.9750	0.9630	40
H01K	0.9737	0.9250	0.9487	40
H01L	0.9268	0.9500	0.9383	40
H01M	0.9737	0.9250	0.9487	40
H01P	0.9722	0.8750	0.9211	40
H01Q	0.8864	0.9750	0.9286	40
H01R	0.8974	0.8750	0.8861	40
H01S	1.0000	0.9500	0.9744	40
H01T	1.0000	0.9750	0.9873	40
accuracy			0.9464	560
macro avg	0.9480	0.9464	0.9466	560

[표 3-41] LSTM#2-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.8780	0.9000	0.8889	40
H01C	0.9487	0.9250	0.9367	40
H01F	0.9459	0.8750	0.9091	40
H01G	0.8837	0.9500	0.9157	40
H01H	0.9524	1.0000	0.9756	40
H01J	0.9487	0.9250	0.9367	40
H01K	0.9500	0.9500	0.9500	40
H01L	0.8780	0.9000	0.8889	40
H01M	0.9744	0.9500	0.9620	40
H01P	0.9211	0.8750	0.8974	40
H01Q	0.8571	0.9000	0.8780	40
H01R	0.8500	0.8500	0.8500	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9268	560
macro avg	0.9277	0.9268	0.9269	560



[그림 3-17] 데이터셋#2 테스트데이터 LSTM 분류모델 f1-score와 정확도

다) 데이터셋 #3에 LSTM 분류모델을 적용한 분류결과

[표 3-42] 데이터셋 #3에 최적인 LSTM 분류모델

	입력	LSTM Cell	학습률	배치	에폭	검증 정확도	검증 손실값
LSTM#3-1	입력#3	32	$1e^{-4}$	16	87	0.957	0.1820
LSTM#3-2	입력#1	32	$1e^{-3}$	8	14	0.954	0.2568
LSTM#3-3	입력#2	64	$1e^{-3}$	16	9	0.964	0.2149

데이터셋 #3의 테스트데이터에 대한 정확도를 비교하면 LSTM#3-2 분류모델의 정확도가 93.29% 이며 LSTM#3-1 과 LSTM#3-3 에 비해 각각 1.32%, 1.84% 높은 정확도를 나타냈다.

[표 3-43] LSTM#3-1 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.9143	0.8000	0.8533	40
C01C	0.9211	0.8750	0.8974	40
C01D	0.9268	0.9500	0.9383	40
C01F	0.9211	0.8750	0.8974	40
C01G	0.8500	0.8500	0.8500	40
H01B	0.8043	0.9250	0.8605	40
H01C	0.9500	0.9500	0.9500	40
H01F	1.0000	0.9000	0.9474	40
H01G	0.8571	0.9000	0.8780	40
H01H	0.9512	0.9750	0.9630	40
H01J	0.9286	0.9750	0.9512	40
H01K	0.9512	0.9750	0.9630	40
H01L	0.9737	0.9250	0.9487	40
H01M	0.8333	0.8750	0.8537	40
H01P	0.9211	0.8750	0.8974	40
H01Q	0.9070	0.9750	0.9398	40
H01R	0.9000	0.9000	0.9000	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9197	760
macro avg	0.9216	0.9197	0.9198	760

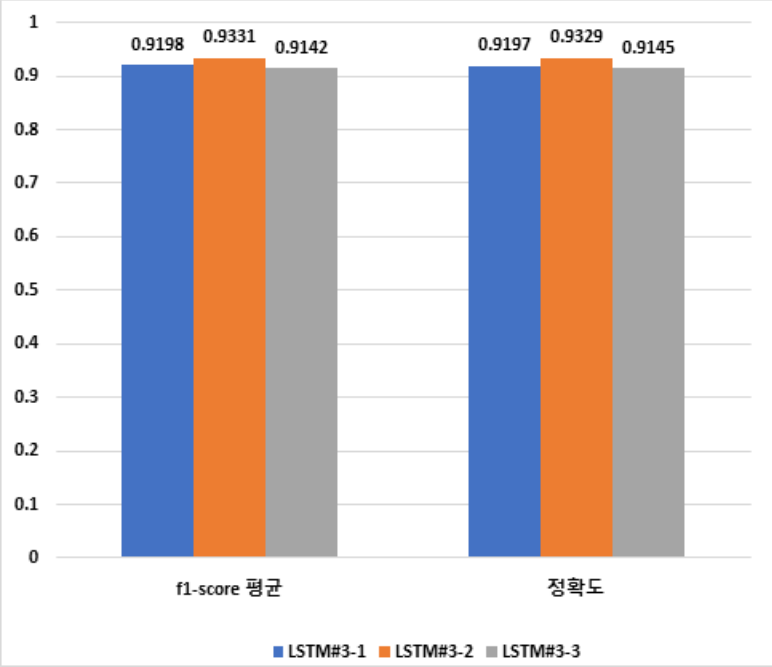
[표 3-44] LSTM#3-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.9024	0.9250	0.9136	40
C01C	0.9722	0.8750	0.9211	40
C01D	0.9024	0.9250	0.9136	40
C01F	0.9231	0.9000	0.9114	40
C01G	0.8333	0.8750	0.8537	40
H01B	0.9231	0.9000	0.9114	40
H01C	0.9286	0.9750	0.9512	40
H01F	1.0000	0.9250	0.9610	40
H01G	0.8636	0.9500	0.9048	40
H01H	0.9512	0.9750	0.9630	40
H01J	1.0000	0.9500	0.9744	40
H01K	0.9756	1.0000	0.9877	40
H01L	0.9250	0.9250	0.9250	40
H01M	0.8780	0.9000	0.8889	40
H01P	0.9250	0.9250	0.9250	40
H01Q	0.8864	0.9750	0.9286	40
H01R	0.9714	0.8500	0.9067	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9329	760
macro avg	0.9348	0.9329	0.9331	760

[표 3-45] LSTM#3-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.7949	0.7750	0.7848	40
C01C	0.9070	0.9750	0.9398	40
C01D	0.8298	0.9750	0.8966	40
C01F	0.9211	0.8750	0.8974	40
C01G	0.9118	0.7750	0.8378	40
H01B	0.8537	0.8750	0.8642	40
H01C	0.9500	0.9500	0.9500	40
H01F	0.9737	0.9250	0.9487	40
H01G	0.8636	0.9500	0.9048	40
H01H	0.9512	0.9750	0.9630	40
H01J	0.8667	0.9750	0.9176	40
H01K	0.9268	0.9500	0.9383	40
H01L	0.9730	0.9000	0.9351	40
H01M	0.8462	0.8250	0.8354	40
H01P	0.9730	0.9000	0.9351	40
H01Q	0.9268	0.9500	0.9383	40
H01R	0.9459	0.8750	0.9091	40
H01S	1.0000	0.9750	0.9873	40

H01T	1.0000	0.9750	0.9873	40
accuracy			0.9145	760
macro avg	0.9166	0.9145	0.9142	760

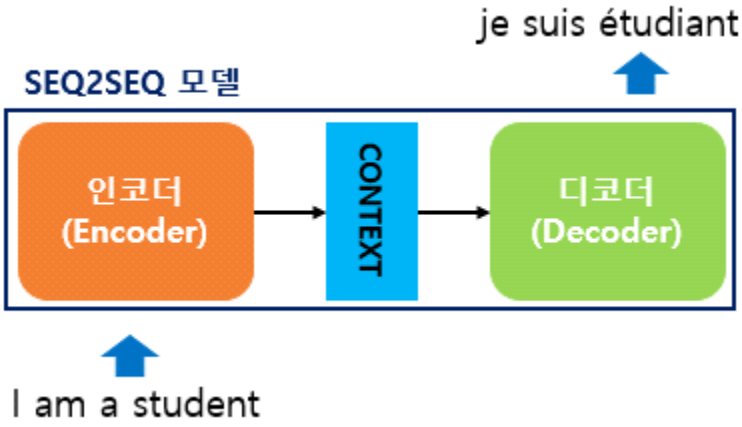


[그림 3-18] 데이터셋#3 테스트데이터 LSTM 분류모델 f1-score와 정확도

제 5 절 Attention(어텐션) 모델을 사용한 특허분류

RNN의 단점의 기울기 소멸을 극복하기 위해 LSTM 모델이 탄생하였다. 기울기 소멸의 이유는 처음 입력의 정보가 뒤로 갈수록 사라지는 RNN의 특성 때문에 생긴 문제이다. 이러한 문제를 감소시키기 위해 LSTM은 입력 중 핵심적인 정보를 잊어버리지 않고 핵심적이지 못한 정보를 잊어버려서 다음 스텝에 전달하는 방식이다.

RNN과 LSTM은 출력이 바로 이전 입력까지만 고려하기 때문에 번역 또는 여러 스텝후의 미래를 예측하기에는 정확도가 떨어진다. 이런 문제를 해결하기 위해 Seq2Seq 가 등장하게 되었다.



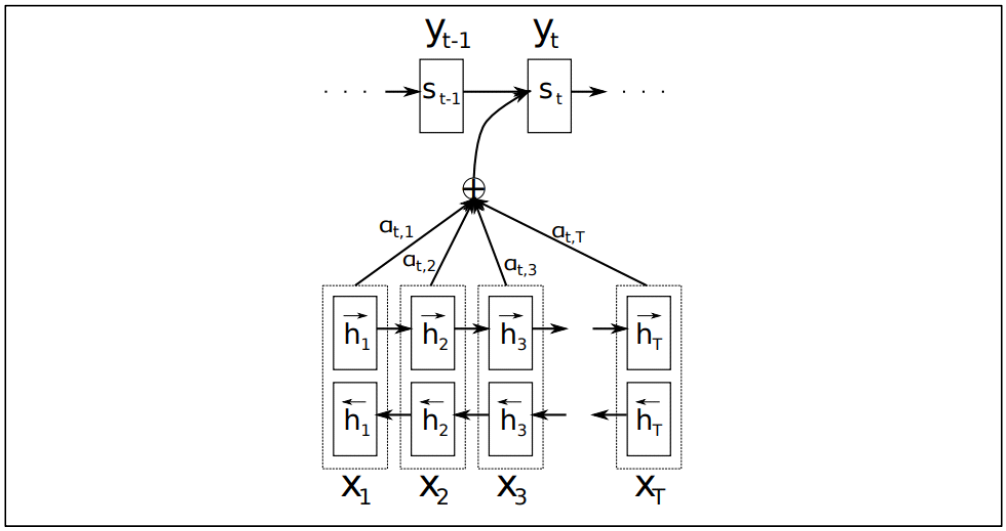
[그림 3-19] Seq2Seq 내부구조

Seq2Seq는 인코더-디코더 네트워크라고 불리며 두 개의 RNN 또는 LSTM으로 구성된 모델이다. 인코더는 입력 시퀀스를 읽고 단일 벡터를 출력하고 이 단일벡터는 컨텍스트 벡터라고 한다. 디코더는 컨텍스트 벡터를 읽어 출력 시퀀스를 생성한다.

컨텍스트 벡터는 고정된 크기의 벡터이고 이를 통해 입력된 전체 데이터의 맥락을 파악한 벡터라고 할 수 있다. 고정된 벡터이기 때문에 입력 데이터의 길이가 매우 길면 LSTM에서도 효율적으로 학습하지 못한다. 즉 하나의 고정된 크기의 벡터에 모든 정보를 압축하려고 하니 정보 손실이 발생한다. RNN(LSTM)의 고질적인 문제인 기울기 소실 문제는 계속 존재하게 된다.

이런 이유로 입력 데이터가 긴 상황에서는 정확도가 떨어지는 현상이 발생하였고, 이런 현상을 극복하기 위해 중요한 입력에 집중하여 디코더에 바로 전달하는 어텐션이 등장하였다.(Bahdanau et al., 2014)

어텐션의 기본 아이디어는 출력을 예측하는 시점에서 전체 입력 데이터를 다시 한번 참고 한다는 점이다. 단, 전체 입력 데이터를 전부 다 동일한 비율로 참고하는 것이 아니라, 해당 시점에서 예측해야 할 출력과 연관이 있는 입력 부분에 좀 더 집중하게 된다.



[그림 3-20] 인코더-디코더 모델에 결합된 어텐션 매커니즘(Bahdanau et al., 2014)

본 연구에서 특허분류를 위해 어텐션 모델을 사용하는 이유는 사람에 의해 수행되는 특허분류에 있어 문장의 맥락과 주요한 키워드를 통해 특허분류

를 판단하는 방법을 묘사하기 위해서이다.

1) 특허분류를 위한 어텐션 분류모델의 구성

가) 데이터 전처리

어텐션의 입력층은 행렬 또는 텐서로 이루어진 데이터를 받아들인다. 문장에 포함된 단어를 벡터로 변환하여 이를 행렬로 변환하므로 본 연구의 특허 문장은 행렬로 변환된다.

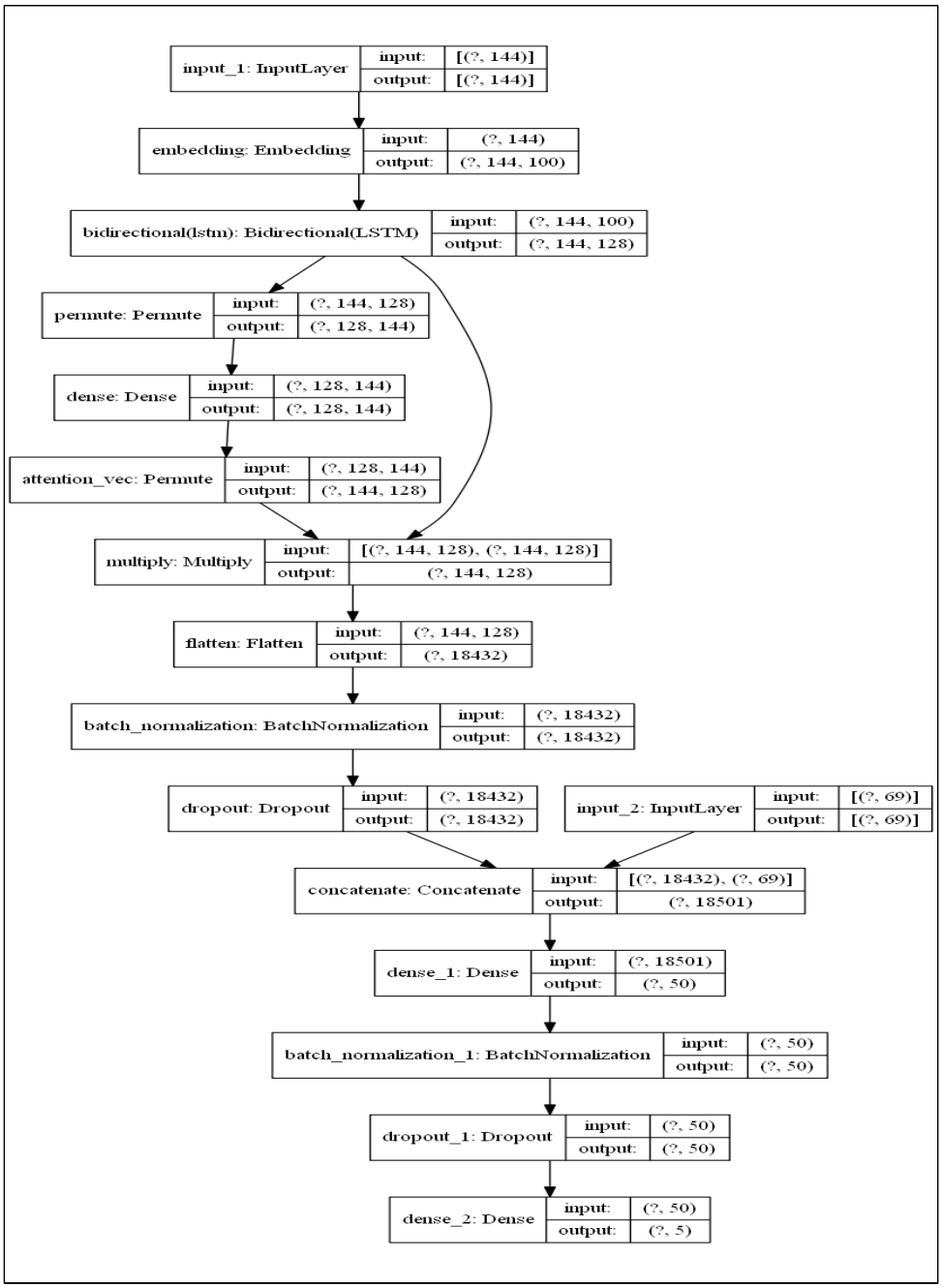
단어를 벡터로 변환하기 위한 분산표현으로 Word2Vec을 사용하였으며 벡터의 사이즈는 100으로 정의하였다. 만약 20개의 단어로 이루어진 문장이라고 하면 20 × 100 의 구조를 가지는 행렬로 변환된다.

문장에서 특수기호와 구두점 및 숫자는 제외하고 영문으로 이루어진 단어만으로 전처리 후 이를 행렬로 변환하는 방법을 사용하였다.

나) 특허분류용 어텐션 분류모델

본 연구에서 사용된 어텐션 분류모델의 구조는 [그림 3-21]과 같은 구조를 가진다. 임베딩 층을 통과한 출력은 양방향 LSTM 층을 통과하고 이 출력이 어텐션 블럭층을 통과하게 된다. 해당 출력의 차원을 변경시켜 완전 연결층, 배치 정규화 층을 통과시며 마지막 출력층을 통과하게 된다.

어텐션 블럭을 위해 사용된 순환신경망 층으로는 LSTM 을 사용하였고 영문 해석에 강점을 보이는 양방향 방식을 사용하여 분류모델 구조를 설계하였다.



[그림 3-21] 데이터셋 #1 에 적용된 어텐션 모델의 구조 예시

2) 매개변수의 구성

[표 3-46] 어텐션의 매개변수 종류 및 적용값

매개변수 종류	적용값
입력셋	입력#1, 입력#2, 입력#3
LSTM Cell 갯수	32, 64
학습률	1e-3, 1e-4
배치사이즈	8, 16
드롭아웃 비율	0.2, 0.3, 0.5

3) 실험 조건

본 연구에서는 Python 3.8 버전, Tensorflow 2.3.0 버전을 사용하였고, Keras 2.4 버전을 이용하여 프로그램을 작성하였다.

각 모델은 총 150 에폭씩 훈련하며 조기종료 기법을 적용하였다. 조기종료의 조건은 매 에폭마다 검증데이터의 손실값(loss)를 측정하여 10 에폭이 지나도 손실값이 감소하지 않으면 종료시키도록 하였다.

하나의 데이터 셋에 사용되는 모델은 입력셋(3), 하이퍼파라미터 조합(24) 적용하여 총 72개 모델에 대해서 훈련시켰으며 이 모델 중 검증데이터의 손실 값이 가장 낮은 분류모델, 검증 데이터의 정확도가 가장 높은 분류모델 그리고 훈련데이터의 손실값과 검증데이터의 손실값의 차가 가장 적은 분류모델을 각각 선택하였다. 즉, 각 데이터셋 당 3개의 최적 어텐션 분류모델을 선정하여 테스트 데이터에 대한 정확도를 측정하였다.

4) 실험 결과

가) 데이터셋 #1 에 어텐션 분류모델을 적용한 분류 결과

[표 3-47] 데이터셋 #1에 최적인 어텐션 분류모델

	입력	Cell	학습률	배치	에폭	검증 정확도	검증 손실값
ATN#1-1	입력#1	64	$1e^{-3}$	16	9	0.900	0.2681
ATN#1-2	입력#1	64	$1e^{-3}$	16	20	0.913	0.3055
ATN#1-3	입력#1	16	$1e^{-3}$	16	15	0.913	0.2376

데이터셋 #1의 테스트데이터에 대한 정확도를 비교하면 ATN#1-1 분류모델의 정확도가 90.05% 이며 ATN#1-3 과 ATN#1-2에 비해 각각 0.5%, 2.5% 높은 정확도를 나타냈다.

[표 3-48] ATN#1-1 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.9250	0.9250	0.9250	40
C01C	1.0000	0.8250	0.9041	40
C01D	0.9231	0.9000	0.9114	40
C01F	0.8810	0.9250	0.9024	40
C01G	0.8261	0.9500	0.8837	40
accuracy			0.9050	200
macro avg	0.9110	0.9050	0.9053	200

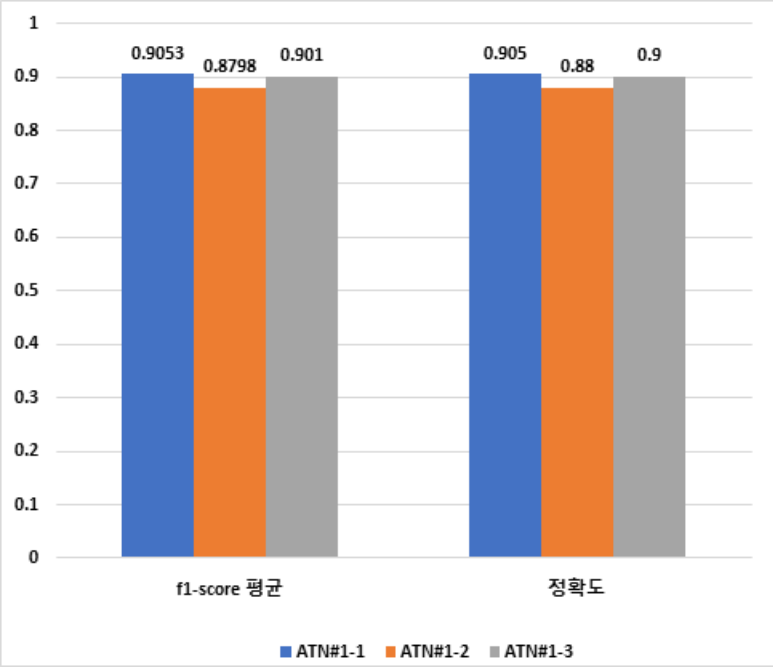
[표 3-49] ATN#1-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.9444	0.8500	0.8947	40
C01C	0.8780	0.9000	0.8889	40
C01D	0.8837	0.9500	0.9157	40
C01F	0.9394	0.7750	0.8493	40
C01G	0.7872	0.9250	0.8506	40
accuracy			0.8800	200
macro avg	0.8866	0.8800	0.8798	200

[표 3-50] ATN#1-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.9487	0.9250	0.9367	40

C01C	1.0000	0.8750	0.9333	40
C01D	0.9024	0.9250	0.9136	40
C01F	0.8571	0.9000	0.8780	40
C01G	0.8140	0.8750	0.8434	40
accuracy			0.9000	200
macro avg	0.9045	0.9000	0.9010	200



[그림 3-22] 데이터셋#1 테스트데이터 어텐션 분류모델 f1-score와 정확도

나) 데이터셋 #2 에 어텐션 분류모델을 적용한 분류 결과

[표 3-51] 데이터셋 #2에 최적인 어텐션 분류모델

	입력	Cell	학습률	배치	에폭	검증 정확도	검증 손실값
ATN#2-1	입력#3	64	$1e^{-4}$	16	19	0.969	0.1551
ATN#2-2	입력#1	32	$1e^{-4}$	16	32	0.969	0.1772
ATN#2-3	입력#2	64	$1e^{-3}$	16	21	0.973	0.1874

데이터셋 #2의 테스트데이터에 대한 정확도를 비교하면 ATN#2-1 분류 모델의 정확도가 96.43% 이며 ATN#2-2 과 ATN#2-3에 비해 각각 0.5%, 2.5% 높은 정확도를 나타냈다.

[표 3-52] ATN#2-1 모델의 테스트 데이터 분류 결과

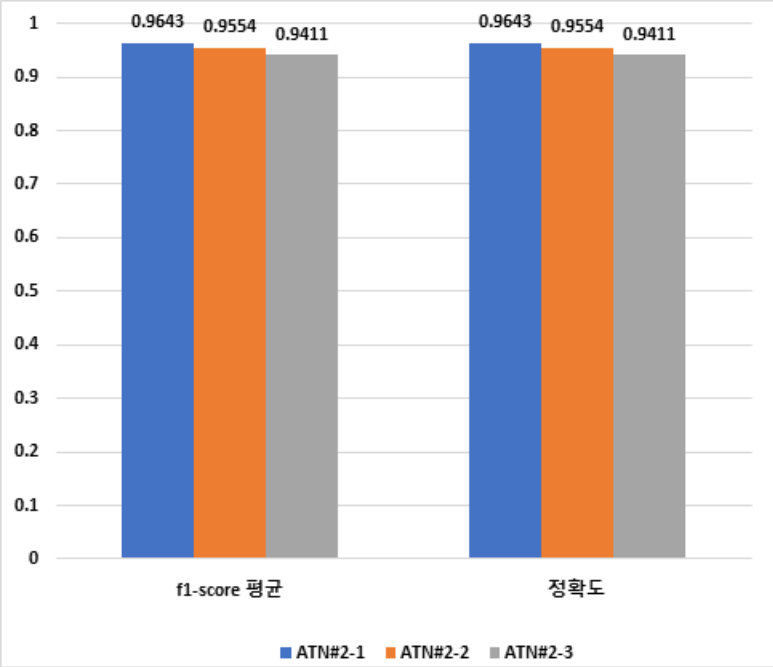
	precision	recall	f1-score	support
H01B	0.9474	0.9000	0.9231	40
H01C	0.9750	0.9750	0.9750	40
H01F	1.0000	0.9500	0.9744	40
H01G	0.9268	0.9500	0.9383	40
H01H	0.9750	0.9750	0.9750	40
H01J	0.9756	1.0000	0.9877	40
H01K	0.9756	1.0000	0.9877	40
H01L	0.9070	0.9750	0.9398	40
H01M	0.9750	0.9750	0.9750	40
H01P	1.0000	0.9250	0.9610	40
H01Q	0.9286	0.9750	0.9512	40
H01R	0.9250	0.9250	0.9250	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9643	560
macro avg	0.9651	0.9643	0.9643	560

[표 3-53] ATN#2-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.9474	0.9000	0.9231	40
H01C	0.9750	0.9750	0.9750	40
H01F	0.9487	0.9250	0.9367	40
H01G	0.8837	0.9500	0.9157	40
H01H	0.9750	0.9750	0.9750	40
H01J	1.0000	0.9500	0.9744	40
H01K	0.9524	1.0000	0.9756	40
H01L	0.9070	0.9750	0.9398	40
H01M	0.9487	0.9250	0.9367	40
H01P	1.0000	0.9250	0.9610	40
H01Q	0.9091	1.0000	0.9524	40
H01R	0.9474	0.9000	0.9231	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9554	560
macro avg	0.9567	0.9554	0.9554	560

[표 3-54] ATN#2-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.9429	0.8250	0.8800	40
H01C	0.9744	0.9500	0.9620	40
H01F	0.9737	0.9250	0.9487	40
H01G	0.8478	0.9750	0.9070	40
H01H	0.8696	1.0000	0.9302	40
H01J	0.9750	0.9750	0.9750	40
H01K	0.8889	1.0000	0.9412	40
H01L	0.9286	0.9750	0.9512	40
H01M	0.9737	0.9250	0.9487	40
H01P	0.9722	0.8750	0.9211	40
H01Q	0.9000	0.9000	0.9000	40
H01R	0.9730	0.9000	0.9351	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	0.9750	0.9873	40
accuracy			0.9411	560
macro avg	0.9443	0.9411	0.9411	560



[그림 3-23] 데이터셋#2 테스트데이터 어텐션 분류모델 f1-score와 정확도

다) 데이터셋 #3 에 어텐션 모델을 적용한 분류 결과

[표 3-55] 데이터셋 #3에 최적인 어텐션 분류모델

	입력	LSTM Cell	학습률	배치	에폭	검증 정확도	검증 손실값
ATN#3-1	입력#2	64	$1e^{-4}$	8	17	0.954	0.2436
ATN#3-2	입력#1	32	$1e^{-4}$	8	11	0.954	0.2602
ATN#3-3	입력#1	64	$1e^{-3}$	8	14	0.954	0.2311

데이터셋 #3의 테스트데이터에 대한 정확도를 비교하면 ATN#3-2 분류 모델의 정확도가 92.89% 이며 ATN#3-1 과 ATN#3-3에 비해 각각 1.05%, 1.05% 높은 정확도를 나타냈다.

[표 3-56] ATN#3-1 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8947	0.8500	0.8718	40
C01C	0.9697	0.8000	0.8767	40
C01D	0.8478	0.9750	0.9070	40
C01F	0.8919	0.8250	0.8571	40
C01G	0.8140	0.8750	0.8434	40
H01B	0.8750	0.8750	0.8750	40
H01C	0.9500	0.9500	0.9500	40
H01F	0.9730	0.9000	0.9351	40
H01G	0.8571	0.9000	0.8780	40
H01H	0.9744	0.9500	0.9620	40
H01J	0.9512	0.9750	0.9630	40
H01K	0.9302	1.0000	0.9639	40
H01L	0.9459	0.8750	0.9091	40
H01M	0.8372	0.9000	0.8675	40
H01P	0.9487	0.9250	0.9367	40
H01Q	0.8864	0.9750	0.9286	40
H01R	0.9487	0.9250	0.9367	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9184	760
macro avg	0.9208	0.9184	0.9184	760

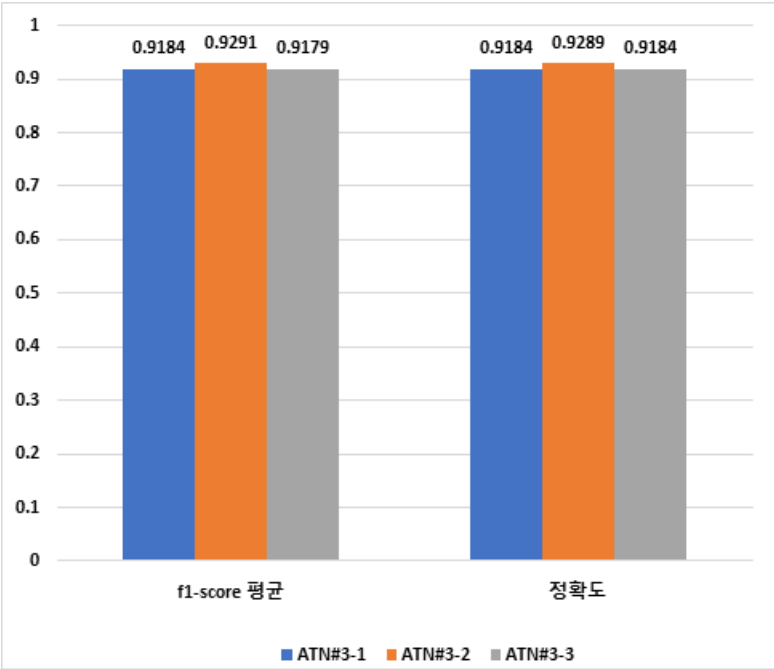
[표 3-57] ATN#3-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.9143	0.8000	0.8533	40
C01C	0.9048	0.9500	0.9268	40
C01D	0.9268	0.9500	0.9383	40
C01F	0.8684	0.8250	0.8462	40
C01G	0.8947	0.8500	0.8718	40
H01B	0.9000	0.9000	0.9000	40
H01C	0.9744	0.9500	0.9620	40
H01F	1.0000	0.9250	0.9610	40
H01G	0.8605	0.9250	0.8916	40
H01H	0.9286	0.9750	0.9512	40
H01J	0.9500	0.9500	0.9500	40
H01K	0.9756	1.0000	0.9877	40
H01L	0.9500	0.9500	0.9500	40
H01M	0.7708	0.9250	0.8409	40
H01P	1.0000	0.9000	0.9474	40

H01Q	0.9091	1.0000	0.9524	40
H01R	0.9730	0.9000	0.9351	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9289	760
macro avg	0.9316	0.9289	0.9291	760

[표 3-58] ATN#3-3 분류모델의 테스트 데이터 분류 결과

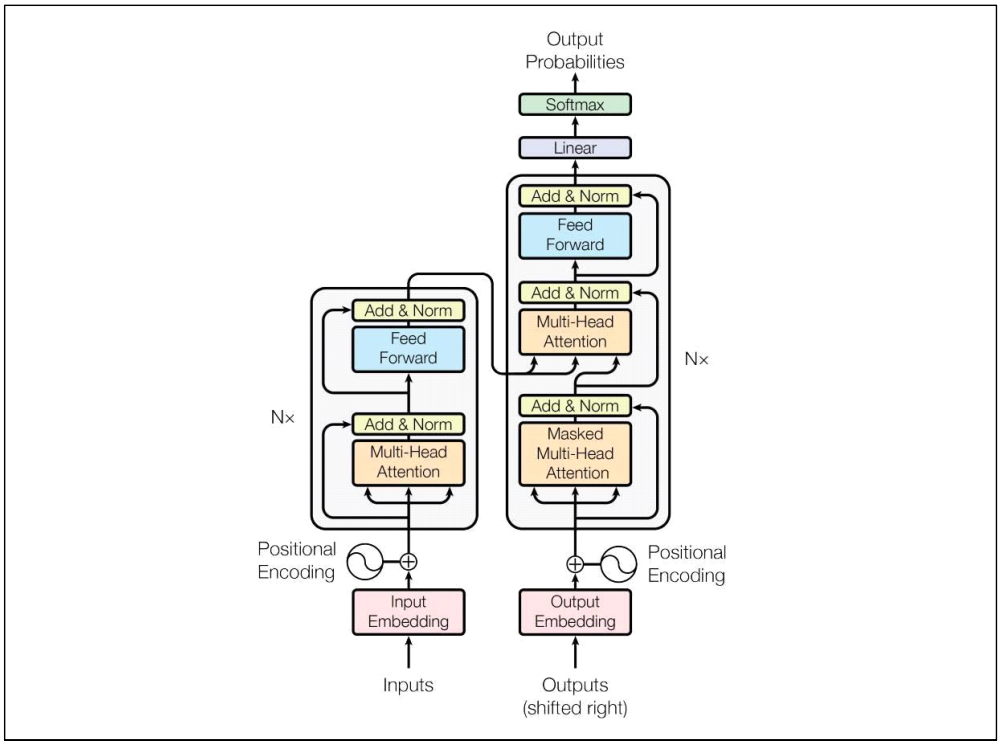
	precision	recall	f1-score	support
C01B	0.9000	0.6750	0.7714	40
C01C	0.8837	0.9500	0.9157	40
C01D	0.9268	0.9500	0.9383	40
C01F	0.8919	0.8250	0.8571	40
C01G	0.8889	0.8000	0.8421	40
H01B	0.8222	0.9250	0.8706	40
H01C	0.9750	0.9750	0.9750	40
H01F	0.9737	0.9250	0.9487	40
H01G	0.8636	0.9500	0.9048	40
H01H	0.9512	0.9750	0.9630	40
H01J	1.0000	0.9500	0.9744	40
H01K	0.9500	0.9500	0.9500	40
H01L	0.8837	0.9500	0.9157	40
H01M	0.7551	0.9250	0.8315	40
H01P	0.9722	0.8750	0.9211	40
H01Q	0.8889	1.0000	0.9412	40
H01R	1.0000	0.8750	0.9333	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9184	760
macro avg	0.9225	0.9184	0.9179	760



[그림 3-24] 데이터셋#3 테스트데이터 어텐션 분류모델 f1-score와 정확도

제 6 절 Transformer(트랜스포머) 모델을 사용한 특허분류

트랜스포머는 2017년 구글이 발표한 논문인 “Attention is all you need" 에서 제안한 모델이다. 이 모델은 기본적인 인코더 디코더를 구현하는 Seq2Seq 구조를 따르지만 RNN 또는 LSTM을 사용하지 않고 어텐션만을 사용해서 Seq2Seq를 구현한 모델이다. 이 모델은 순환신경망을 사용한 Seq2Seq 보다 높은 성능을 낸다고 보고되었다.



[그림 3-25] The Transformer - model architecture (Vaswani et al., 2017)

[그림 3-25]에서 왼쪽 부분은 인코더이고 오른쪽 부분은 디코더이다. 인코더와 디코더는 멀티헤드 어텐션을 포함하고 있다. 또한 디코더에는 마스크드

멀티헤드 어텐션을 포함하고 있다.

인코더의 멀티헤드 어텐션 층은 관련이 많은 단어에 더 많은 주의를 기울이면서 각 단어와 동일한 문장에 있는 다른 단어의 관계를 인코딩 한다. 이런 메카니즘을 셀프어텐션 이라고 한다.

디코더의 마스크드 멀티헤드 어텐션 층도 동일한 작업을 수행한다. 다만 각 단어는 이전에 등장한 단어에만 주의를 기울일 수 있다. 디코더의 위쪽 멀티헤드 어텐션 층은 디코더가 입력 문장에 있는 단어에 주위를 기울이는 곳이다.

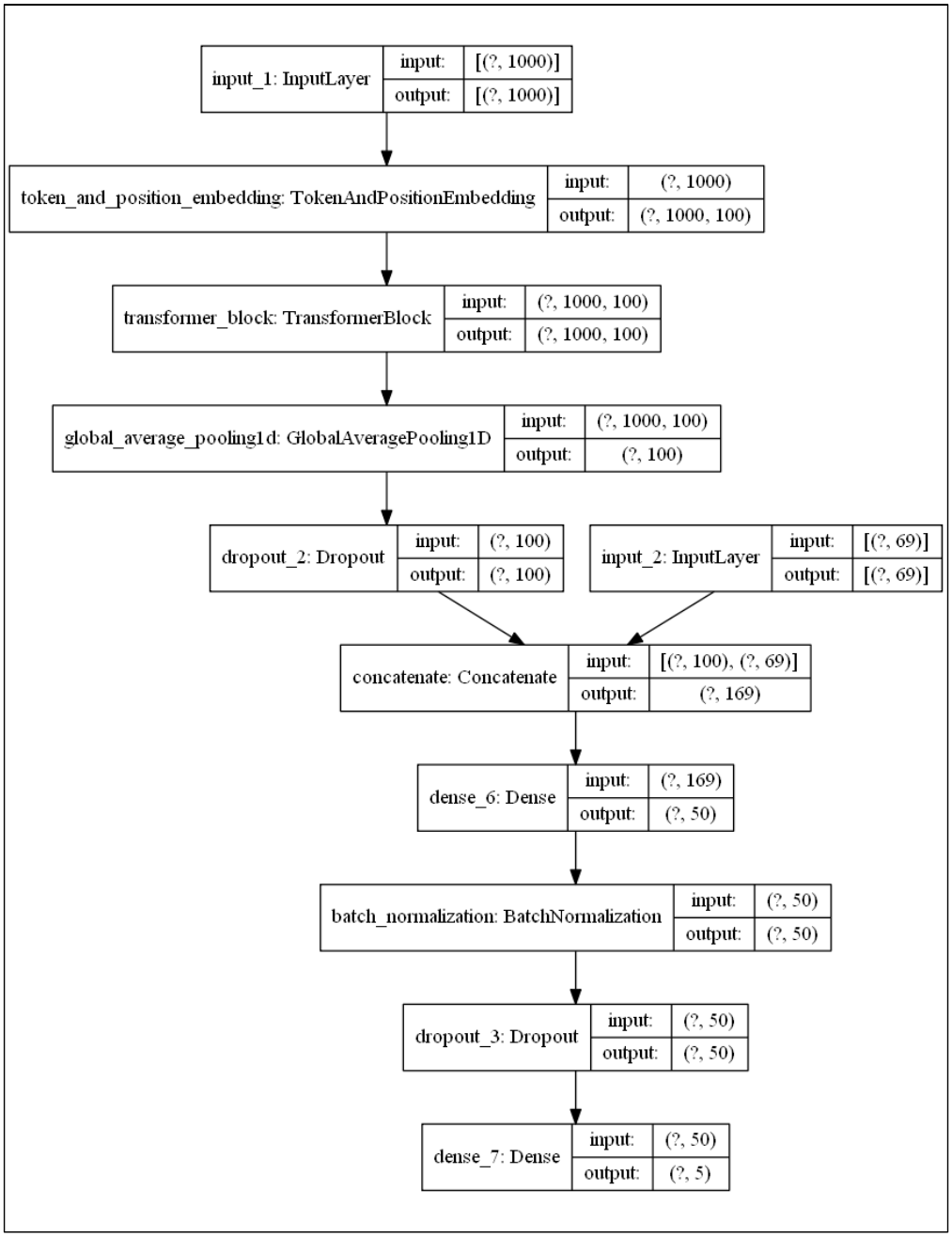
1) 특허분류를 위한 트랜스포머 분류모델의 구성

가) 특허분류용 트랜스포머 분류모델

트랜스포머 기본 모델을 위해 TokenAndPositionEmbedding 층을 사용하였다. 이를 통과한 출력은 트랜스포머 블럭을 통과한 후 완전 출력층을 만들기 위해 Global Average Pooling 1D Layer를 통과하게 된다. 이를 드롭아웃 층, Dense 층, 배치정규화 층을 통과하면서 최종 출력층을 통과하게 된다.

트랜스포머 모델은 기존의 순환신경망과 달리 병렬처리도 가능한 구조이므로 멀티헤드를 2 또는 5를 적용하여 모델을 구성하였다.

[그림 3-26]은 본 연구에서 사용된 트랜스포머의 기본 구조를 나타내었다.



[그림 3-26] 데이터셋 #1 에 적용된 트랜스포머 분류모델의 구조 예시

2) 매개변수 구성

[표 3-59] 트랜스포머 의 매개변수 종류 및 적용값

매개변수 종류	적용값
입력셋	입력#1, 입력#2, 입력#3
Position-wise FFNN(Feed Forward Neural Networks)	32, 64
Heads 수	2, 5
학습률	$1e^{-3}$, $1e^{-4}$
배치사이즈	8, 16
드롭아웃 비율	0.2, 0.3

3) 실험 조건

본 연구에서는 Python 3.8 버전, Tensorflow 2.3.0 버전을 사용하였고, Keras 2.4 버전을 이용하여 프로그램을 작성하였다.

각 모델은 총 150 에폭씩 훈련하며 조기종료 기법을 적용하였다. 조기종료의 조건은 매 에폭마다 검증데이터의 손실값(loss)를 측정하여 10 에폭이 지나도 손실값이 감소하지 않으면 종료시키도록 하였다.

하나의 데이터 셋에 사용되는 모델은 입력셋(3), 하이퍼파라미터 조합(32) 적용하여 총 96개 모델에 대해서 훈련시켰으며 이 모델 중 검증데이터의 손실 값이 가장 낮은 분류모델, 검증 데이터의 정확도가 가장 높은 분류모델 그리고 훈련데이터의 손실값과 검증데이터의 손실값의 차가 가장 적은 분류모델을 각각 선택하였다. 즉, 각 데이터셋 당 3개의 최적 트랜스포머 분류모델을 선정하여 테스트 데이터에 대한 정확도를 측정하였다.

4) 실험 결과

가) 데이터셋 #1 에 트랜스포머 분류모델을 적용한 분류 결과

[표 3-60] 데이터셋 #1에 최적인 트랜스포머 분류모델

	입력	FFNN	Num Head	학습률	배치	에폭	검증 정확도	검증 손실값
TSF#1-1	입력#2	64	5	$1e^{-4}$	8	75	0.938	0.2854
TSF#1-2	입력#3	64	3	$1e^{-3}$	8	7	0.938	0.3592
TSF#1-3	입력#2	64	3	$1e^{-3}$	8	8	0.925	0.2578

데이터셋 #1의 테스트 데이터에 대한 정확도를 비교하면 TSF#1-3 분류 모델의 정확도가 89.00% 이며 TSF#1-1 과 TSF#1-2에 비해 각각 3.0%, 4.5% 높은 정확도를 나타냈다.

[표 3-61] TSF#1-1 분류모델의 테스트 데이터 분류 결과

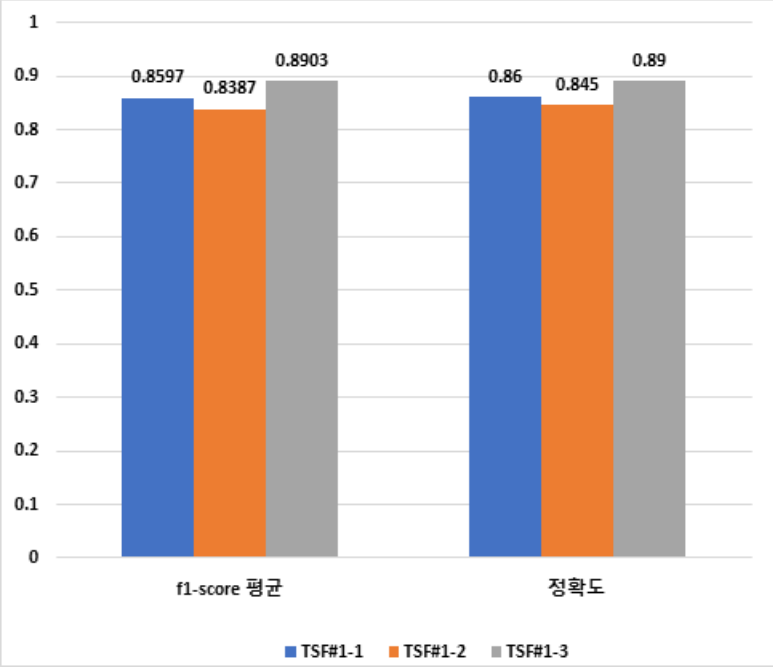
	precision	recall	f1-score	support
C01B	0.7895	0.7500	0.7692	40
C01C	0.9024	0.9250	0.9136	40
C01D	0.9211	0.8750	0.8974	40
C01F	0.8750	0.8750	0.8750	40
C01G	0.8140	0.8750	0.8434	40
accuracy			0.8600	200
macro avg	0.8604	0.8600	0.8597	200

[표 3-62] TSF#1-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8750	0.5250	0.6563	40
C01C	0.9744	0.9500	0.9620	40
C01D	0.7037	0.9500	0.8085	40
C01F	0.8571	0.9000	0.8780	40
C01G	0.8780	0.9000	0.8889	40
accuracy			0.8450	200
macro avg	0.8577	0.8450	0.8387	200

[표 3-63] TSF#1-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8250	0.8250	0.8250	40
C01C	1.0000	0.9500	0.9744	40
C01D	0.8444	0.9500	0.8941	40
C01F	0.8974	0.8750	0.8861	40
C01G	0.8947	0.8500	0.8718	40
accuracy			0.8900	200
macro avg	0.8923	0.8900	0.8903	200



[그림 3-27] 데이터셋#1 테스트데이터 트랜스포머 분류모델 f1-score와 정확도

나) 데이터셋 #2 에 트랜스포머 분류모델을 적용한 분류 결과

[표 3-64] 데이터셋 #2에 최적인 트랜스포머 분류모델

	입력	FFN N	Num Head	학습률	배치	에폭	검증 정확도	검증 손실값

TSF#2-1	입력#2	64	3	1e-4	8	36	0.960	0.2000
TSF#2-2	입력#2	64	3	1e-3	8	16	0.964	0.2576
TSF#2-3	입력#3	64	3	1e-3	16	9	0.960	0.1837

데이터셋 #2의 테스트 데이터에 대한 정확도를 비교하면 TSF#2-2 분류 모델의 정확도가 95.18% 이며 TSF#2-1 과 TSF#2-3에 비해 각각 0.18%, 0.89% 높은 정확도를 나타냈다.

[표 3-65] TSF#2-1 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.9048	0.9500	0.9268	40
H01C	0.9744	0.9500	0.9620	40
H01F	0.9231	0.9000	0.9114	40
H01G	0.8636	0.9500	0.9048	40
H01H	0.9750	0.9750	0.9750	40
H01J	1.0000	0.9750	0.9873	40
H01K	0.9756	1.0000	0.9877	40
H01L	0.8864	0.9750	0.9286	40
H01M	0.9737	0.9250	0.9487	40
H01P	1.0000	0.9000	0.9474	40
H01Q	0.8864	0.9750	0.9286	40
H01R	0.9714	0.8500	0.9067	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9500	560
macro avg	0.9524	0.9500	0.9502	560

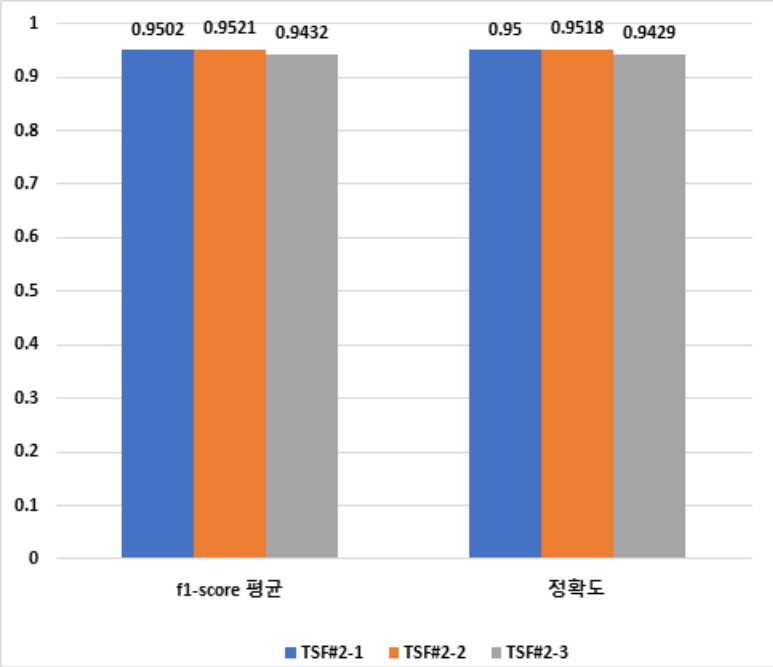
[표 3-66] TSF#2-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.9268	0.9500	0.9383	40
H01C	0.9737	0.9250	0.9487	40
H01F	0.9744	0.9500	0.9620	40
H01G	0.8444	0.9500	0.8941	40
H01H	0.9750	0.9750	0.9750	40
H01J	0.9744	0.9500	0.9620	40
H01K	0.9756	1.0000	0.9877	40
H01L	0.9268	0.9500	0.9383	40
H01M	0.9487	0.9250	0.9367	40
H01P	0.9737	0.9250	0.9487	40
H01Q	0.9500	0.9500	0.9500	40

H01R	0.9048	0.9500	0.9268	40
H01S	1.0000	0.9500	0.9744	40
H01T	1.0000	0.9750	0.9873	40
accuracy			0.9518	560
macro avg	0.9534	0.9518	0.9521	560

[표 3-67] TSF#2-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.8409	0.9250	0.8810	40
H01C	0.9744	0.9500	0.9620	40
H01F	1.0000	0.9000	0.9474	40
H01G	0.9048	0.9500	0.9268	40
H01H	0.9750	0.9750	0.9750	40
H01J	0.8478	0.9750	0.9070	40
H01K	0.9524	1.0000	0.9756	40
H01L	0.9459	0.8750	0.9091	40
H01M	0.9487	0.9250	0.9367	40
H01P	1.0000	0.9000	0.9474	40
H01Q	0.9286	0.9750	0.9512	40
H01R	0.9474	0.9000	0.9231	40
H01S	1.0000	0.9750	0.9873	40
H01T	0.9750	0.9750	0.9750	40
accuracy			0.9429	560
macro avg	0.9458	0.9429	0.9432	560



[그림 3-28] 데이터셋#2 테스트데이터 트랜스포머 분류모델 f1-score와 정확도

다) 데이터셋 #3 에 트랜스포머 분류모델을 적용한 분류 결과

[표 3-68] 데이터셋 #3에 최적인 트랜스포머 분류모델

	입력	FFN N	Num Head	학습률	배치	에폭	검증 정확도	검증 손실값
TSF#3-1	입력#3	64	3	$1e^{-4}$	8	21	0.944	0.2574
TSF#3-2	입력#2	64	3	$1e^{-4}$	8	21	0.944	0.2381
TSF#3-3	입력#2	64	3	$1e^{-3}$	8	7	0.951	0.3083

데이터셋 #3의 테스트 데이터에 대한 정확도를 비교하면 TSF#3-2 분류 모델의 정확도가 95.18% 이며 TSF#3-1 과 TSF#3-3에 비해 각각 0.18%, 0.89% 높은 정확도를 나타냈다.

[표 3-69] TSF#3-1 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8636	0.9500	0.9048	40
C01C	0.9474	0.9000	0.9231	40
C01D	0.9048	0.9500	0.9268	40
C01F	0.8889	0.8000	0.8421	40
C01G	0.8750	0.8750	0.8750	40
H01B	0.9024	0.9250	0.9136	40
H01C	0.9744	0.9500	0.9620	40
H01F	0.9744	0.9500	0.9620	40
H01G	0.9070	0.9750	0.9398	40
H01H	0.9512	0.9750	0.9630	40
H01J	0.9500	0.9500	0.9500	40
H01K	0.9756	1.0000	0.9877	40
H01L	0.9250	0.9250	0.9250	40
H01M	0.8605	0.9250	0.8916	40
H01P	0.9737	0.9250	0.9487	40
H01Q	0.9070	0.9750	0.9398	40
H01R	1.0000	0.8250	0.9041	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9342	760
macro avg	0.9358	0.9342	0.9340	760

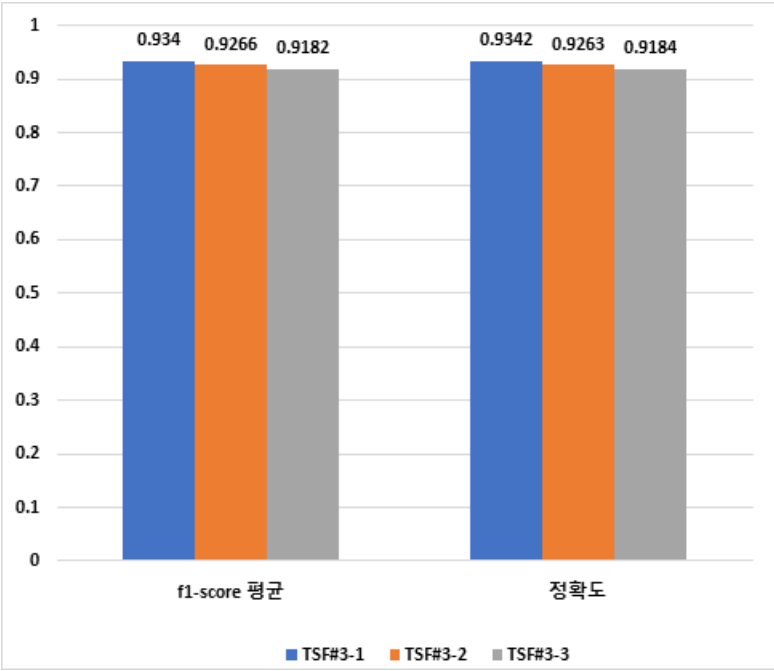
[표 3-70] TSF#3-2 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8000	0.9000	0.8471	40
C01C	0.9730	0.9000	0.9351	40
C01D	0.9268	0.9500	0.9383	40
C01F	0.8500	0.8500	0.8500	40
C01G	0.8974	0.8750	0.8861	40
H01B	0.8182	0.9000	0.8571	40
H01C	0.9750	0.9750	0.9750	40
H01F	0.9737	0.9250	0.9487	40
H01G	0.8605	0.9250	0.8916	40
H01H	0.9512	0.9750	0.9630	40
H01J	0.9744	0.9500	0.9620	40
H01K	1.0000	1.0000	1.0000	40
H01L	0.9250	0.9250	0.9250	40
H01M	0.8919	0.8250	0.8571	40
H01P	0.9459	0.8750	0.9091	40
H01Q	0.9070	0.9750	0.9398	40
H01R	1.0000	0.8750	0.9333	40

H01S	0.9756	1.0000	0.9877	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9263	760
macro avg	0.9287	0.9263	0.9266	760

[표 3-71] TSF#3-3 분류모델의 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8293	0.8500	0.8395	40
C01C	0.9211	0.8750	0.8974	40
C01D	0.9048	0.9500	0.9268	40
C01F	0.8718	0.8500	0.8608	40
C01G	0.8409	0.9250	0.8810	40
H01B	0.8571	0.9000	0.8780	40
H01C	0.9744	0.9500	0.9620	40
H01F	0.9744	0.9500	0.9620	40
H01G	0.8636	0.9500	0.9048	40
H01H	0.9744	0.9500	0.9620	40
H01J	0.8889	1.0000	0.9412	40
H01K	0.9756	1.0000	0.9877	40
H01L	1.0000	0.8250	0.9041	40
H01M	0.8611	0.7750	0.8158	40
H01P	0.9231	0.9000	0.9114	40
H01Q	0.8864	0.9750	0.9286	40
H01R	0.9444	0.8500	0.8947	40
H01S	1.0000	0.9750	0.9873	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9184	760
macro avg	0.9206	0.9184	0.9182	760



[그림 3-29] 데이터셋#3 테스트데이터 트랜스포머 분류모델 f1-score와 정확도

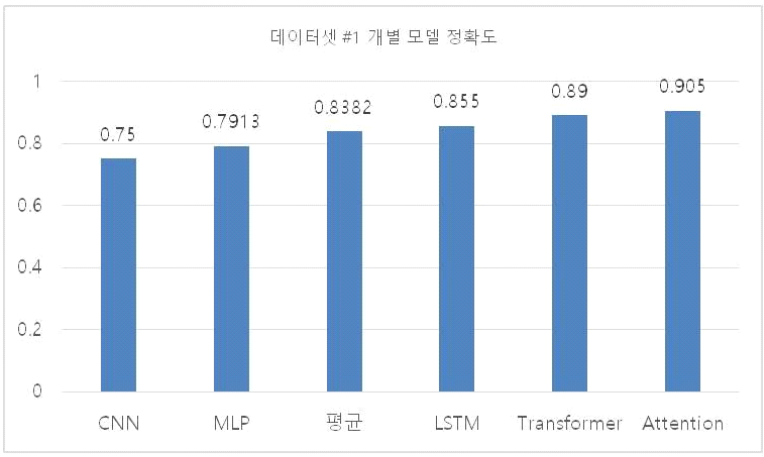
제 7 절 개별 모델간 정확도 비교

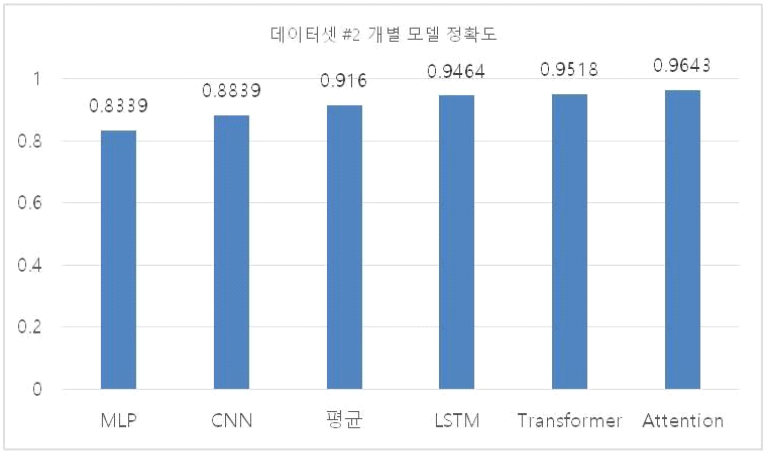
[그림 3-30]은 3가지 데이터셋에 대한 개별 모델의 정확도를 보여준다. 각 데이터셋 당 하나의 딥러닝 알고리즘에는 3가지 분류모델이 만들어 진다. 3가지 분류모델 중 테스트 데이터의 분류 정확도가 가장 높은 분류모델을 선택하여 각 딥러닝 알고리즘의 대표 분류모델로 선택하였으며, 분류 정확도의 산술 평균을 추가하여 각 분류모델의 정확도 순으로 정렬하였다.

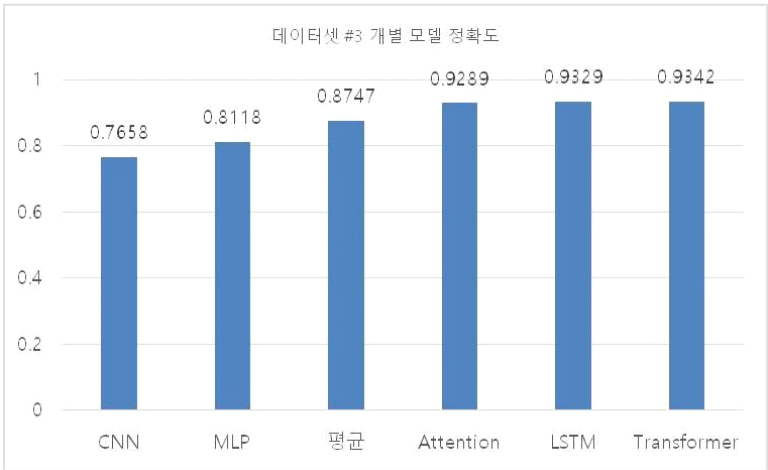
데이터셋 #1 의 5가지 분류모델의 분류 정확도 평균은 85.5% 이다. 분류 정확도가 가장 낮은 분류모델은 CNN 으로 75.0%를 기록하였고 어텐션 분류모델은 90.5%로 최고 정확도를 기록하였다. 분류 정확도 순서는 어텐션(90.5%), 트랜스포머(89.0%), LSTM(85.5%), MLP(79.13%), CNN(75.0%) 순이다. 이중 CNN 과 MLP 의 분류 정확도는 는 평균 이하에 위치하고, LSTM, 트랜스포머, 어텐션의 분류 정확도는 평균을 상회하는 정확도를 기록하였다.

데이터셋 #2 의 5가지 분류모델의 분류 정확도의 평균은 91.6% 이다. 분류 정확도가 가장 낮은 모델은 MLP 모델로 83.39%를 기록하였고 어텐션 모델은 96.43%로 최고 정확도를 기록하였다. 분류 정확도 순서는 어탠션(96.43%), 트랜스포머(95.18%), LSTM(94.64%), CNN(88.39%), MLP(83.39%) 순이다. 이중 MLP 와 CNN 의 분류 정확도는 는 평균 이하에 위치하고, LSTM, 트랜스포머, 어텐션의 분류 정확도는 평균을 상회하는 정확도를 기록하였다.

데이터셋 #3 의 5가지 분류모델의 분류 정확도의 평균은 87.47% 이다. 분류 정확도가 가장 낮은 모델은 CNN 으로 76.58%를 기록하였고 트랜스포머 모델은 93.42%로 최고 정확도를 기록하였다. 분류 정확도 순서는 트랜스포머(93.42%), LSTM(93.29%), 어텐션(92.89%), MLP(81.18%), CNN(76.58%) 순이다. 이중 CNN 와 MLP 의 분류 정확도는 는 평균 이하에 위치하고, 어텐션, LSTM, 트랜스포머의 분류 정확도는 평균을 상회하는 정확도를 기록하였다.







[그림 3-30] 3가지 데이터셋의 개별모델 정확도

3가지 데이터셋에 대한 결과에서 문장 기반의 분류모델인 LSTM, Transformer, Attention 분류모델이 키워드 기반의 분류모델인 MLP, CNN 모델 보다 높은 정확도를 기록하였다. MLP, CNN의 분류 정확도는 산술 평균 이하에 위치하였고, LSTM, 트랜스포머, 어텐션모델은 평균 이상에 위치한다. 이러한 경향은 3가지 데이터셋 모두에서 보인다.

데이터셋 #1과 데이터셋 #2 에서는 두가지 경우 모두 어텐션 모델이 가장 높은 정확도를 보였고, 트랜스포머, LSTM 모델 순이었다. 반면 데이터셋 #3 에서는 트랜스포머 모델이 가장 높은 정확도를 보였고 어텐션 모델은 3순위로 하락함을 보였다.

본 연구에서 사용한 3가지 데이터셋으로 한정하여 특허문헌의 분류 결과를 분석한다면 키워드 기반의 분류 보다는 문장 기반의 분류가 높은 정확도를 보였다. 문장 기반의 3가지 분류 모델 중 어텐션 모델이 2가지 경우(데이터셋 #1, 데이터셋 #2)에서 1위를 기록하였지만, 1가지 경우(데이터셋 #3)에서는 분류 정확도 순으로 3위를 기록하였다. 트랜스포머 모델은 2가지 경우(데이터셋 #1, 데이터셋 #2)에서 2위를 기록하였지만, 1가지 경우(데이터셋 #3)의 경우에는 1위를 기록하였다.

3가지 한정된 데이터셋으로 실험한 결과를 바탕으로 최적의 모델을 선정한다고 한다면 어텐션 또는 트랜스포머 분류모델이 선택 될 수 있을 것이다. 그러나 각 데이터셋에서 어텐션 또는 트랜스포머에서 사용한 매개변수는 모두 상이 하였다. 데이터셋 #1 과 데이터셋 #2에서 최고의 분류 정확도를 나타내는 어텐션 분류모델의 매개변수가 다르므로 이는 별개의 분류모델로 봐야 한다. 즉 3가지 데이터셋에서 모두 최고의 분류정확도를 나타내는 분류 모델은 존재하지 않았다.

테스트 데이터에서의 분류 정확도 순위와 검증데이터에서의 분류 정확도 순위는 일치하지 않는 경우가 많았다. 검증데이터의 분류 정확도를 이용해서 테스트 데이터에서 최고의 정확도를 가지는 분류모델을 선택할 수 없었다.

본 연구의 목적인 다양한 사용자 분류체계에 적용될 특허분류 모델의 구성을 위해, 하나의 특정 분류모델을 선택하는 방법으로는 목적을 달성할 수

없다고 판단된다. 이를 다음장 에서 설명하는 앙상블 기법으로 해결가능한지 살펴보고자 한다.

제 4 장 앙상블 기법을 이용한 특허문서 분류

제 1 절 앙상블 기법을 이용한 특허 분류

앙상블 기법의 아이디어는 여러 모델을 통합하여 예측 모델을 구축하는 것이다. 앙상블 기법을 사용하여 예측 성능을 향상시킬 수 있다는 것은 잘 알려져 있다.(Rokach, 2010)

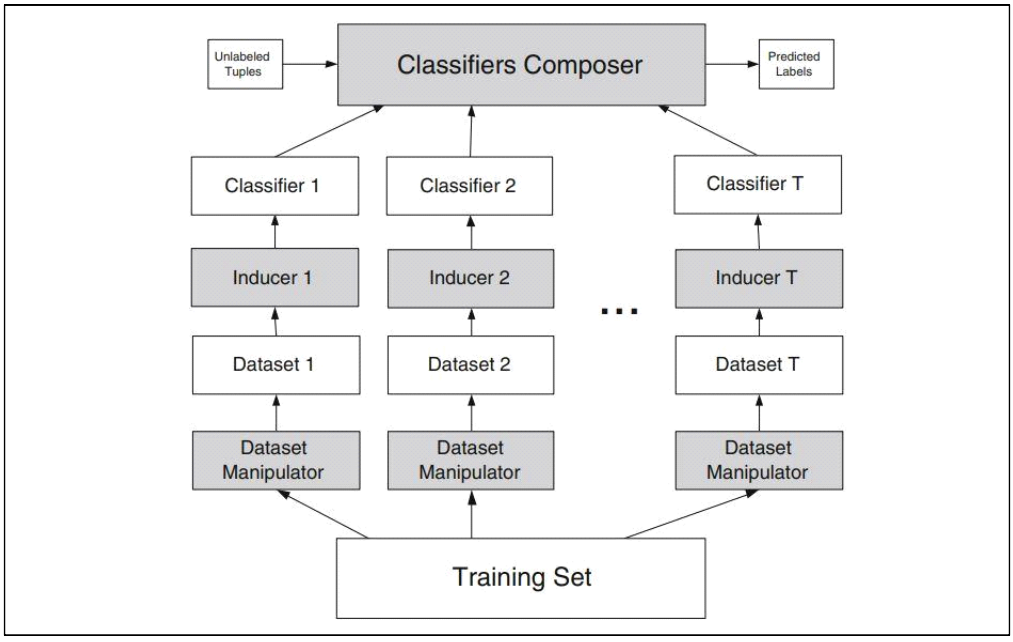
3장에서 만들어진 특허문서 분류를 위한 5개의 딥러닝 알고리즘 (MLP, CNN, LSTM, 어텐션, 트랜스포머)으로부터 만들어진 분류모델을 3가지의 앙상블 기법을 이용하여 각 데이터셋의 테스트 데이터의 분류 정확도를 측정하여 개별 분류모델과의 정확도를 비교하였다.

앙상블 기법은 개별 모델의 출력결과가 다음 모델의 구성에 사용되는 종속 프레임워크방식이 있으며 대표적인 예로 부스팅 기법이 있다. 또한 각 분류 모델이 독립적으로 구축되고 결과가 개별적으로 출력되며 이 출력의 결과를 다양한 방식으로 결합하는 독립 프레임워크 방식이 있으며 대표적인 예로 Bagging(배깅) 기법이 있다.

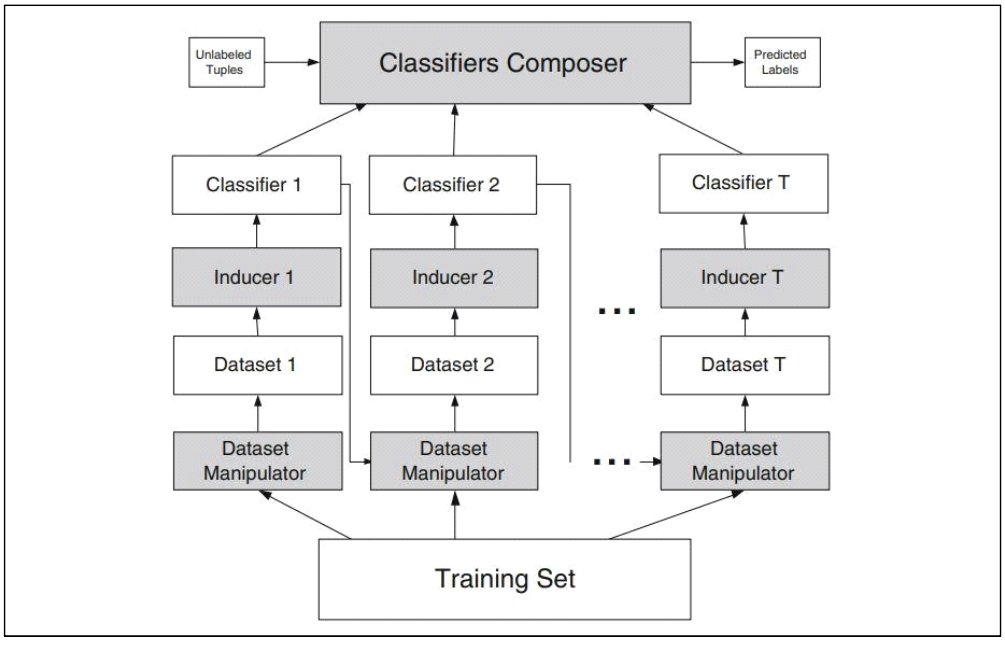
가장 잘 알려진 독립 프레임워크 방식은 배깅이다. 학습된 분류 모델의 다양한 출력을 단일 예측으로 병합하여 향상된 복합 분류 모델을 생성하여 정확도를 높이는 것이다.

Require: I (an inducer), T (the number of iterations), S (the training set), μ (the subsample size).
Ensure: $M_t; t = 1, \ldots, T$
1: $t \leftarrow 1$
2: **repeat**
3: $S_t \leftarrow$ Sample μ instances from S with replacement.
4: Build classifier M_t using I on S_t
5: $t++$
6: **until** $t > T$

[그림 4-1] The bagging algorithm(Breiman, 1996)



[그림 4-2] Independent methods (Rokach, 2010)



[그림 4-3] Dependent methods (Rokach, 2010)

1) 독립 프레임워크 앙상블 기법인 Summation, Weighting, Voting 기법

가) Summation 기법

Summation 기법은 각 개별모델이 분류체계별로 출력하는 출력값의 전체 합을 이용해서 출력값의 합이 가장 큰 값을 이용해서 분류하는 기법이다. 본 연구에서는 총 5개의 알고리즘을 사용하고 각 알고리즘 별로 3개의 모델을 선택한다. 총 15개의 분류 모델의 출력값의 합을 이용해서 분류하는 Summation 기법을 사용하였다.

나) Weighting 기법

Weighting 기법은 각 모델이 출력하는 출력값에 특정 지표를 가중치로 사용하여 출력값에 가중치를 부여하고 이들의 합을 이용해서 합이 가장 큰 값을 이용해서 분류하는 기법이다. 본 연구에서는 가중치로 사용되는 특정지표로 각 모델의 검증데이터의 정확도를 사용하였다. 총 5개의 알고리즘을 사용하고 각 알고리즘 별로 3개의 모델을 선택한다. 총 15개의 분류 모델의 출력값과 검증데이터의 정확도의 곱을 이용해서 분류하는 Weighting 기법을 사용하였다.

다) Voting 기법

Voting 기법은 각 개별 모델간 출력값 및 가중치에 의한 값과 상관없이 개별 모델이 선택한 분류체계에 대한 결정값에 대해 동등하게 처리하는 기법이다. 즉 하나의 특허문헌에 대해 15개의 모델이 가장 많이 선택한 분류체계를 Voting 기법이 선택한 분류로 확정하는 것이다. 총 5개의 알고리즘을 사용하고 각 알고리즘 별로 3개의 모델을 선택한다. 총 15개의 분류 모델이 가장 많이 선택한 분류체계를 최종 분류로 선택하는 Voting 기법을 사용하였다.

2) 데이터셋 #1 의 앙상블 기법을 이용한 분류 정확도

데이터셋 #1의 훈련을 통해 만들어진 15개의 개별 분류모델을 Summation, Weighting, Voting 의 3가지 앙상블 기법으로 테스트 데이터의 분류 정확도를 측정하였다. 각 기법이 적용된 모델을 각각 Summation#1, Weighting#1, Voting#1 이라 명명하겠다.

[표 4-1] Summation#1 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8974	0.8750	0.8861	40
C01C	0.9500	0.9500	0.9500	40
C01D	0.8780	0.9000	0.8889	40
C01F	0.9474	0.9000	0.9231	40
C01G	0.8333	0.8750	0.8537	40
accuracy			0.9000	200
macro avg	0.9012	0.9000	0.9003	200

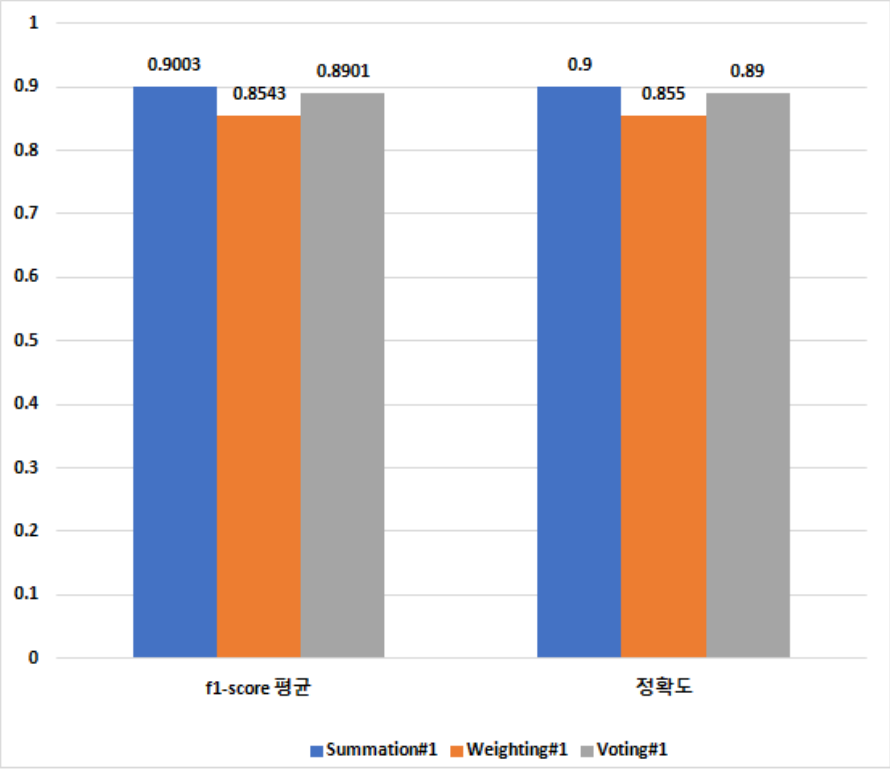
[표 4-2] Weighting#1 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8750	0.7000	0.7778	40
C01C	0.9000	0.9000	0.9000	40
C01D	0.8222	0.9250	0.8706	40
C01F	0.9459	0.8750	0.9091	40
C01G	0.7609	0.8750	0.8140	40
accuracy			0.8550	200
macro avg	0.8608	0.8550	0.8543	200

[표 4-3] Voting#1 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8947	0.8500	0.8718	40
C01C	0.9231	0.9000	0.9114	40
C01D	0.8810	0.9250	0.9024	40
C01F	0.9231	0.9000	0.9114	40
C01G	0.8333	0.8750	0.8537	40
accuracy			0.8900	200
macro avg	0.8910	0.8900	0.8901	200

데이터셋 #1의 테스트데이터에 대한 정확도를 비교하면 Summation#1 모델의 정확도가 90.0% 였으며 Voting#1 과 Weighting#3 에 비해 각각 1.0%, 4.5% 높은 정확도를 나타냈다.



[그림 4-4] 데이터셋 #1 앙상블 모델의 f1-score 와 정확도

3) 데이터셋 #2 의 앙상블 기법을 이용한 분류 정확도

데이터셋 #2의 훈련을 통해 만들어진 15개의 개별 분류모델을 Summation, Weighting, Voting 의 3가지 앙상블 기법으로 테스트 데이터의 분류 정확도를 측정하였다. 각 기법이 적용된 모델을 각각 Summation#2, Weighting#2, Voting#2 이라 명명하겠다.

[표 4-4] Summation#2 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.8837	0.9500	0.9157	40
H01C	0.9750	0.9750	0.9750	40
H01F	1.0000	0.9000	0.9474	40
H01G	0.8837	0.9500	0.9157	40
H01H	0.9512	0.9750	0.9630	40
H01J	1.0000	1.0000	1.0000	40
H01K	1.0000	1.0000	1.0000	40
H01L	0.9512	0.9750	0.9630	40
H01M	0.9737	0.9250	0.9487	40
H01P	1.0000	0.9250	0.9610	40
H01Q	0.9091	1.0000	0.9524	40
H01R	0.9730	0.9000	0.9351	40
H01S	1.0000	1.0000	1.0000	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9625	560
macro avg	0.9643	0.9625	0.9626	560

[표 4-5] Weighting#2 테스트 데이터 분류 결과

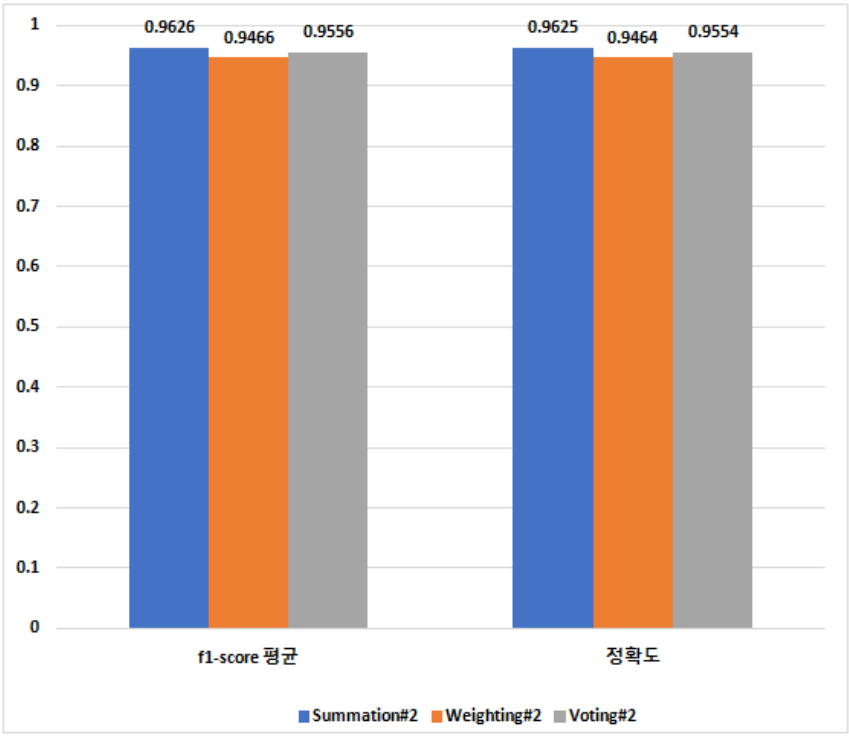
	precision	recall	f1-score	support
H01B	0.9024	0.9250	0.9136	40
H01C	0.9737	0.9250	0.9487	40
H01F	0.9487	0.9250	0.9367	40
H01G	0.8667	0.9750	0.9176	40
H01H	0.9512	0.9750	0.9630	40
H01J	0.9744	0.9500	0.9620	40
H01K	1.0000	1.0000	1.0000	40
H01L	0.8864	0.9750	0.9286	40
H01M	0.9737	0.9250	0.9487	40
H01P	0.9714	0.8500	0.9067	40
H01Q	0.8837	0.9500	0.9157	40

H01R	0.9487	0.9250	0.9367	40
H01S	1.0000	1.0000	1.0000	40
H01T	1.0000	0.9500	0.9744	40
accuracy			0.9464	560
macro avg	0.9486	0.9464	0.9466	560

[표 4-6] Voting#2 테스트 데이터 분류 결과

	precision	recall	f1-score	support
H01B	0.8837	0.9500	0.9157	40
H01C	0.9744	0.9500	0.9620	40
H01F	1.0000	0.9000	0.9474	40
H01G	0.8636	0.9500	0.9048	40
H01H	0.9512	0.9750	0.9630	40
H01J	1.0000	1.0000	1.0000	40
H01K	1.0000	1.0000	1.0000	40
H01L	0.9070	0.9750	0.9398	40
H01M	0.9737	0.9250	0.9487	40
H01P	1.0000	0.9000	0.9474	40
H01Q	0.8889	1.0000	0.9412	40
H01R	0.9722	0.8750	0.9211	40
H01S	1.0000	1.0000	1.0000	40
H01T	1.0000	0.9750	0.9873	40
accuracy			0.9554	560
macro avg	0.9582	0.9554	0.9556	560

데이터셋 #2의 테스트데이터에 대한 정확도를 비교하면 Summation#2 모델의 정확도가 96.25% 였으며 Voting#2 와 Weighting#2 에 비해 각각 0.71%, 1.61% 높은 정확도를 나타냈다.



[그림 4-5] 데이터셋 #2의 앙상블 모델의 f1-score 와 정확도

4) 데이터셋 #3 의 앙상블 기법을 이용한 분류 정확도

데이터셋 #3의 훈련을 통해 만들어진 15개의 개별 분류모델을 Summation, Weighting, Voting 의 3가지 앙상블 기법으로 테스트 데이터의 분류 정확도를 측정하였다. 각 기법이 적용된 모델을 각각 Summation#3, Weighting#3, Voting#3 이라 명명하겠다.

[표 4-7] Summation#3 테스트 데이터 분류 결과

	precision	recall	f1-score	support
C01B	0.8780	0.9000	0.8889	40
C01C	1.0000	0.9500	0.9744	40
C01D	0.8864	0.9750	0.9286	40
C01F	0.8974	0.8750	0.8861	40
C01G	0.9189	0.8500	0.8831	40
H01B	0.8810	0.9250	0.9024	40
H01C	0.9744	0.9500	0.9620	40

H01F	1.0000	0.9250	0.9610	40
H01G	0.8636	0.9500	0.9048	40
H01H	0.9286	0.9750	0.9512	40
H01J	0.9750	0.9750	0.9750	40
H01K	1.0000	1.0000	1.0000	40
H01L	0.9487	0.9250	0.9367	40
H01M	0.8750	0.8750	0.8750	40
H01P	0.9730	0.9000	0.9351	40
H01Q	0.8889	1.0000	0.9412	40
H01R	1.0000	0.9000	0.9474	40
H01S	1.0000	1.0000	1.0000	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9395	760
macro avg	0.9415	0.9395	0.9396	760

[표 4-8] Weighting#3 테스트 데이터 분류 결과

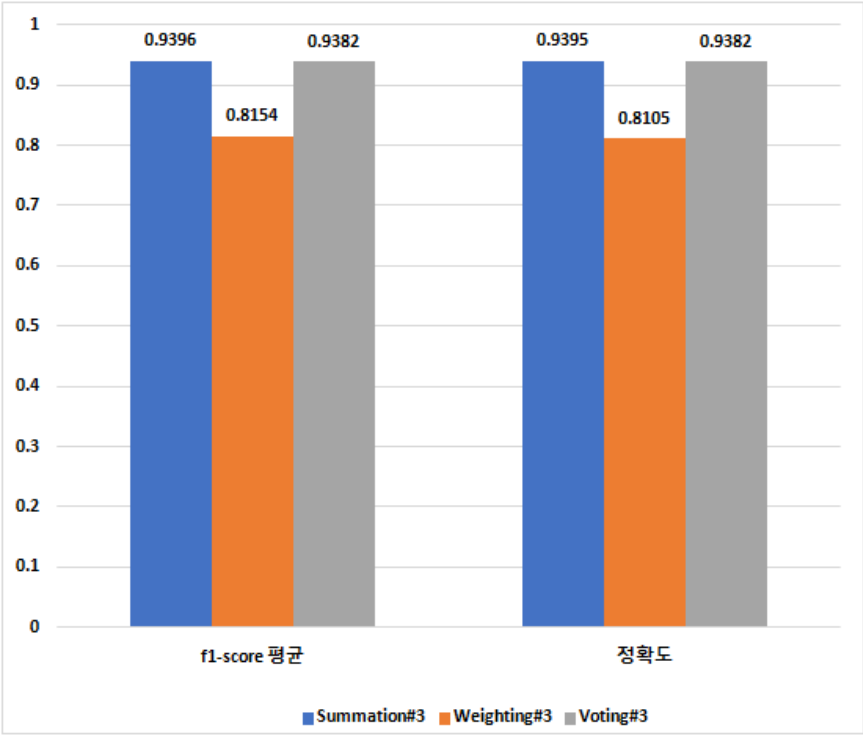
	precision	recall	f1-score	support
C01B	0.8182	0.4500	0.5806	40
C01C	0.8444	0.9500	0.8941	40
C01D	0.4000	0.9500	0.5630	40
C01F	0.8824	0.7500	0.8108	40
C01G	0.8438	0.6750	0.7500	40
H01B	0.7632	0.7250	0.7436	40
H01C	0.8780	0.9000	0.8889	40
H01F	1.0000	0.8500	0.9189	40
H01G	0.8824	0.7500	0.8108	40
H01H	0.7872	0.9250	0.8506	40
H01J	0.8919	0.8250	0.8571	40
H01K	0.9730	0.9000	0.9351	40
H01L	0.9231	0.6000	0.7273	40
H01M	0.8438	0.6750	0.7500	40
H01P	0.7778	0.8750	0.8235	40
H01Q	0.8409	0.9250	0.8810	40
H01R	0.9412	0.8000	0.8649	40
H01S	0.8605	0.9250	0.8916	40
H01T	0.9500	0.9500	0.9500	40
accuracy			0.8105	760
macro avg	0.8474	0.8105	0.8154	760

[표 4-9] Voting#3 테스트 데이터 분류 결과

	precision	recall	f1-score	support

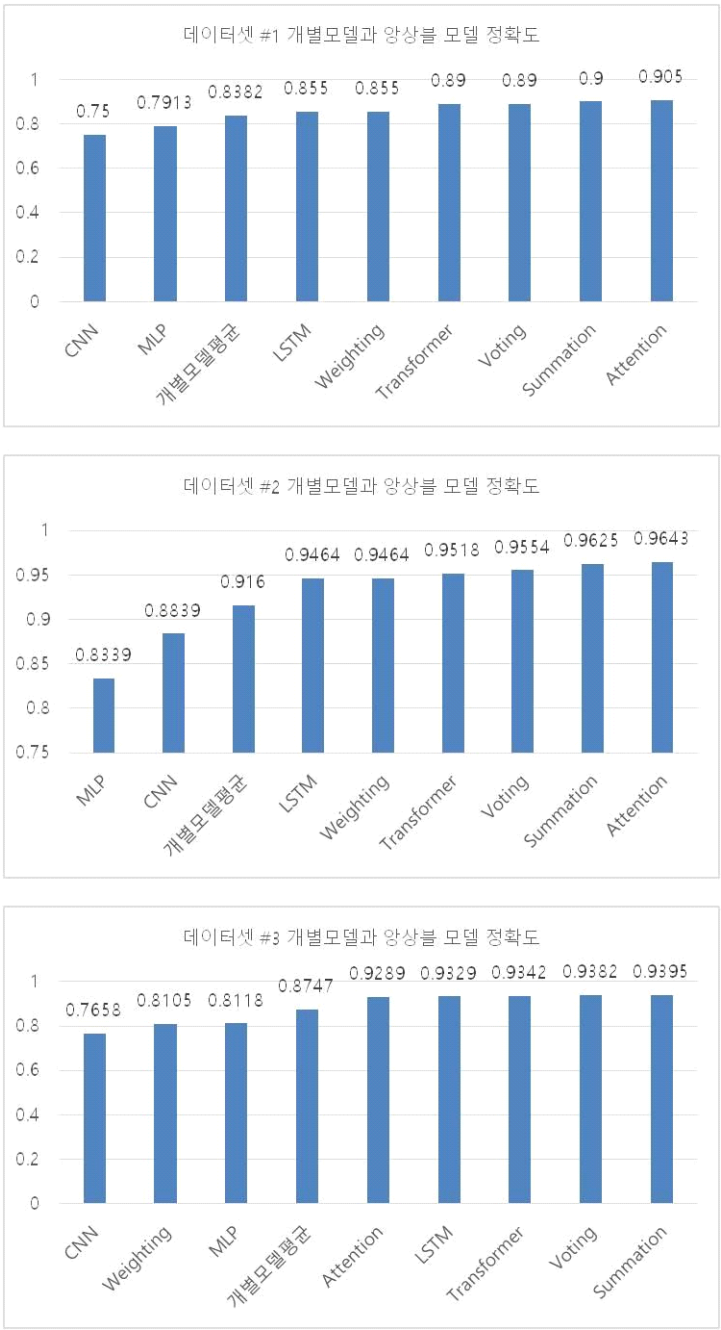
C01B	0.8810	0.9250	0.9024	40
C01C	1.0000	0.9500	0.9744	40
C01D	0.8864	0.9750	0.9286	40
C01F	0.9211	0.8750	0.8974	40
C01G	0.9189	0.8500	0.8831	40
H01B	0.8810	0.9250	0.9024	40
H01C	0.9744	0.9500	0.9620	40
H01F	0.9737	0.9250	0.9487	40
H01G	0.8636	0.9500	0.9048	40
H01H	0.9070	0.9750	0.9398	40
H01J	0.9750	0.9750	0.9750	40
H01K	1.0000	1.0000	1.0000	40
H01L	0.9487	0.9250	0.9367	40
H01M	0.8750	0.8750	0.8750	40
H01P	0.9722	0.8750	0.9211	40
H01Q	0.8889	1.0000	0.9412	40
H01R	1.0000	0.8750	0.9333	40
H01S	1.0000	1.0000	1.0000	40
H01T	1.0000	1.0000	1.0000	40
accuracy			0.9382	760
macro avg	0.9404	0.9382	0.9382	760

데이터셋 #3의 테스트데이터에 대한 정확도를 비교하면 Summation#3 모델의 정확도가 93.95% 였으며 Weighting#3 과 Voting#3 에 비해 각각 0.13%, 12.9% 높은 정확도를 나타냈다.



[그림 4-6] 데이터셋 #3의 앙상블 모델의 f1-score 와 정확도

제 2 절 앙상블 기법 과 개별모델 간의 분류 정확도 비교



[그림 4-7] 3가지 데이터셋의 개별모델과 앙상블의 정확도

[그림 4-7]은 3가지 데이터셋의 테스트 데이터의 분류결과에 대해 개별 분류모델의 분류 정확도, 개별 분류모델 정확도의 평균 그리고 3가지 앙상블 모델의 분류 정확도를 정확도 순으로 정렬하여 비교하였다.

데이터셋 #1에서 3가지 앙상블 모델인 Summation#1, Voting#1, Weighting#1 의 분류 정확도는 각각 90.0%, 89.0%, 85.5%를 기록하였다. 데이터셋 #1의 개별 분류모델 중 최고 성능 모델인 어텐션 분류모델의 정확도는 90.5%이고 앙상블 모델 중 최고 성능 모델인 Summation#1 모델의 분류 정확도는 90.0% 이다. 두 모델간의 분류 정확도 차이는 0.5% 이다. 데이터셋 #1의 테스트 데이터는 전체 200건이므로 어텐션 모델은 181건의 정답을 예측했으며 Summation #1은 180건의 정답을 예측하였다.

데이터셋 #2에서 3가지 앙상블 모델인 Summation#2, Voting#2, Weighting#2 의 분류 정확도는 각각 96.25%, 95.54%, 94.64%를 기록하였다. 데이터셋 #2의 개별 모델 중 최고 성능 모델인 어텐션 모델의 정확도는 96.43%이고 앙상블 모델 중 최고 성능 모델인 Summation#2 모델의 분류 정확도는 96.25% 이다. 두 모델간의 분류 정확도 차이는 0.18% 이다. 데이터셋 #2의 테스트 데이터는 전체 560건이므로 Attention 모델은 540건의 정답을 예측했으며 Summation #2은 539건의 정답을 예측하였다.

데이터셋 #3에서 3가지 앙상블 모델인 Summation#3, Voting#3, Weighting#3 의 분류 정확도는 각각 93.95%, 93.82%, 81.05%를 기록하였다. 데이터셋 #3의 개별 모델 중 최고 성능 모델인 트랜스포머 분류모델의 정확도는 93.42%이고 앙상블 모델 중 최고 성능 모델인 Summation#3 모델의 분류 정확도는 93.95% 이다. 두 모델간의 분류 정확도 차이는 0.53% 이다. 데이터셋 #3의 테스트 데이터는 전체 760건이므로 트랜스포머 분류모델은 710건의 정답을 예측했으며 Summation #1은 714건의 정답을 예측하였다.

3가지 데이터셋의 대해 앙상블 기법은 데이터셋 #1 과 데이터셋 #2에 대해서는 테스트 데이터의 분류 정확도에서 2 순위를 기록하였다. 데이터셋 #3에서는 테스트 데이터의 분류 정확도에서 1 순위를 기록하였다. 분류 정확도

의 차이는 각각 0.5%, 0.18%, 0.53% 이다. 건수 기준으로는 데이터셋 #1 에서는 200건의 테스트 데이터 중 1건의 차이로 2순위를 기록하였으며 데이터셋 #2에서는 560건의 테스트 데이터 중 1건의 차이로 2순위를 기록하였다. 그리고 데이터셋 #3에서는 760건의 테스트 데이터 중 4건의 차이로 1순위를 기록하였다. 앙상블 기법 중 Summation 과 Voting 기법을 사용한 모델은 기본 모델 중 최고 정확도를 기록한 개별 모델의 정확도와 거의 유사한 정확도를 보였다. 데이터셋 #1 에서는 각각 2순위와 3순위, 데이터셋 #2에서는 2순위와 3순위 그리고 데이터셋 #3 에서는 1순위와 2순위를 기록하였다.

개별모델과 앙상블 기법의 비교 결과에서 앙상블 기법을 사용한 모델은 앙상블에 참여한 개별 분류모델 중 최고 정확도를 보이는 개별 분류모델의 정확도와 거의 유사한 수준으로 낮거나 또는 조금 높은 정확도를 보인다.

제 3 절 딥러닝 과 앙상블 기법을 이용한 특허문서 분류기

3장에서 실험한 개별 알고리즘 테스트에서 3가지 데이터에 대해 최고의 분류성능을 내는 모델, 입력 필드, 및 매개변수는 모두 상이했다. 즉 하나의 데이터셋에 최고의 분류성능을 내는 분류모델은 다른 데이터셋에서는 최고 분류성능을 내지 못했다.

단일 분류모델을 구성하는 변수는 입력 필드, 분류 알고리즘 그리고 알고리즘 매개변수이다. 본 연구의 목적인 다양한 관점과 목적을 가진 다수의 사용자 정의 분류체계에 대응하는 특허분류를 위해서는 단일 분류모델로서는 원하는 결과를 얻지 못한다고 앞선 실험 결과를 통해 판단하였다.

이런 문제를 해결하기 위해서는 두 가지 접근법이 가능하다. 첫 번째는 사용자 정의 분류체계를 가지는 훈련데이터를 정해진 세트에 따라 훈련하고 측정하여 최적의 모델을 하나 선택하는 것이다. 이런 경우에는 사용자 정의 분류체계가 변경되고 데이터셋이 변경될 때 마다 새로운 분류 모델을 만들어야 한다는 단점이 발생한다.

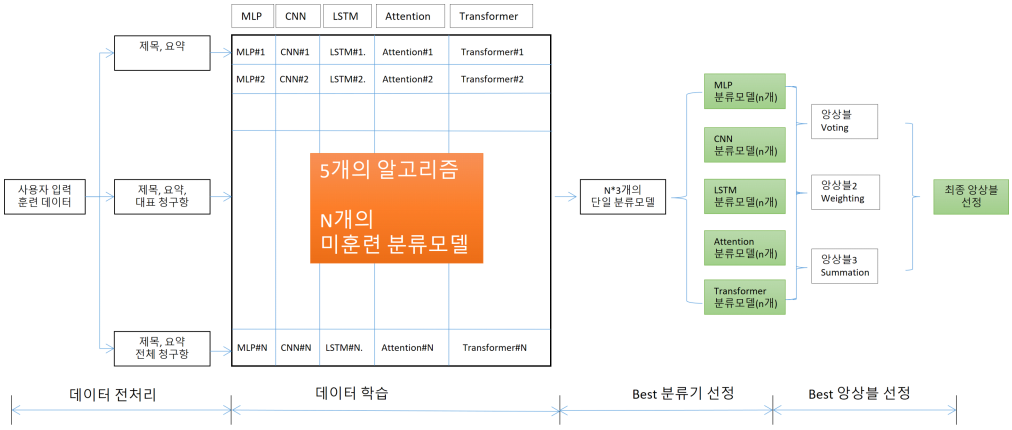
두 번째는 다양한 조합의 예비 분류 모델 집합을 두고 이를 훈련데이터로 훈련 후 기준 이상의 분류 모델 다수를 선택한 후 앙상블 기법을 적용하여 분류를 수행하는 것이다.

본 연구의 3장 과 4장 1절, 2절의 연구결과에서 앙상블 기법을 적용한 분류정확도는 단일 최고 분류성능을 가진 모델과 비교해서 유사한 수준에서 낮거나 높은 분류 정확도를 보임을 확인하였다.

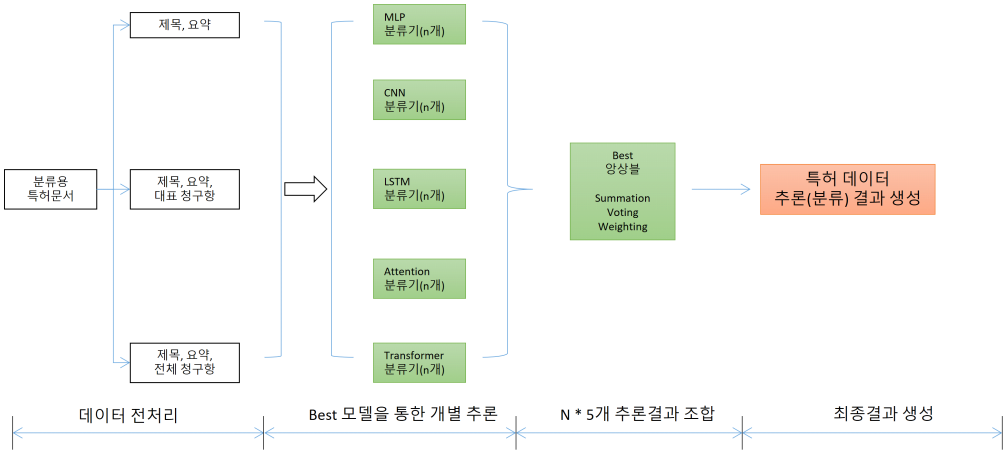
실제 산업현장에서 다양한 사용자 정의 분류 체계를 가진 특허분류기의 요구사항은 다음과 같다. 사용자는 자신들만의 분류 체계를 확립하고 분류 체계별로 일정량의 특허 문서를 수작업으로 분류한 데이터를 훈련데이터로 사용한다. 이러한 훈련데이터를 분류기에 입력하면 자동으로 훈련이 되어야 한다. 또 다른 분류체계로 만들어진 훈련데이터도 동일하게 분류기에 입력하면 자동으로 훈련이 되어야 한다. 그리고 훈련된 모델을 대상으로 분류되지 않은 데이터를 입력하면 자동으로 분류되어야 한다. 그리고 일정 수준 이상의 분류

성능이 보장되어야 한다.

사용자 요구사항과 앞선 실험의 결과를 바탕으로 본 연구에서는 다음과 같은 특허문서 자동분류 아키텍처를 제안한다.



[그림 4-8] 사용자 정의 분류체계 분류를 위한 특허분류기 훈련 아키텍쳐



[그림 4-9] 사용자 정의 분류체계 분류를 위한 특허분류기 추론 아키텍쳐

본 연구에서 제안하는 사용자 정의 분류 체계 분류를 위한 특허문서 분류기 아키텍쳐는 다음과 같다.

사용자 정의 분류 체계에 따른 특허문서 분류기는 사용자가 입력한 특허 데이터에 대해 다양한 분류모델을 통해 훈련시키고 검증데이터의 테스트 기준으로 알고리즘별 3개의 분류기를 선정한다. 그 후 3가지 앙상블 방법

(Summation, Voting, Weighting)으로 검증데이터의 분류 정확도를 측정하고 최종 앙상블 기법을 선정한다. 하나의 사용자 분류체계가 학습되면 다수의 개별분류모델과 어떠한 앙상블 기법을 사용할지가 결정된다. 개별 분류모델에는 입력으로 어떠한 데이터 필드를 사용하는지에 대한 정보도 포함되어 있다. 분류되지 않은 특허문서를 분류시 문서를 3가지 입력필드로 나눈 후 선택된 15개의 분류모델로 분류를 수행하고 각각의 결과를 선택된 앙상블 기법을 사용하여 분류를 수행하게 된다.

본 연구에서 제안한 특허 문서분류 아키텍처는 다음과 같은 장점이 있다.

1. 사용자 분류체계의 구성(분류체계)과 관계없이 적용가능
2. 딥러닝 알고리즘과 매개변수 세트의 추가로 인해 확장이 용이함
3. 개별 모델의 분류성능의 편차를 앙상블 기법을 통해 상향평준화 가능

제 5 장 결론

특허는 기술 혁신에 대한 권리를 보호하는 특성을 가진다. 이러한 특성으로 인해 대부분의 기업에서 특허는 중요한 자산으로 간주된다. 기업들은 자사의 특허를 보호하기 위해, 타사의 제품이 자사의 특허 권리를 침해하였는지 또는 자사의 기술 및 제품이 타사의 특허의 권리를 침해하는지에 대한 정보를 얻기 위해 특허 검색 및 개별 특허의 권리 범위에 대한 개별 분석을 수행한다.

특허는 기술의 흐름 분석에도 사용된다. 개별 특허의 분석이 아니라 산업의 기술발전 흐름, 국가별 기술 경쟁력, 특정 산업분야의 기술 발전 추이 및 경쟁사의 기술 보유 현황 및 경쟁사 기술 개발 방향 등의 분석에도 사용된다. 이러한 군집화 분석에 특허기술 분류는 분석의 기본 재료가 된다.

각국 특허청 및 국제특허기구(WIPO)에서 관리하는 공식적인 특허 분류가 존재하며 이를 통해 많은 분석이 이루어진다. 이러한 공식분류에는 IPC, CPC 등이 존재하며 이를 통해 많은 특허 분석이 이루어진다. 그러나 이러한 공식적인 분류로는 개별 기업이 분석하고자 하는 관점의 분류와 일치하지 않는 경우가 발생할 수 있으므로 이러한 경우 개별 기업에서는 사용자 정의 분류체계를 만들어 사용하게 된다. 이러한 사용자 분류 체계를 통한 분류는 특허 전문가 및 도메인 전문가에 의해 수작업으로 이루어지므로 많은 비용과 시간이 소요된다.

이러한 비용과 시간의 절감을 위해 특허 문서 자동분류는 좋은 대안이 될 수 있고, 이에 관한 다양한 연구가 이루어졌다. 특허를 구성하고 있는 단어 및 문장을 기계학습의 입력으로 사용되는 벡터로 변환하는 방식의 개선을 통해 분류 성능을 향상하는 방법, 다양한 딥러닝 알고리즘을 이용해서 분류 성능을 향상하는 방법 그리고 앙상블 기법을 사용하여 다수의 분류알고리즘의 결과를 결합하여 분류 성능을 향상하는 방법 등의 특허 문서 자동분류와 관련된 연구가 이루어 졌다. 이러한 기존의 대다수 연구는 IPC 등의 공식분류의 분류 성능 향상에 연구 초점이 맞춰져 있었기 때문에 다양한 관점과 목적

을 가진 다양한 사용자 분류체계에 맞는 분류기에 대한 연구에 대해서는 미비했다고 할 수 있다.

본 연구의 목적은 다양한 사용자 분류체계를 처리할 수 있는 자동화된 특허 문서분류기의 아키텍처 및 방법을 제안하는 것이다. 이를 위해 3가지 데이터셋을 선정하여 이들 데이터셋의 분류 성능을 함께 높일 수 있는 방법을 찾고자 하였다. 이를 위해 키워드 관점의 알고리즘인 MLP, CNN 모델을 사용하였고, 문장 관점의 알고리즘인 LSTM, Attention 그리고 Transformer 구조를 사용하였으며 각 알고리즘별로 다수의 하이퍼파라메타를 적용하여 다양한 분류 모델을 생성하였다. 생성된 다양한 분류 모델중 3가지 데이터셋 모두에서 최고의 분류 성능을 가지는 분류모델이 존재하는지에 대해서 탐색하였다. 실험결과 데이터셋 각각에서 최고의 분류성능을 내는 분류 모델은 모두 상이하였다. 하나의 분류 모델로 다수의 사용자 분류체계에 맞는 분류기를 생성하는 것에는 한계가 있다고 판단하였다.

이러한 한계를 극복하기 위해 본 연구에서는 앙상블 기법에 주목하였다. 앙상블 기법은 다수의 분류모델의 결과를 조합하여 하나의 결과를 제시하는 기법이다. 다수의 선행연구에서 앙상블 기법은 각 분류모델이 가지는 일반화의 오류를 상당부분 개선할 수 있다고 보고되었다. 대부분의 연구에서는 하나의 분류 알고리즘에 매개변수의 집합을 변경함으로써 다수의 분류기를 생성하고 이를 앙상블 기법을 적용하였다. 본 연구에서는 다수의 알고리즘과 다수의 매개변수의 집합을 이용해서 만들어진 분류모델을 이용하여 앙상블 기법을 적용하였다. 각 알고리즘의 특성이 특허 문서에 포함된 키워드 관점과 문장 관점의 분류모델이므로 이들의 결과를 앙상블 기법으로 조합하면 보다 높은 분류 성능을 얻을 수 있으리라 가정하였다. 이러한 가정을 바탕으로 5개의 알고리즘과 각 알고리즘 별로 다수의 매개변수 집합을 이용하여 분류 모델을 생성하였다. 생성된 분류 모델중 검증 데이터의 정확도와 오류치를 기반으로 각 3개의 대표 분류 모델을 선택하였다. 총 15개의 분류모델을 3가지의 앙상블 기법을 적용하였고 이중 가장 검증데이터 대비 가장 높은 정확도를 가지는 앙상블을 선택하였다. 즉 학습이 완료되면 15개의 분류모델과 하나의 앙상블 기법이 선택되는 것이다. 이를 이용하여 테스트 데이터로 분류 정확도를

측정하였을 때 3가지 데이터셋에 대해서 각각의 데이터셋의 최고 분류 정확도를 기록한 개별 모델과의 정확도를 비교하였다.

실험결과 1가지의 데이터셋에서는 개별 분류 모델중 가장 높은 분류성능을 나타내는 분류모델 보다 성능이 높게 나왔고, 나머지 2개의 데이터셋에 대해서는 거의 유사하거나 조금 못 미치는 분류 정확도가 나왔다. 즉 앙상블 기법을 통한 분류 성능은 각 데이터셋의 개별 모델 최고 분류정확도에 수렴한다는 결과를 얻었다. 이러한 결과를 통해 다수의 개별 알고리즘과 이를 통한 앙상블 기법은 다양한 사용자 분류체계를 위한 분류기에 적합한 방법이라고 결론 내렸다.

이러한 실험결과와 실제 산업현장에서 요구하는 특허문서 자동분류기의 요구사항을 바탕으로 본 연구에서는 제품화를 위한 특허문서 자동분류기의 아키텍처를 제안하였다. 특허문서 자동분류기의 아키텍처는 다양한 사용자 관점의 분류체계와 분류데이터에 맞게 자동으로 분류모델을 생성하여 분류되지 않은 데이터를 분류하는데 있어 데이터에 맞춘 개별분류모델과 유사한 또는 능가하는 분류성능을 갖추는 것을 목표로 하고 있고 이러한 일련의 과정을 자동화 하는 것을 지원하는 아키텍처이다.

본 연구에서 제안하는 분류방법과 아키텍처를 통해 사용자 관점의 분류체계와 분류가 활발히 만들어지고 이를 통해 특허의 다양하고 정교한 분석이 이루어짐을 통해, 특허 권리 침해 없는 다양한 R&D와 기술 발전이 지길 기대한다.

참 고 문 헌

1. 국내문헌

강필성, 이형주, 조성준. (2004). 데이터 불균형 문제에서의 SVM 앙상블 기법의 적용.『한국정보과학회 학술발표논문집』, 31(2), 706-708.

김성훈, 김승천. (2021). 특허문서 자동분류를 위한 딥러닝 개별 모델 분류기와 앙상블 분류기의 성능비교.『전자공학회논문지』, 58(9), 34-41.

박찬정, 성동수, 이건배. (2012). 기계학습을 이용한 특허 문서의 자동 IPC분류.『한국정보기술학회 논문지』, 10(4), 119-128.

오송첨단의료산업진흥재단. (2017). 의료기기 특허 출원 및 등록 동향 분석. 충북: 오송첨단의료산업진흥재단.

오송첨단의료산업진흥재단. (2017). 분야별 의료기기 특허 증가추이 분석. 충북: 오송첨단의료산업진흥재단.

이재안, 서형국, 한규열. (2011). 특허 문서 검색 결과를 이용한 KNN 기반의 특허 분류 시스템. 『한국정보과학회 학술발표논문집』, 38(2A), 256-259.

한국특허전략개발원. (2017).『정부 R&D 특허기술동향조사 가이드북』. 서울: 한국특허전략개발원.

특허청. (2020).『특허·실용신안 심사기준』. 대전: 특허청.

특허청. 선진특허분류(CPC) 코드. https://www.kipo.go.kr

특허청. 국제특허분류(IPC)코드. https://www.kipo.go.kr

2. 국외문헌

Bahdanau, D., Cho, K., Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473.*

Bbeiman, L. (1996). Bagging Predictors. *Machine Learning*, 24(2), pp. 123-140.

Benities, F., Malmasi, S., Zampieri, M. (2018). Classifying patent applications with ensemble methods. *arXiv prepreint arXiv:1811:04695.*

Chen, Y. L., Chang, Y. C. (2012). A three-phase method for patent classification. *Information Processing and Management*, 48(6), pp. 1017-1030.

Fall, C. J., Törcsvári, A., Benzineb, K., Karetka, G. (2003). Automated categorization in the international patent classification. *ACM Sigir Forum*, 37(1), pp. 10-25.

Gardner, M. W., Dorling, S. R. (1998). Artificial newral networks (The multilayer perceptron)-A Review of applications in the atomospheric sciences. *Atmospheric Environment*, 32(14-15), pp. 2627-2736.

Gomez, J. C. (2019). Analysis of the effect of data properties in automated patent classification. *Scientometrics,* 121(3), pp. 1239-1268.

Grawe, M. F., Martins, C. A., Bonfante, A. G. (2017). Automated Patent Classification Using Word Embedding. *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. pp. 408-411.

Hansen, L. K., Salamon, P. (1990). Neural Network Ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10),

pp. 993-1001.

Hochreiter, S., Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), pp. 1735-1780.

Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., Weinberger, K. Q. (2017). Snapshot ensembles: Train 1, get M for free. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 1-14.

Hubel, D. H. (1959). Single unit activity in striate cortex of unrestrained cats. *The Journal of physiology*, 147(2), pp. 226-238.

Ioffe, S., Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International conference on machine learning. PMLR,* pp. 448-456.

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746-1751.

Lecun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE,* 86(11), pp. 2278-2324.

Lee, J. S., Hsiang, J. (2020). Patent classification by fine-tuning BERT language model. *World Patent Information*, 61, 101965.

Li, S., Hu, J., Cui, Y., Hu, J. (2018). DeepPatent: patent classification with convolutional neural networks and word embedding. *Scientometrics*, 117(2), pp. 721-744.

Pennington, J., Socher, R., Manning, C. D. (2014). GloVe: Global vectors for word representation. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pp. 1532-1543.

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence*

Review, 33(1-2), pp. 1-39.

Smith, H. (2002). Automation of patent classification. *World Patent Information*, 24(4), pp. 269-271.

Tang, P., Jiang, M., Xia, B.N, Pitera, J. W., Welser, J., Chawla, N. V. (2020). Multi-label patent categorization with non-local attention-based graph convolutional network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(5), pp. 9024-9031.

Trappey, A. J., Hsu, F. C., Trappey, C. V., Lin, C. I. (2006). Development of a patent document classification and search platform using a back-propagation network. *Expert Systems with Applications,* 31(4), pp. 755-765.

Van der Malsburg, C. (1986). Frank Rosenblatt: Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. In A. Palm Güntherand Aertsen. *Brain Therory*, pp. 245-248, Springer Berlin Heidelberg.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. (2017). Attention Is All You Need. *Advances in neural information processing systems*, pp. 5998-6008.

Vieira, J. P. A., Moura, R. S. (2017). An analysis of convolutional neural networks for sentence classification. *2017 43rd Latin American Computer Conference, CLEI 2017*, 2017-January, pp. 1-5.

Wu, C. H., Ken, Y., Huang, T. (2010). Patent classification system using a new hybrid genetic algorithm support vector machine. *Applied Soft Computing*, 10(4), pp. 1164-1177.

Xie, J., Xu, B., Chuang, Z. (2013). Horizontal and Vertical Ensemble with Deep Representation for Classification. *arXiv preprint arXiv:1306.2759.*

Yang, Y., Pedersen, J. O. (1997). A Comparative Study on Feature

Selection in Text Categorization. *ICML 97: Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 412-420.

Young, T., Hazarika, D., Poria, S., Cambria, E. (2018). Recent trends in deep learning based natural language processing [Review Article]. *IEEE Computational Intelligence Magazine*, 13(3), pp. 55-75.

Yun, J., Geum, Y. (2020). Automated classification of patents: A topic modeling approach. *Computers and Industrial Engineering*, 147, 106636.

ABSTRACT

Automatic Classification of Patent Documents Based on Deep Learning According to User-defined Taxonomy

Kim, Sung-Hoon

Major in Smart Convergence Product

Dept. of Smart Convergence Consulting

The Graduate School

Hansung University

Patents with properties that protect the right to technological innovation are considered and important asset for most companies. Patents also play an important role in the diffusion of some technological innovations as they provide a sufficient source to represent technological advancement and diversification. In order to perform patent analysis such as technology development trend and competitor technology analysis using these patents, the classification of patent documents must be preceded. For patent analysis suitable for the purpose of a general company, a user-defined classification system defined by a domain expert is more suitable than an official classification system such as International Patent Classification (IPC) or Cooperative Patent Classification (CPC). The classification of patent documents is mostly done manually by experts, so

it takes a lot of time and cost. The purpose of this study is to find an optimal classification model using deep learning to automatically perform patent document classification suitable for various user classification systems to reduce such time and cost. Three classification datasets were defined, and 80% of each dataset was used as training data and 20% was used as test data. Two keyword-based classification algorithms and three sentence-based classification algorithms were selected for a total of five deep learning algorithms. A number of classification models were created for each algorithm and the classification accuracy of the test data of each dataset was measured. In addition, the classification accuracy of the test data of each dataset was measured using the ensemble method by combining the results of the classification model. There was no single classification model with the highest classification accuracy in all three datasets. The ensemble method showed the highest classification accuracy in one dataset among the three datasets, and the second-ranked classification accuracy was recorded in two datasets. As a model for patent classification suitable for the multiple user classification system, the purpose of this study, it was found that a classification model using an ensemble technique that combines multiple classification models is more suitable than a single classification using a specific algorithm. Based on the experimental results, in this study, a classification architecture for automatic classification of patent documents suitable for a user-defined classification system was proposed. It is expected that the proposed patent document automatic classification architecture will be used in the actual patent classification task, creating an environment where patent domain experts can focus more on patent analysis.

【Keyword】 automatic classification of patent documents, user-defined taxonomy, deep learning, ensemble