KIPS Trans. Softw. and Data Eng.
Vol.12, No.7 pp.295~302
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2023.12.7.295

A Study on Improving Performance of Software Requirements Classification Models by Handling Imbalanced Data

Jong-Woo Choi† · Young-Jun Lee†† · Chae-Gyun Lim†† · Ho-Jin Choi†††

ABSTRACT

Software requirements written in natural language may have different meanings from the stakeholders' viewpoint. When designing an architecture based on quality attributes, it is necessary to accurately classify quality attribute requirements because the efficient design is possible only when appropriate architectural tactics for each quality attribute are selected. As a result, although many natural language processing models have been studied for the classification of requirements, which is a high-cost task, few topics improve classification performance with the imbalanced quality attribute datasets. In this study, we first show that the classification model can automatically classify the Korean requirement dataset through experiments. Based on these results, we explain that data augmentation through EDA(Easy Data Augmentation) techniques and undersampling strategies can improve the imbalance of quality attribute datasets, and show that they are effective in classifying requirements. The results improved by 5.24%p on F1-score, indicating that handling imbalanced data helps classify Korean requirements of classification models. Furthermore, detailed experiments of EDA illustrate operations that help improve classification performance.

Keywords : Requirements Classification, Imbalanced Data, Data Augmentation, Undersampling, BERT

불균형 데이터 처리를 통한 소프트웨어 요구사항 분류 모델의 성능 개선에 관한 연구

최 종 우† · 이 영 준†† · 임 채 균†† · 최 호 진†††

요 약

자연어로 작성되는 소프트웨어 요구사항은 이해관계자가 바라보는 관점에 따라 의미가 달라질 수 있다. 품질 속성 기반으로 아키텍처 설계 시에 품질 속성별로 적합한 설계 전술(Tactic)을 선택해야 효율적인 설계가 가능해 품질 속성 요구사항의 정확한 분류가 필요하다. 이에 따라 고비용 작업인 요구사항 분류에 관한 자연어처리 모델이 많이 연구되고 있지만, 품질 속성 데이터셋(dataset)의 불균형을 처리해 분류 성능을 개선하는 주제는 많이 다루고 있지 않다. 본 연구에서는 먼저 실험을 통해 분류 모델이 한국어 요구사항 데이터셋을 자동으로 분류할 수 있음을 보인다. 이 결과를 바탕으로 EDA(Easy Data Augmentation) 기법을 통한 데이터 증강과 언더샘플링(undersampling) 전략으로 품질 속성 데이터셋의 불균형을 개선할 수 있음을 설명하고 요구사항의 카테고리 분류에 효과가 있음을 보인다. 실험 결과 F1 점수(F1-Score) 기준으로 최대 5.24%p 향상되어 불균형 데이터 처리 기법이 분류 모델의 한국어 요구사항 분류에 도움이 됨을 확인할 수 있다. 또한, EDA의 세부 실험을 통해 분류 성능 개선에 도움이 되는 데이터 증강 연산에 관해 설명한다.

키워드 : 요구사항 분류, 불균형 데이터, 데이터 증강, 언더샘플링, BERT

1. 서 론

소프트웨어 요구사항은 고객이나 이해관계자들이 앞으로 만들어질 소프트웨어에 대해 기대하는 사항들이다[1]. 주로 자연어로 작성되는 소프트웨어 요구사항은 자연어가 가지는 모호성으로 인해 이해관계자가 바라보는 관점에 따라 의미가 달라질 수 있다. 소프트웨어 요구사항의 구분 방법에 대해서는 논란이 있을 수 있지만 일반적으로 기능 요구사항과 비기능 요구사항으로 구분된다.

비기능 요구사항은 품질 속성 요구사항과 제약사항으로 분

※ 이 논문은 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No. 2013-2-00131, (엑소브레인-총괄/1세부) 휴먼 지식증강 서비스를 위한 지능진화형 WiseQA 플랫폼 기술 개발).
※ 이 논문은 2022년 소프트웨어공학연구회의 우수논문으로 "효율적인 아키텍처 패턴 적용을 위한 소프트웨어 요구사항 분류체계에 관한 연구"의 제목으로 발표된 논문을 확장한 것임.
† 준 회 원 : 한국과학기술원 소프트웨어대학원 석사
†† 비 회 원 : 한국과학기술원 전산학부 박사과정
††† 종신회원 : 한국과학기술원 전산학부 교수
Manuscript Received : December 22, 2022
First Revision : April 17, 2023
Accepted : April 21, 2023
* Corresponding Author : Ho-Jin Choi(hojinc@kaist.ac.kr)

류되고, 추가로 품질 속성 요구사항은 수십여 가지의 카테고리로 구성된다[2, 3]. 품질 속성 요구사항을 기반으로 소프트웨어 아키텍처 설계 시에 카테고리별로 적합한 품질 속성 설계 전술(Tactic)을 사용해야 효율적으로 설계할 수 있어 정확한 요구사항 분류가 매우 중요하게 여겨진다[4].

실무에서는 소프트웨어 아키텍트나 개발자들이 품질 속성 요구사항을 수작업으로 카테고리별 분류를 하는데 이는 많은 시간과 노력이 필요한 고비용의 업무이다. 이에 따라 자동으로 소프트웨어 요구사항을 분류하는 자연어처리 방법들이 연구되고 있다[5-7].

소프트웨어 요구사항은 개발 프로젝트마다 작성 스타일이 다르고 작성자마다 기술하는 요구사항 문장의 품질이 달라서 일반화된 데이터셋을 구성하는 것이 쉽지 않다. 요구사항 분류 연구에 일반적으로 사용되는 PROMISE NFR 요구사항 데이터셋이 있으나[8] 데이터 크기가 작고 카테고리 간에 데이터 불균형을 이루고 있다. 기존 연구에서는 소프트웨어 요구사항의 자동 분류를 위한 자연어처리 모델이 많이 연구되고 있지만 데이터 증강 기법을 이용한 품질 속성 데이터셋의 불균형 개선에 관한 연구가 적고, 더욱이 한국어 소프트웨어 요구사항 분류에 대해서는 다루고 있지 않다.

본 연구에서는 먼저 실험을 통해 분류 모델들이 한국어 요구사항 데이터셋을 자동으로 분류할 수 있음을 보인다. 이 결과를 바탕으로 EDA(Easy Data Augmentation) 기법[9]을 통한 데이터 증강과 최소 데이터 클래스를 기준으로 언더샘플링(undersampling) 전략을 적용해 요구사항 데이터셋의 불균형을 개선할 수 있음을 설명하고, 실험을 통하여 이 전략들이 카테고리별 분류 성능뿐만 아니라 전체 분류 성능 개선에 효과가 있음을 보인다. 또한 EDA 기법의 세부 연산 분석을 통해 분류 모델의 성능에 도움이 되는 방법에 대해 설명한다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구로 요구사항 분류 모델에 관한 연구와 불균형 데이터에 관한 연구에 대해 기술한다. 3장에서는 분류 모델의 요구사항 분류에 도움이 되기 위해 데이터 불균형을 개선하는 방법에 관해 설명한다. 4장에서는 3장에서 설명한 방법들을 실험하기 위한 실험 구성에 대해 작성하고, 실험한 결과와 효과성에 관해 기술한다. 5장에서는 결론과 후속 연구 방향을 제시한다. 이를 통해 소프트웨어 요구사항 분류 모델이 자동으로 한국어 소프트웨어 요구사항들을 분류하는 데 도움을 주고자 한다.

2. 관련 연구

소프트웨어 요구사항 분류 모델에 관한 연구들과 불균형 요구사항 데이터에 관한 연구들이 있다.

Dias et al.[10]은 k-최근접 이웃 알고리즘(k-Nearest Neighbor, k-NN)과 서포트 벡터 머신(Support Vector Machine, SVM)과 같은 분류기로 PROMISE_exp[11] 데이터셋을 분류하는 연구를 하였다. 기능 요구사항과 비기능 요구사항에 대한 이진 분류에서 F1 점수(F1-Score) 기준 91%와 비기능 요구사항을 세분화한 다중 분류에서 F1 점수 기준 78%를 달성 하였으나, 불균형 데이터를 처리하는 방법은 제시하지 않고 있다.

Navarro-Almanza et al.[12]은 심층학습(Deep Learning) 모델 중 합성곱 신경망(Convolutional Neural Network, CNN) 모델을 이용하여 PROMISE NFR 데이터셋을 분류하는 연구를 하였다. 비기능 요구사항을 세분화한 다중 분류에서 F1 점수 기준 77%를 달성 하였으나 불균형 데이터 처리에 대한 방법은 기술하지 않고 있다.

Hey et al.[13]은 전이 학습(Transfer Learning)이 가능한 BERT(Bidirectional encoder representation from transformer) 기반의 NoRBERT(Non-functional and functional Requirements classification using BERT) 모델을 제시하며 아직 학습해보지 않은 데이터(unseen data)에 대한 분류 성능을 높이는 것을 목표로 하고, PROMISE NFR 데이터셋을 분류하는 연구를 진행하였다. 불균형 데이터 처리를 위해 오버샘플링(oversampling)과 언더샘플링 기법을 활용해 기능 요구사항과 비기능 요구사항을 포함한 다중 분류에서 F1 점수 기준 90%를 기록하였다.

Lima et al.[11]에 따르면 요구사항 분류 연구의 타당성을 위해 양질의 요구사항 데이터셋이 필요하다. 일반적으로 사용되는 PROMISE NFR 데이터셋은 데이터 수가 적고 데이터가 불균형을 이루고 있어 확장이 필요하다고 지적하고, 이를 위해 새로운 프로젝트들의 요구사항을 통합하여 PROMISE NFR에 344개의 데이터를 추가한 PROMISE_exp 데이터셋을 제시하였다. 이를 서포트 벡터 머신 등에서 확인 실험을 한 결과, 분류 성능이 PROMISE NFR 데이터셋에 대한 분류 성능보다 F1 점수 기준 0.01%p가 향상되었다.

관련 연구를 살펴보았지만, 한국어 요구사항 분류에 관한 연구는 다루고 있지 않다. 본 논문에서는 알려진 불균형 데이터 개선 방법 중 상대적으로 구현 복잡도가 낮은 언더샘플링 기법과 문맥을 유지하면서 데이터 증강을 하는 EDA 기법으로 요구사항 데이터셋의 불균형을 개선하여 분류 모델의 한국어 요구사항 분류 성능을 개선하고자 한다.

3. 연구 방법

3.1 한국어 요구사항 데이터셋

1) PROMISE NFR 데이터셋 분석

본 논문에서는 관련 연구들에서 보이지 않은 한국어 요구사항 데이터셋을 만들고, 분류 모델이 영문 요구사항들과 유사하게 한국어 요구사항도 자동으로 분류할 수 있음을 실험적으로 접근하여 보인다.

영문 데이터셋은 관련 연구에서 일반적으로 사용하는 PROMISE NFR을 이용한다. PROMISE NFR은 15개 프로젝트의 소프트웨어 요구사항 명세서(Software Requirement Specification, SRS)에서 요구사항 문장들을 추출하여 전체 625개의 데이터로 구성한 데이터셋이다. 데이터는 255개의 기능 요구사항과 370개의 비기능 요구사항으로 구성되어 있고, 비기능 요구사항은 각각 11개의 품질 속성 클래스(class)로 분류되어 있다. 클래스별로 데이터 수를 비교해보면 Table 1과

Table 1. Data Distribution of the PROMISE NFR Dataset

No.	Class	Count
1	F (Functional)	255
2	A (Availability)	21
3	FT (Fault Tolerance)	10
4	L (Legal)	13
5	LF (Look and Feel)	38
6	MN (Maintainability)	17
7	O (Operational)	62
8	PE (Performance)	54
9	PO (Portability)	1
10	SC (Scalability)	21
11	SE (Security)	66
12	US (Usability)	67

같이 클래스별로 최대 255개, 최소 1개 데이터를 가지고 있어 불균형을 이루고 있다.

2) 한국어 데이터셋 생성

우리가 알고 있는 바로는 한국어 요구사항 데이터셋을 생성하는 연구가 시도된 적이 없기 때문에, 본 연구에서는 구글 번역[1]을 이용해 PROMISE NFR 데이터셋의 영문 요구사항 문장을 한국어로 번역하여 생성한다. 일반적으로 실무에서 소프트웨어 요구사항을 작성할 때 표준으로 ISO/IEC/IEEE 29148:2018[14]를 따르면서 문서 명세와 문장 구성 항목을 유사하게 작성하기 때문에 요구사항을 작성하는 데 한국어와 영어가 크게 다르지 않다. 또한, 비영어권에서 PROMISE NFR 데이터셋을 번역하여 연구[15]하고 있으므로 한국어로 번역된 요구사항 데이터셋을 활용하는 본 실험은 유의미한 시도가 될 것이다.

번역은 요구사항 문장만 번역하고, 각 문장의 클래스(Table 1 참조)는 원본 영문을 그대로 유지한다. PROMISE NFR 데이터셋의 요구사항 문장들이 기술적인 단어의 사용 빈도가 낮고 평이한 문장으로 작성되어 있어서 번역된 문장이 이해하는 데 문제없는 수준으로 생성되었다. 예를 들어, 요구사항에서 특정 시스템명으로 사용되는 'ChoiceParts'와 같은 고유명사나 'HTML'과 같은 약어는 원본 영문을 그대로 번역해주고, 'display'와 같은 용어는 다른 한국어로 번역하지 않고 외래어 표기법처럼 '디스플레이'로 번역해주었다. 이에 따라 번역된 문장들이 어색하지 않으며 의미적으로 중요한 정보를 유지하고 있는 것으로 판단된다.

번역된 결과 예시는 Table 2와 같다. Table 2에 번역된 한국어 문장의 구글 번역 TTS(Text-to-Speech) 발음을 발음기호 형식으로 같이 표기하였다. 발음기호는 모델학습에는 이용되지 않는다.

학습 데이터와 실험 데이터가 분류되어 있지 않아 데이터셋을 8:2 비율로 학습 데이터와 실험 데이터로 분리하여 사용한다.

3.2 소프트웨어 요구사항 분류 모델

BERT(Bidirectional Encoder Representations from Transformers)[16]는 구글에서 공개한 대용량 데이터를 사전 학습한(pre-trained) 언어 모델로 여러 자연어 처리 태스크(task)에서 좋은 성능을 보였다. 문장 분류 분야에서도 적은 양의 데이터셋으로 미세조정(fine-tuning) 하여 활용이 가능하다. 본 연구에 BERT 언어 모델의 사용이 적절한지 확인하기 위해 BERT Base 모델과 BERT Large 모델로 PROMISE NFR 데이터셋을 분류하는 사전 실험을 하였다. Table 3과 같이 Navarro-Almanza et al.[12]이 합성곱 신경망 모델을 통해 다중 분류에서 달성한 F1 점수인 77%보다 높은 84.46%를 보여 BERT가 본 연구에서 사용하기에 적합함을 알 수 있다.

이를 바탕으로 한국어 문장을 분류할 수 있는 다음 모델들을 실험 모델로 선정한다.

Table 2. Requirement Sentence Examples of Translation to Korean by Google Translate

Type	Requirement sentence	Class
Original	The system shall refresh the display every 60 seconds.	PE
Translated	시스템은 60초마다 디스플레이를 새로 고칩니다. [siseutem-eun 60chomada diseupeulleileul saelo gochibnida.]	PE

Table 3. Performance Results of Multi-class Classification Using CNN and BERT Models

Metric	CNN	BERT Base	BERT Large
Precision (%)	81.00	83.91	85.40
Recall (%)	78.50	83.97	84.62
F1-score (%)	77.00	83.92	84.46

1) Multilingual BERT

Multilingual BERT[16]는 구글이 발표한 다국어 버전 모델로 104개 언어의 위키피디아를 사용하여 사전 학습하였고, 다양한 언어에서 일반화 성능이 높은 것으로 알려져 있다.

2) bert-kor-base

bert-kor-base는 한국어 코퍼스(corpus)를 사전 학습한 BERT 계열의 모델로 상대적으로 잘 정제된 뉴스와 국내 주요 커머스 리뷰, 모두의 말뭉치, 위키피디아와 나무위키 데이터를 학습하였다[17]. 현재는 base 모델만 공개되어 있다.

3) kcbert

kcbert도 한국어 데이터를 사전 학습한 BERT 계열의 언어 모델로 bert-kor-base와는 다르게 특정 기간 댓글이 많은 뉴스 기사의 댓글들을 수집한 데이터만 학습하였다[18]. kcbert-base 모델과 kcbert-large 모델이 공개되어 있어 파라미터 수에 따른 성능 비교가 가능하다.

3.3 데이터 불균형 개선 방법

1) 데이터 증강

데이터 증강은 학습 데이터가 부족하거나 데이터가 불균형을 이루고 있을 때 데이터 변형을 통해 데이터를 강제로 증가

1) https://translate.google.co.kr/

Table 4. Korean Sentences Generated by Each of EDA Operations

Operation	Sentence
None	결함 보고서는 연중무휴로 기술 부서에 제공됩니다. [gyeolham bogoseoneun yeonjungmuhyulo gisul buseoe jegongdoebnida.]
SR	결함 보고서는 연중무휴로 재주 부서에 제공됩니다. [gyeolham bogoseoneun yeonjungmuhyulo jaeju buseoe jegongdoebnida.]
RI	재주 결함 보고서는 연중무휴로 기술 부서에 제공됩니다. [jaeju gyeolham bogoseoneun yeonjungmuhyulo gisul buseoe jegongdoebnida.]
RS	기술 보고서는 연중무휴로 결함 부서에 제공됩니다. [gisul bogoseoneun yeonjungmuhyulo gyeolham buseoe jegongdoebnida.]
RD	보고서는 연중무휴로 기술 부서에 제공됩니다. [bogoseoneun yeonjungmuhyulo gisul buseoe jegongdoebnida.]

시키는 기법이다. 컴퓨터 비전 분야에서는 활발하게 사용되었지만, 자연어는 데이터 변형 시 문장의 의미가 달라질 수 있어 데이터 증강이 쉽지 않았다. 자연어 데이터를 증강하는 여러 기법이 있으나 그중에 EDA는 간단한 몇 가지의 연산을 통해 문장의 문맥을 유지하면서 데이터를 증강하는 기법이다.

EDA 기법은 4가지 연산을 통해 데이터를 증강한다. 임의의 동의어 중 하나로 교체하는 유의어 교체(Synonym Replacement, SR), 임의로 선택한 단어의 동의어를 선택하여 임의의 위치에 삽입하는 랜덤 삽입(Random Insertion, RI), 무작위로 두 단어를 선택하고 서로 위치를 교체하는 랜덤 교체(Random Swap, RS) 그리고 특정 확률로 문장 내의 각 단어를 임의로 삭제하는 랜덤 삭제(Random Deletion, RD) 연산으로 이루어져 있다. 동의어나 유의어 교체를 위해 의미 어휘목록인 워드넷 중에 KAIST(Korea Advanced Institute of Science and Technology, 한국과학기술원)에서 공개한 한국어 워드넷인 KWN(Korean WordNet)[19]을 이용한다. 데이터 증강 시에 문장의 클래스가 바뀌면 문제가 될 수 있는데 본 연구에서는 데이터 확장이 목적이므로 생성된 문장의 클래스를 새로 판단하지 않고, 원본 문장의 클래스를 그대로 이용한다.

각 연산을 통해 새로 생성된 한국어 문장의 예시는 Table 4와 같고, 본 연구에서는 EDA 연산을 모두 이용한다. Table 4에 새로 생성된 한국어 문장마다 구글 번역의 TTS 발음을 발음기호 형식으로 같이 표기하였다. 발음기호는 모델학습에는 이용되지 않는다.

2) 언더샘플링

데이터 불균형을 해소하기 위한 다른 방법으로 언더샘플링 기법을 이용한다. 데이터 수가 1개인 PO(Portability) 클래스의 데이터는 제외하고 그다음으로 적은 10개 데이터로 구성된 FT(Fault Tolerance) 클래스를 기준으로 다른 클래스의 데이터를 임의로 언더샘플링하여 모든 클래스의 데이터를 각각 10개로 균형을 맞춘다.

4. 실험 구성 및 실험 결과

4.1 연구 질문(Research Questions)

1) [RQ1] 한국어 소프트웨어 요구사항도 분류가 되는가?

구글 번역을 통해서 생성한 한국어 요구사항 데이터셋을 이용해 분류 모델이 한국어 소프트웨어 요구사항 분류에도 적합한지 확인하기 위해 분류 모델로 분류 성능을 측정해보고 기존 영문 요구사항 데이터셋의 분류 성능과 비교한다.

2) [RQ2] 불균형 데이터 개선이 요구사항 분류 모델의 성능을 향상 시키는가?

EDA 기법과 언더샘플링 기법을 적용하여 한국어 소프트웨어 요구사항 데이터셋의 불균형을 개선하면서 분류 모델의 분류 성능을 측정하고 개선 전 분류 성능과 비교한다.

4.2 실험 환경 구성

Google Colab[2]에서 실험 환경을 구성하고 Colab에서 제공하는 Nvidia Tesla T4 GPU와 8GB GPU Memory를 이용하여 모델을 학습한다.

분류 모델의 하이퍼파라미터는 모델에서 기본으로 설정하는 값인 2e-5 learning rate, 32 batch size, 32 epoch을 유지하고, EDA 연산 별 α 파라미터는 모두 0.1로 설정한다. 이때 α 파라미터는 변경된 문장 내 단어의 비율로 0부터 1 사이의 값으로 설정할 수 있다. 문장의 길이와 α 파라미터를 곱하여 변화되는 단어의 수를 정하고, 변화되는 단어는 문장에서 랜덤하게 정해진다.

4.3 분류 모델별 분류 실험

본 논문에서는 이진 분류 실험과 다중 분류 실험을 한다. 이진 분류 실험에서는 먼저 기능 요구사항이 아닌 나머지 비기능 요구사항들의 클래스를 NFR(Non Functional Requirements)로 재설정한다. 그리고 기능 요구사항을 의미하는 F(Functional) 클래스와 NFR 클래스를 분류한다. 성능 평가 지표는 분류 태스크에서 주로 사용하는 혼동 행렬(Confusion Matrix)을 이용한다. 데이터 불균형으로 인해 정확도(Accuracy)로 성능 비교가 어려워 정밀도(Precision), 재현율(Recall), F1 점수를 측정하고 F1 점수를 이용해 분류 모델 간 성능 비교를 한다. Table 5에서처럼 이진 분류 실험에서는 bert- kor-base, kbert-large, Multilingual BERT가 비슷한 성능을 보이고 있으나 kcbert-large가 F1 점수 기준 92.73%로 가장 좋은 분류 성능을 보였고, kcbert-base가 F1 점수 기준 89.55%로 가장 낮은 성능을 보였다.

다중 분류 실험은 기존에 데이터셋에 기록된 클래스를 분류하는 실험이다. 다만, 데이터를 1개 가지고 있는 PO 클래스는 부적절한 데이터라고 판단되어 제거하고 실험한다. Table 6과 같이 다중 분류는 bert-kor-base가 F1 점수 기준 80.15%로 가장 좋은 성능을 보였다. 또한, PROMISE NFR을 실험한 Table 3의 BERT Base 보다 약 3.8%p가 낮게 나왔는데 이는 데이터셋의 번역 오류로 인한 성능 차이로 생각된다. 약간의

2) https://colab.research.google.com/

Table 5. Binary Classification Results for Each Classification Model

Classification Model	Precision (%)	Recall (%)	F1-score (%)
bert-kor-base	92.27	92.00	91.91
kcbert-base	89.60	89.60	89.55
kcbert-large	**92.98**	**92.80**	**92.73**
Multilingual BERT	92.08	91.11	91.60

Table 6. Multi-class Classification Results for Each Classification Model

Classification Model	Precision (%)	Recall (%)	F1-score (%)
bert-kor-base	**81.45**	**80.41**	**80.15**
kcbert-base	72.46	73.60	72.16
kcbert-large	74.30	75.00	73.76
Multilingual BERT	74.19	74.36	73.04

성능 차이는 있지만 RQ1과 관련하여 위 실험 결과를 바탕으로 분류 모델이 한국어 요구사항 분류에도 적합함을 알 수 있다.

kcbert-large 모델이 이진 분류에서는 가장 좋은 성능을 보였지만 다중 분류에서 F1 점수 기준 73.76%로 bert-kor-base의 F1 점수 기준 80.15%보다 낮은 성능을 보였다. 이진 분류는 두 클래스의 데이터 수가 충분해 균형을 이루고 있다고 볼 수 있으나, 다중 분류에서는 kcbert 모델이 불균형 데이터로 인해 분류 성능이 상대적으로 안 좋은 것으로 생각된다. kcbert-base 모델도 kcbert-large와 마찬가지로 불균형 데이터에 대한 문제가 있고, 사전 학습 데이터 수가 상대적으로 적어서 분류 성능이 안 좋은 것으로 판단된다. Multilingual BERT는 이진 분류에서는 다른 모델들과 유사한 성능을 보였으나, 다중 분류에서 낮은 성능을 보였는데, Eisenschlos et al.[20] 에 의하면 Multilingual BERT가 다중 분류 시에 데이터 수가 적으면 분류 성능이 떨어짐을 알 수 있다.

4.4 불균형 데이터 처리 후 분류 모델별 분류 실험

1) 데이터 증강 기법 적용 결과

EDA 기법을 적용하여 실험한 결과를 보면 Table 7에서 보는 것처럼 이진 분류 실험에서는 bert-kor-base가 0.67%p, Multilingual BERT가 0.33%p 상승을 보이고, kcbert-base가 0.79 %p, kcbert-large가 1.59%p 하락하는 결과를 보였다. 이미 비기능 요구사항 데이터를 하나로 통합하면서 데이터 불균형이 일부 개선된 상태라 EDA 기법은 분류 모델의 분류 성능 개선에 큰 도움이 되지 않는 것으로 보인다. Wei et al.[9]에 의하면 대규모 코퍼스를 사전 학습한 언어 모델에서는 EDA 기법이 필요 없을 수도 있다고 하였는데 본 실험 항목과 관련이 있어 보인다.

EDA 기법을 적용하여 실험한 결과를 보면 Table 8에서 보는 것처럼 다중 분류 실험에서는 모든 모델의 분류 성능이 향상되었다. bert-kor-base가 3.6%p, kcbert-base가 2.46%p 상승하였고, Multilingual BERT가 1.44%p 상승을 보이고, kcbert-large가 0.23%p로 약간 상승한 결과를 보였다. RQ2와 관련하여 불균형 데이터가 있을 때 EDA 기법을 통한 불균형 데이터개선이 분류 모델의 분류 성능 향상에 도움이 됨을 확인할 수 있다.

Table 7. Binary classification Results for Each Classification Model with Dataset Augmented by EDA

Classification Model	Precision (%)	Recall (%)	F1-score (%)	diff.
bert-kor-base	92.27	92.00	91.91	0
+EDA	**92.59**	**92.60**	**92.58**	+0.67
kcbert-base	89.60	89.60	89.55	0
+EDA	88.77	88.80	88.76	-0.79
kcbert-large	92.98	92.80	92.73	0
+EDA	91.27	91.20	91.14	-1.59
Multilingual BERT	92.08	91.11	91.60	0
+EDA	91.94	91.90	91.93	+0.33

Table 8. Multi-class Classification Results for Each Classification Model with the Balanced and Augmented Dataset

Classification Model	Precision (%)	Recall (%)	F1-score (%)	diff.
bert-kor-base	81.45	80.41	80.15	0
+EDA	**84.36**	**83.97**	**83.75**	+3.6
+UNDER	81.82	77.27	76.34	-3.81
+UNDER+EDA	82.06	75.00	73.23	-6.92
kcbert-base	72.46	73.60	72.16	0
+EDA	74.50	75.60	74.50	+2.46
+UNDER	80.30	68.18	69.09	-3.07
+UNDER+EDA	80.30	77.27	76.32	+4.16
kcbert-large	74.30	75.00	73.76	0
+EDA	74.95	75.00	73.99	+0.23
+UNDER	83.33	77.27	76.97	+3.21
+UNDER+EDA	82.92	79.55	79.00	+5.24
Multilingual BERT	74.19	74.36	73.04	0
+EDA	74.91	75.32	74.48	+1.44
+UNDER	62.27	68.18	62.77	-10.27
+UNDER+EDA	82.06	75.00	73.23	+0.19

2) 언더샘플링 기법 적용 결과

Table 8에서 언더샘플링 기법을 이용하여 실험한 결과를 보면 kcbert-large 모델을 제외하고 모두 성능이 하락하였다. 이는 데이터의 균형은 맞추었으나 클래스별 데이터 수가 너무 작아 학습이 충분히 안 된 것으로 생각된다. Fig. 1을 보면 검증 오차(validation loss)가 일정 시점부터 줄지 않는 것으로 보아 과소적합(underfitting)도 의심된다. kcbert- large 모델은 3.21%p 상승을 보였는데 불균형 데이터 실험 시에 문제가 되는 모델이라 언더샘플링만으로도 분류 모델의 분류 성능 향상에 도움이 된 것으로 판단된다.

3) 언더샘플링과 데이터 증강 조합 적용 결과

최소 데이터 클래스의 크기가 작아 언더샘플링만으로는 분류 성능 향상에 크게 도움이 되지 않아 EDA 기법과 같이 사용하여 실험한 결과이다. Table 8에서 결과를 살펴보면 kcbert-large가 5.24%p , kcbert-base가 4.16%p 성능 향상을 보였고, Multilingual BERT가 0.19%p로 약간의 성능 향상을 보였지만 bert-kor-base는 6.92%p 하락을 보였다. 불균형 데이터에 문제가 더 큰 kcbert-base와 kcbert-large에 언더샘플링과 데이터 증강을 조합한 기법이 큰 도움이 될 수 있음을 확인할 수 있다.

4.5 각 클래스에 대한 분류 성능 비교

Table 9에서 데이터 불균형 개선 기법 적용에 따른 클래스별 성능 수치를 볼 수 있다. PE 클래스나 A 클래스를 제외하면 대부분 클래스에서는 데이터 증강이나 언더샘플링 기법을 사용할 때 분류 성능이 향상됨을 볼 수 있다. 모든 모델에서 F 클래스는 F1 점수 기준 100%에 가까울 정도로 높아졌다. F 클래스는 기능 요구사항으로 다른 비기능 요구사항과는 확연하게 분류가 가능한 문장으로 구성되어 있기 때문으로 판단된다. 또한, kcbert-base에서 분류를 못 하던 FT 클래스는 언더샘플링 기법을 사용하면 F1 점수 기준 40~50%로 향상됨을 볼 수 있고, kcbert-large 모델에서 분류를 못 하던 MN 클래스는 EDA 기법과 언더샘플링 전략 모두 F1 점수를 66.7%~85.7%로 향상됨을 볼 수 있다. 반대로 Multilingual BERT에서 US 클래스나 MN 클래스의 경우에는 언더샘플링만 적용했을 때 분류를 못하는 상황이 발생하기도 했는데, 언더샘플링 과정에서 양질의 데이터가 선택되지 않아서 발생했을 것으로 판단된다.

각 모델별 모든 클래스의 F1-점수 평균값을 보면 bert-kor-base는 언더샘플링 기법만 사용할 때와 언더샘플링과 EDA 기법을 모두 사용할 때 유사한 결과를 보였고, kcbert-base, kcbert-large, Multilingual BERT는 모두 언더샘플링과 EDA 기법을 모두 사용할 때 가장 높은 평균값을 보였다. 모든 모델에서 동일하게 성능이 향상하거나 모든 카테고리에서 분류 성능이 향상되지는 않았지만, 해당 수치들을 통해 EDA 기법이나 언더샘플링 기법으로 클래스별 분류 성능이 향상됨을 확인할 수 있다.

4.6 EDA 개별 연산 실험

EDA 기법을 통한 데이터 불균형 해소로 분류 모델들의 분류 성능이 향상되어 어떤 연산이 분류 모델의 분류 성능에 영향을 더 주는지 실험적으로 추이를 분석해 보고자 한다. 본 실험에서 EDA 연산 별 파라미터는 변경된 문장 내 단어의 비율로 0부터 1 사이의 값으로 설정 가능하다. 문장의 길이와 파라미터를 곱하여 변화되는 단어의 수를 정하고, 변화되는 단어는 문장에서 랜덤하게 정해진다. 0.5 비율로 설정하면 50%의 단어에 변화를 주면서 증강하게 된다.

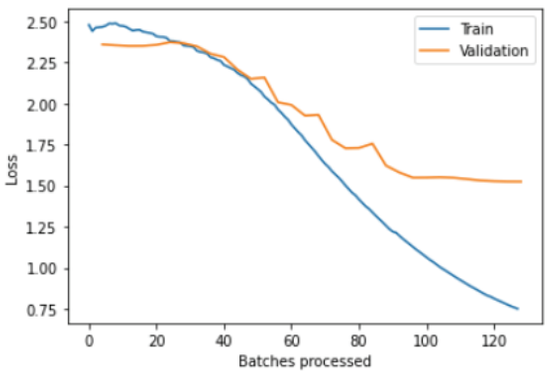

Fig. 1. Loss Graph While Bert-kor-base is Learning Dataset

Fig. 2를 보면 유의어 교체(SR)는 연산 파라미터를 0에서 0.2 비율로 올릴 때까지 성능이 좋아지다가 0.3 비율 설정부터는 성능 향상이 줄어들고 있다. 이는 워드넷에 소프트웨어공학과 관련된 단어의 수가 적어 불필요한 변경이 높아지면 분류 성능 향상에 도움이 되지 않는 것으로 판단된다.

랜덤 삽입(RI)은 연산 파라미터 변경에 대해 유사한 성능을 계속 보였는데 워드넷에 소프트웨어공학에 관련된 단어의 수가 적어서 문장의 다양성에 주는 영향도가 작아 분류 성능 향상에 큰 도움이 되지 않는 것으로 생각된다. 랜덤 교체(RS)는 연산 파라미터를 0에서 0.3 비율로 올릴 때까지 성능이 좋아지다가 0.4 비율 설정부터 성능 향상이 낮아지고 있지만 다른 연산들 대비 분류 모델의 분류 성능 향상 수준이 제일 높은 것으로 보인다. 그러나 일정 비율 이상의 교체는 문장 구조를 깨는 수준이 되어 분류 성능에 도움이 되지 않는 것으로 보인다. 랜덤 삭제(RD)는 연산 파라미터를 0에서 0.2 비율로 올릴 때까지는 성능이 좋아지다가 0.3 비율 설정부터 성능이 안 좋아지는 데 삭제 비율이 높아지면 문장 구조를 깨는 수준이 높아지고 분류 모델이 문장을 해석하기 어려워지기 때문으로 분석된다.

Table 9. Classification Results for Each Class in the Dataset (F1-score, %)

Classification Model	F	US	SE	O	PE	LF	A	SC	MN	L	FT	Avg.
bert-kor-base	84.4	76.9	72.7	69.6	**100.0**	85.7	**100.0**	50.0	28.6	75.0	**66.7**	73.6
+EDA	88.0	**88.0**	69.0	**78.3**	95.7	**87.5**	**100.0**	**80.0**	66.7	75.0	0.0	75.3
+UNDER	**100.0**	66.7	80.0	50.0	**100.0**	50.0	66.7	**80.0**	**100.0**	**80.0**	**66.7**	**76.4**
+UNDER+EDA	88.9	85.7	**88.9**	75.0	88.9	57.1	66.7	66.7	80.0	75.0	**66.7**	76.3
kcbert-base	86.0	74.1	56.0	60.0	80.0	71.4	60.0	33.3	33.3	**75.0**	0.0	57.2
+EDA	86.6	76.9	60.0	60.0	**85.1**	73.3	**66.7**	53.3	33.3	71.4	0.0	60.6
+UNDER	**100.0**	66.7	**100.0**	**66.7**	66.7	80.0	**66.7**	66.7	66.7	40.0	40.0	69.1
+UNDER+EDA	**100.0**	**85.7**	85.7	**66.7**	75.0	**88.9**	**66.7**	**72.7**	**85.7**	50.0	50.0	**75.2**
kcbert-large	85.6	73.7	60.6	56.0	**88.9**	53.3	80.0	75.0	0.0	57.1	**80.0**	64.6
+EDA	84.9	74.3	64.7	66.7	82.8	53.3	72.7	66.7	80.0	**85.7**	66.7	72.6
+UNDER	**100.0**	**100.0**	**100.0**	80.0	66.7	66.7	**100.0**	**100.0**	66.7	66.7	0.0	77.0
+UNDER+EDA	**100.0**	88.9	85.7	**88.9**	72.7	**85.7**	75.0	88.9	**85.7**	**85.7**	28.6	**80.5**
Multilingual BERT	83.7	72.2	73.3	56.0	92.3	28.6	61.5	**75.0**	28.6	**85.7**	40.0	63.4
+EDA	40.0	**74.3**	67.8	66.7	84.2	45.2	**66.7**	66.7	53.3	72.7	84.4	65.6
+UNDER	**100.0**	0.0	66.7	66.7	**100.0**	66.7	**66.7**	57.1	0.0	66.7	**100.0**	62.8
+UNDER+EDA	**100.0**	40.0	**80.0**	**72.7**	88.9	**75.0**	**66.7**	72.7	**57.1**	66.7	85.7	**73.2**

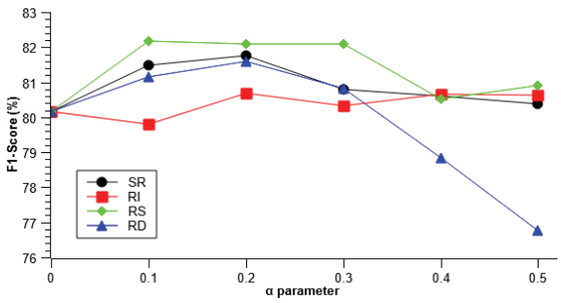

Fig. 2. Classification Results Based on Changing the Parameter α for Each Operation of the EDA

4.7 결과 고찰

RQ1과 관련하여 분류 모델이 기본적으로 번역된 한국어 요구사항 데이터셋도 자동으로 분류하는 데 적합함을 실험을 통해 확인하였다. RQ2와 관련하여 언더샘플링 전략만 사용하면 불균형 데이터 분류 시에 성능이 더 좋지 않았던 모델에서는 성능 향상 효과가 있었고, 나머지 모델에서는 과소적합으로 판단되는 양상을 보이며 성능이 하락하는 경향을 보였다. 그러나 데이터 증강 기법을 이용하거나, 언더샘플링과 데이터 증강 기법을 같이 활용하면 분류 모델이 한국어 요구사항에 대해 클래스별 분류 성능뿐만 아니라 전체 분류 성능 향상에도 도움이 될 수 있을 것으로 분석된다.

또한 크기가 더 큰 코퍼스를 사전 학습한 분류 모델이 파라미터 수가 더 크지만, 더 작은 코퍼스를 사전 학습한 분류 모델보다 분류 성능이 좋았다. 이를 통해 복잡한 파라미터 수보다 데이터양이 분류 성능에 영향을 더 많이 주는 것을 알 수 있다. EDA 개별 연산 실험 중 동의어 교체(SR) 연산과 랜덤 교체(RS)가 효과가 좋았고, 랜덤 삭제(RD)는 비율이 커지면 문장 구조를 깨뜨려 분류 모델이 문장을 이해하는 데 도움이 안 되기 때문에 연산 파라미터를 0.2 비율로 설정하면 효과가 있을 것으로 보인다.

데이터 불균형이 개선된 데이터셋을 학습한 분류 모델이 임의의 문장도 분류를 잘하는지 보기 위해 "시스템은 참조 가능한 색 구성 맵을 가져야 한다."라는 문장을 입력해보았다. 이 문장으로 평가해 보면 분류 모델은 F 클래스가 아닌 LF 클래스로 분류해 주었다. 요구사항 분류에 대해 논쟁이 될 수 있지만 보는 관점에 따라 기능 요구사항인 F 클래스로 분류할 수 있는 문장을 분류 모델은 품질 속성 중 LF 클래스로 분류하였다. 이를 보면 사람이 해석하면서 모호해 하는 문장들은 분류 모델도 만족스럽게 분류하지 못할 것으로 보인다.

5. 결론 및 향후 연구

고비용의 요구사항 분류 업무에 대해 자동 분류의 필요성으로 많은 연구가 이루어지고 있으나, 데이터 증강을 통한 분류 성능 향상 연구가 적고, 특히 한국어 요구사항 분류에 관해서는 연구된 바가 없다. 본 연구에서는 BERT 계열의 모델을 통해 분류 모델들이 한국어 요구사항 데이터를 분류할 수 있고, 불균형 데이터 처리가 분류 모델의 성능을 향상 시키는데 효과가 있음을 보였다. 특히, 언더샘플링 전략은 불균형 데이터 분류 시에 성능이 더 좋지 않았던 모델에서만 성능 개선 효과가 있었고, 데이터 증강을 사용하거나 언더샘플링과 데이터 증강을 같이 사용할 때 분류 성능이 개선되는 데 효과가 있음을 확인하였다.

본 연구에서는 기존에 많이 연구된 불균형 데이터 개선 기법들을 이용하여 소프트웨어 요구사항 분류 모델의 성능을 향상시키고 있어 후속 연구가 필요하다. 번역 오류로 인한 성능 저하가 있을 수 있어 후속 연구에서 한국어로 작성된 소프트웨어 요구사항 명세서에서 추출한 데이터셋을 대상으로 실험을 수행함으로써 각 모델이 다른 경향성을 보이는지에 대한 검증이 요구된다. 추가로 Stack overflow와 같은 플랫폼에서 소프트웨어 용어 수집을 통해 워드넷에 소프트웨어 공학 관련 동의어와 유의어를 보완하면 EDA 기법 중 동의어 관련 연산의 효과를 더 높여볼 수 있을 것으로 판단된다. 추가로 오버샘플링 기법을 이용하면 다각도로 분류 성능을 비교해 볼 수 있고, 다양한 모델 비교 측면에서 파라미터 수가 더 큰 언어 모델에서도 데이터 불균형을 해소하는 기법이 분류 성능 개선에 도움이 되는지 평가해 볼 수 있을 것이다.

References

[1] K. Wiegers and J. Beatty, "Software Requirements," Sydney: Pearson Education, 2013.

[2] J. Eckhardt, A. Vogelsang, and D. M. Fernández, "Are non-functional requirements really non-functional?: An investigation of non-functional requirements in practice," *Proceedings of the 38th International Conference on Software Engineering*, 2016.

[3] M. Glinz, "On non-functional requirements," in *15th IEEE International Requirements Engineering Conference (RE)*, pp.21-26, 2007.

[4] L. Bass, P. Clements, and R. Kazman, "Software architecture in practice," Upper Saddle River (N.J.): Addison-Wesley, 2013.

[5] Z. S. H. Abad, O. Karras, P. Ghazi, M. Glinz, G. Ruhe, and K. Schneider, "What works better? A study of classifying requirements," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*, pp.496-501, 2017.

[6] M. Binkhonain and L. Zhao, "A machine learning approach for hierarchical classification of software requirements," *arXiv preprint arXiv:2302.12599*, 2023.

[7] X. Luo, Y. Xue, Z. Xing, and J. Sun, "PRCBERT: Prompt learning for requirement classification using BERT-based pretrained language models," In *37th IEEE/ACM International Conference on Automated Software Engineering*, pp.1-13, 2022.

[8] Z. Kurtanovic and W. Maalej, "Automatically classifying functional and non-functional requirements using supervised machine learning," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*, pp.490-495, 2017.

[9] J. W. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," *arXiv:1901.11196*, 2019.
[10] E. Dias Canedo and B. Cordeiro Mendes, "Software requirements classification using machine learning algorithms," *Entropy*, Vol.22, No.9, pp.1057, 2020.
[11] M. Lima, V. Valle, E. Costa, F. Lira, and B. Gadelha, "Software engineering repositories: Expanding the PROMISE database," in *Proceedings of the 33rd Brazilian Symposium on Software Engineering*, pp.427-436, 2019.
[12] R. Navarro-Almanza, R. Juarez-Ramirez, and G. Licea, "Towards supporting software engineering using deep learning: A case of software requirements classification," in *2017 5th International Conference in Software Engineering Research and Innovation (CONISOFT)*, pp.116-120, 2017.
[13] T. Hey, J. Keim, A. Koziolek, and W. Tichy, "NoRBERT: Transfer learning for requirements classification," In *2020 IEEE 28th International Requirements Engineering Conference (RE)*, pp.169-179, 2020.
[14] ISO/IEC/IEEE 29148:2018. Systems and software engineering. Life cycle processes. Requirements engineering (2018) [Internet], https://www.iso.org/standard/72089.html
[15] M. I. Limaylla-Lunarejo, N. Condori-Fernandez, and M. R. Luaces, "Towards an automatic requirements classification in a new Spanish dataset," *2022 IEEE 30th International Requirements Engineering Conference (RE)*, IEEE, 2022.
[16] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pretraining of deep bidirectional transformers for language understanding," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol.1 (Long and Short Papers). Linguistics, pp.4171-4186, 2019.
[17] Github - kiyoungkim1/LMKor, Github, 2022. [Internet], https://github.com/kiyoungkim1/LMkor
[18] Github - Beomi/KcBERT, Github, 2022. [Internet], https://github.com/Beomi/KcBERT
[19] Wordnet.kaist.ac.kr, 2022. [Internet], http://wordnet.kaist.ac.kr/
[20] J. Eisenschlos, S. Ruder, P. Czapla, M. Kadras, S. Gugger, and J. Howard, "MultiFiT: Efficient multi-lingual language model fine-tuning," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp.5706-5711, 2019.

최 종 우
https://orcid.org/0009-0001-2718-7608
e-mail : jongwoos.choi@gmail.com
2010년 건국대학교 컴퓨터공학부(학사)
2023년 한국과학기술원 소프트웨어대학원 (석사)
2010년 ~ 2013년 삼성탈레스 해양/시스템연구소 연구원
2013년 ~ 2015년 LG전자 SW플랫폼연구소 주임연구원
2015년 ~ 현 재 LG전자 HE연구소 책임연구원
관심분야 : Software Engineering, Software Architecture, Software Reliability, Artificial Intelligence

이 영 준
https://orcid.org/0000-0002-6904-8278
e-mail : yj2961@kaist.ac.kr
2017년 성균관대학교 소프트웨어학과 (학사)
2019년 한국과학기술원 전산학부(석사)
2019년 ~ 현 재 한국과학기술원 전산학부 박사과정
관심분야 : Social Dialogue System, Multi-Modal learning, Natural Language Processing

임 채 균
https://orcid.org/0000-0002-4534-4005
e-mail : rayote@kaist.ac.kr
2011년 을지대학교 의료전산학과(학사)
2015년 경희대학교 컴퓨터공학과(석사)
2015년~현 재 한국과학기술원 전산학부 박사과정
관심분야 : Temporal Information Extraction, Multi-task Learning, Big Data Analysis, Bioinformatics

최 호 진
https://orcid.org/0000-0002-3398-9543
e-mail : hojinc@kaist.ac.kr
1982년 서울대학교 컴퓨터공학(학사)
1982년~1989년 ㈜데이콤(현 LG U+) 정보통신연구소 선임연구원
1985년 Newcastle University 컴퓨터과학(석사)
1995년 Imperial College London 인공지능학(박사)
1995년~1996년 Imperial College London IC-PARC 박사후연구원
1997년~2002년 한국항공대학교 전자정보통신컴퓨터공학부 조교수
2002년~현 재 한국과학기술원 전산학부 교수
관심분야 : Artificial Intelligence, Data Mining, Software Engineering, Biomedical Informatics