JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 29 No. 6, pp. 113-120, June 2024
https://doi.org/10.9708/jksci.2024.29.06.113

Improving Classification Accuracy in Hierarchical Trees via Greedy Node Expansion

Byungjin Lim*, Jong Wook Kim*

*Student, Dept. of Computer Science, Sangmyung University, Seoul, Korea
*Professor, Dept. of Computer Science, Sangmyung University, Seoul, Korea

[Abstract]

With the advancement of information and communication technology, we can easily generate various forms of data in our daily lives. To efficiently manage such a large amount of data, systematic classification into categories is essential. For effective search and navigation, data is organized into a tree-like hierarchical structure known as a category tree, which is commonly seen in news websites and Wikipedia. As a result, various techniques have been proposed to classify large volumes of documents into the terminal nodes of category trees. However, document classification methods using category trees face a problem: as the height of the tree increases, the number of terminal nodes multiplies exponentially, which increases the probability of misclassification and ultimately leads to a reduction in classification accuracy. Therefore, in this paper, we propose a new node expansion-based classification algorithm that satisfies the classification accuracy required by the application, while enabling detailed categorization. The proposed method uses a greedy approach to prioritize the expansion of nodes with high classification accuracy, thereby maximizing the overall classification accuracy of the category tree. Experimental results on real data show that the proposed technique provides improved performance over naive methods.

▸ **Key words**: Text Data Classification, Category Tree, Machine Learning

[요 약]

정보통신 기술이 발전함에 따라 우리는 일상에서 다양한 형태의 데이터를 손쉽게 생성하고 있다. 이처럼 방대한 데이터를 효율적으로 관리하려면, 체계적인 카테고리별 분류가 필수적이다. 효율적인 검색과 탐색을 위해서 데이터는 트리 형태의 계층적 구조인 범주 트리로 조직화되는데, 이는 뉴스 웹사이트나 위키피디아에서 자주 볼 수 있는 구조이다. 이에 따라 방대한 양의 문서를 범주 트리의 단말 노드로 분류하는 다양한 기법들이 제안되었다. 그러나 범주 트리를 대상으로 하는 문서 분류 기법들은 범주 트리의 높이가 증가할수록 단말 노드의 수가 기하급수적으로 늘어나고 루트 노드부터 단말 노드까지의 길이가 길어져서 오분류 가능성이 증가하며, 결국 분류 정확도의 저하로 이어진다. 그러므로 본 연구에서는 사용자의 요구 분류 정확도를 만족시키면서 세분화된 분류를 구현할 수 있는 새로운 노드 확장 기반 분류 알고리즘을 제안한다. 제안 기법은 탐욕적 접근법을 활용하여 높은 분류 정확도를 갖는 노드를 우선적으로 확장함으로써, 범주 트리의 분류 정확도를 극대화한다. 실데이터를 이용한 실험 결과는 제안 기법이 단순 방법보다 향상된 성능을 제공함을 입증한다.

▸ **주제어**: 문서 분류, 범주 트리, 기계학습

• First Author: Byungjin Lim, Corresponding Author: Jong Wook Kim
*Byungjin Lim (dkhsjr@gmail.com), Dept. of Computer Science, Sangmyung University
*Jong Wook Kim (jkim@smu.ac.kr), Dept. of Computer Science, Sangmyung University
• Received: 2024. 05. 07, Revised: 2024. 06. 05, Accepted: 2024. 06. 05.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

정보 통신 기술의 발전으로 우리의 일상생활에서 다양한 모바일 기기를 사용할 수 있게 되었으며, 이로 인해 일상생활에서 다양한 형태의 데이터를 손쉽게 생성할 수 있게 되었다 [1]. 이런 데이터는 문서, 사진, 영상 등 다양한 형태로 나타나며, 이는 개인 뿐만 아니라 기업, 공공기관에서도 발생되는 현상이다. 예를 들어, 사람들은 소셜 미디어에 사진이나 비디오를 업로드하고, 기업들은 업무 보고서나 고객 데이터를 관리하며, 공공기관은 행정 정보나 정책 문서를 디지털화하여 보관한다. 우리의 일상, 기업 활동, 그리고 공공기관의 작업을 통해 생산되는 데이터 양은 계속해서 증가하고 있다. 이러한 데이터를 효과적으로 수집하고 저장하며 분석하는 것은 정보의 활용을 극대화하고, 이는 결국 사회, 경제, 문화의 발전에 크게 기여할 것이다 [2,3].

방대한 양의 문서 데이터를 효과적으로 관리하기 위해서는 문서 데이터를 카테고리별로 분류하는 것이 필수적이다. 다양한 주제를 가진 문서들을 사람이 직접 분류하는 것은 사실상 불가능하다. 이러한 문제를 해결하기 위해 자동 데이터 분류 기법이 필요하게 되었으며, 이와 관련하여 여러 연구가 진행되어 왔다. 초기에는 데이터 마이닝 기법인 의사결정나무와 베이지안 분류기가 주로 사용되었고, 그 뒤로 고차원 데이터를 효과적으로 처리할 수 있는 SVM(Support Vector Machine)과 같은 기계학습 기법들이 도입되었다. 최근에는 더 복잡하고 다양한 데이터 구조를 처리할 수 있는 딥러닝 기반의 자동 분류 기법이 각광받고 있다 [4-6].

효율적인 데이터 탐색과 검색을 위해서 데이터는 주로 트리 형태의 계층적 구조(즉, 범주 트리)로 조직화 된다. 범주 트리에서 루트 노드는 보다 일반화된 개념을 나타내며, 단말 노드로 갈수록 세분화된 개념을 표현한다. 이러한 범주 트리는 뉴스 웹사이트나 위키피디아에서 자주 볼 수 있다 [7]. 그러나, 범주 트리를 사용한 문서 분류는 항목 수가 증가할수록 분류의 정확도가 저하되는 문제가 있다. 범주 트리에서 데이터는 트리의 단말 노드에 위치하는데, 트리의 높이가 높아질수록 단말 노드의 수가 급격히 증가한다. 이로인해 분류 과정에서의 오분류 가능성이 높아지고, 결국 분류 정확도가 낮아지는 문제가 발생한다.

데이터 분류 문제를 해결하기 위해, 자동 분류 알고리즘과 사용자의 수동 개입을 결합한 방법이 활용될 수 있다 [8]. 이 접근법에서는 사용자가 설정한 특정 분류 정확도를 목표로 하여, 그 목표가 달성될 때까지 데이터를 자동으로 분류한다. 그런 다음 초기 자동 분류 과정 후에는 만족스러운 정확도에 도달하지 못한 데이터를 사용자가 직접 수동으로 처리한다. 이 방식의 주요 이점은 사용자가 분류의 정확도를 직접 조절할 수 있다는 것이다. 알고리즘에 의한 초기 처리는 대규모 데이터를 신속하게 처리하는 데 기여하며, 사용자의 수동 개입은 분류의 정밀도를 높이고 세부 조정을 가능하게 한다. 이와 같은 결합 방식은 데이터 분류의 효율성과 정확성을 균형 있게 유지하는 유연한 해결책을 제공할 수 있다.

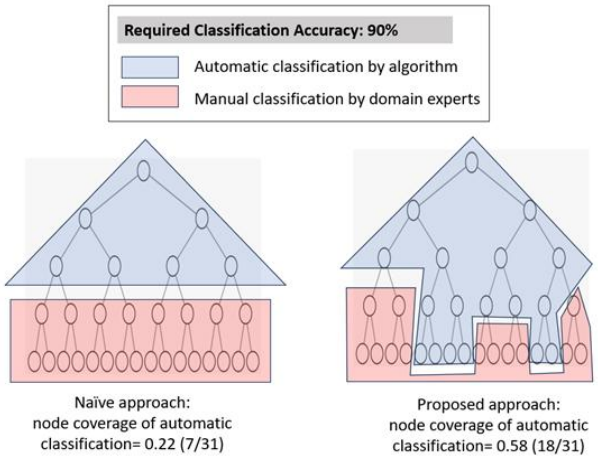


Fig. 1. Motivating example

이 방법을 적용할 때는 사용자가 요구하는 최소 분류 정확도를 만족시키기 위해 알고리즘을 최적화하는 것이 중요하다. 그림 1에서 단순 기법을 사용하는 경우 사용자가 수동으로 처리해야 할 작업이 많아질 수 있다. 그러나 사용자가 설정한 최소 분류 정확도를 기준으로 계층 구조를 효과적으로 확장하면, 수동 분류 작업에 필요한 노력을 줄일 수 있다. 이에 따라, 본 논문에서는 사용자의 요구에 부합하는 최소 분류 정확도를 충족시키기 위해 계층 구조의 노드를 최적화하여 사용하는 새로운 계층적 분류 알고리즘을 제안한다. 본 논문의 제안 기법은 분류 작업의 효율성을 높이고, 수동 분류의 부담을 감소시키는 데 기여할 수 있다. 본 논문의 주요 기여는 다음과 같다.

- 사용자의 요구에 부합하는 분류 정확도를 달성하면서 가능한 세분화된 분류를 가능하게 하는 노드 확장 기반의 새로운 알고리즘을 제안한다.
- 제안 기법은 탐욕 알고리즘(greedy algorithm)을 이용하여 분류 정확도가 높은 노드를 우선적으로 확장한다. 이렇게 하여 각 노드의 자식 노드들의 평균 분류 정확도가 큰 노드를 선택하게 되어, 분류 정확도를 최대화하는 방향으로 범주 트리를 확장한다.

- 다양한 실데이터를 활용하여 제안 알고리즘의 성능을 실험적으로 검증한다.

본 논문은 다음과 같이 구성된다. 2장에서는 본 논문과 관련된 선행 연구들을 설명한다. 3장에서는 본 논문의 제안 기법을 자세히 설명한다. 4장에서는 실데이터를 이용하여 제안 기술의 성능을 실험적으로 평가하고, 마지막으로, 5장에서는 연구 결과를 요약하고 결론을 맺는다.

II. Related Work

문서 분류와 같은 데이터 관리 문제는 점점 증가하는 데이터 볼륨과 복잡성 때문에 중요한 도전 문제가 되었다. 이러한 문제를 해결하기 위해 자동 데이터 분류 기법의 필요성이 커지고 있으며, 이 분야에서는 다양한 연구가 진행되어 왔다. 초기 연구에서는 데이터 마이닝 기법인 의사결정나무 [9]와 베이지안 분류기 [10]가 주로 사용되었다. 그 이후, SVM과 같은 고차원 데이터를 효과적으로 처리할 수 있는 기계학습 기법이 도입되기 시작하였다 [11-12]. SVM은 데이터를 분류하기 위해 고차원 공간에서 최적의 결정 경계를 찾는 방법으로, 특히 복잡한 분류 문제에서 뛰어난 성능을 보인다. 최근에는 더 복잡하고 다양한 데이터 구조를 처리할 수 있는 딥러닝 기반의 자동 분류 기법이 각광받고 있다. 딥러닝 기법은 다층 퍼셉트론(Multi-Layer Perceptron, MLP), 컨볼루션 신경망(Convolutional Neural Network, CNN), 순환 신경망(Recurrent Neural Network, RNN)과 같은 다양한 신경망 구조를 활용하여 특징을 자동으로 학습하고, 이를 바탕으로 분류를 수행한다 [4-6,13-14].

계층적 텍스트 데이터 분류(hierarchical text data classification)는 텍스트 데이터를 사전에 정의된 카테고리 체계(즉, 범주 트리)에 따라 분류하는 기법이다 [15]. 이 방법은 문서나 텍스트가 다양한 계층으로 구성된 범주로 나누어지는 경우 사용된다. [16]의 연구는 계층적 데이터 분류를 위해 딥러닝 기법을 사용하였다. 제안 기법은 텍스트 데이터를 단어 그래프로 변환하고, 이 그래프에 그래프 컨볼루션 연산을 적용함으로써 단어 간의 복잡한 의미적 관계를 효과적으로 분석할 수 있는 그래프-CNN 기반의 모델을 사용하였다. [17]은 워드 임베딩 기술과 다양한 기계학습 알고리즘을 통합하여 계층적 텍스트 데이터 분류의 효율성을 개선하는 방법을 제안하였다. 제안 연구에서는 텍스트 데이터의 의미적 특성을 더 깊이 파악하고, 이를 계층적 구조에 효과적으로 매핑하기 위해 워드 임베딩을 사용하여 단어의 벡터 표현을 생성하고, 이러한 표현들을 기반으로 기계학습 알고리즘을 적용함으로써 분류의 정확도를 높이는 방식을 연구하였다. [18]의 연구에서는 계층적 분류 문제 SVM 분류기의 적용 가능성을 탐구하였다. 연구를 통해, 계층적 구조를 사용하는 SVM 모델이 전통적인 모델보다 약간 더 높은 정확도를 달성하는 것을 확인하였다. [19]에서는 계층 구조를 방향 그래프로 나타내고, 이를 위해 계층 구조 인코더를 도입하여 계층적 텍스트 분류의 정확도를 높였다. MATCH [20]는 메타데이터 및 계층 구조를 문서에 추가하여 분류기 학습을 진행하여 계층적 다중 분류의 정확도를 향상시켰다. [21]은 인지 구조 학습을 계층적 텍스트 분류에 도입하는 연구를 제안했으며, [22]는 계층 구조 간의 관계를 학습하기 위한 계층 구조 임베딩 기법을 제안했다. 또한, [23]에서는 아랍어 문서의 계층적 분류를 위한 새로운 분류 방법을 제안하였다.

III. Proposed Method

본 장에서는 제안 기법을 설명한다. 본 논문의 제안 기법은 크게 3단계로 구성된다.

- 첫 번째 단계에서는 텍스트 문서를 벡터 형태로 변환하는 전처리 과정을 수행한다.
- 두 번째 단계에서는 벡터화된 문서들을 범주 트리의 각 계층별로 분류하고, 이를 통해 트리 내 각 노드의 분류 정확도를 계산한다.
- 마지막 단계에서는 노드별 정확도를 바탕으로, 루트 노드에서 시작하여 범주 트리를 확장하며 문서 분류를 진행한다.

3.1 Text Data Vectorization

텍스트 데이터를 분류하기 위해서는 자연어로 작성된 텍스트를 벡터 형태로 변환하는 과정이 필수적이다. 이를 위해 Word2Vec [24], GloVe [25], BERT [26]와 같은 워드 임베딩 기법이 사용될 수 있다. 본 연구에서는 특히 BERT(Bidirectional Encoder Representations from Transformers) 모델을 이용해 문서를 벡터화한다. BERT는 구글에 의해 개발된 자연어 처리를 위한 모델로서, 다양한 자연어 이해 작업에서 뛰어난 성능을 보여주고 있다. BERT는 특히 문맥상의 의미를 파악하는 능력에서 큰 발전을 이루었으며, 이는 그 구조가 문장의 앞과 뒤를 동시에 고려하는 양방향성을 가지고 있기 때문이다.

BERT는 양방향 언어 모델링을 통해 단어의 문맥을 보다 정확하게 파악할 수 있다. 이는 단방향 언어 모델링을 사용하는 Word2Vec이나 GloVe에 비해 문서의 의미를 더 잘 반영하는 벡터를 생성할 수 있음을 의미한다. 그러므로 본 연구에서는 BERT를 사용하여 텍스트 데이터를 벡터 형태로 변환하였다. 분류 대상 텍스트 데이터셋을 $D = \mathrm{d}_1, \mathrm{d}_2, \ldots, \mathrm{d}_\mathrm{k}$로 표현하자. 이때, $d_i \in D$는 i번째 문서를 의미한다. 또한, 벡터화된 텍스트 데이터들의 집합을 $V = \mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_\mathrm{k}$로 표현하자. 여기서 $v_i \in V$는 BERT를 통해 벡터화된 i번째 문서 $d_i \in D$의 벡터를 나타낸다. 이 벡터 집합 V는 다음 단계에서 분류 작업을 위한 입력 데이터로 사용된다.

3.2 Training Classifier for Each Node

본 절에서는 범주 트리의 각 노드별로 다중 분류(Multiclass Classification)를 수행하며, 그 과정에서 범주 트리의 각 노드별로 분류기를 학습한다. 범주 트리의 특정 노드 n_i의 자식 노드들의 집합을 $Chld(n_i)$라 가정하자. 또한 벡터화된 텍스트 데이터들의 집합 $V = \mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_\mathrm{k}$중에서, n_i에 속하는 데이터들의 집합을 $V_{n_i} \subset V$이라 가정하자. 즉, V_{n_i}은 n_i의 자손 노드로 분류되는 데이터들의 집합에 해당한다. 본 절에서는 각각의 노드 n_i에 대하여, 데이터 집합 V_{n_i}를 n_i의 자식 노드들로 분류하는 다중 분류기 c_{n_i}을 학습한다.

Algorithm 1. Training classifier for each non-leaf node

input: a set of vectorized text data V
a category tree T
output: a set of classifier C

1: $C = \varnothing$
2: **for** each node $n_i \in T$
3: **if** n_i is not leaf node
4: V_{n_i} = Extract_Data(V, n_i)
5: c_{n_i} = TrainClassifier(V_{n_i}, $Chld(n_i)$)
6: $C = C \cup \{c_{n_i}\}$
7: **return** C

알고리즘 1은 범주 트리의 각 노드별로 다중 분류기를 학습하는 의사코드를 나타낸다. 알고리즘 1의 입력은 벡터화된 텍스트 데이터들의 집합 $V = \mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_\mathrm{k}$와 범주 트리 T이며, 출력은 분류기 집합 C이다. 1번 줄(Line)에서 C를 공집합으로 초기화한다. 4번 줄에서는 각각의 노드 n_i에 속하는 데이터들의 집합 V_{n_i}를 구한 후, 5번 줄에서 다중 분류기 c_{n_i}을 학습한다. 6번 줄에서는 c_{n_i}를 C에 추가한다. 범주 트리의 모든 노드에 대하여 위의 과정을 반복한 후, 마지막으로 분류기 집합 C를 반환한다.

3.3 Expanding Each Node using Greedy Algorithm

이전 단계에서 범주 트리의 각 노드에 대하여 분류기를 학습하였다. 본 절에서는 이를 이용하여, 사용자가 사전에 설정한 요구 분류 정확도를 만족하는 동시에 범주 트리의 노드를 최대로 확장할 수 있는 탐욕 알고리즘 기반의 방법을 제안한다.

그림 2는 노드 확장 기법을 직관적으로 보여준다. 초기에 1번 노드가 확장되어, 텍스트 데이터가 2번과 3번 노드로 분류된다고 가정한다. 이후 범주 트리에서 순차적으로 확장 가능한 노드는 2번과 3번이다. 만약 2번 노드를 확장하면, 분류기 c_1, c_2를 사용하여 데이터를 3번, 4번, 5번 노드로 분류한다. 반면, 3번 노드를 확장할 경우, 분류기 c_1, c_3를 사용하여 데이터를 2번, 6번, 7번 노드로 분류한다. 이때, 탐욕적 알고리즘을 사용하면 분류 정확도가 높아지는 방향으로 노드를 확장하게 된다.

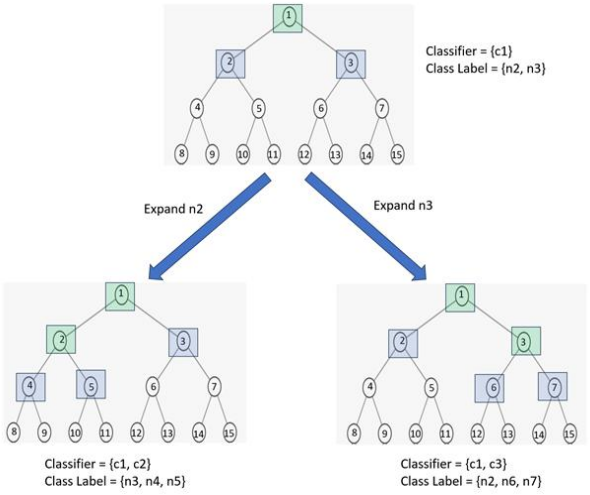


Fig. 2. Intuitive example of node expansion

본 논문의 제안 기법은 그림 2의 방식을 따른다. 알고리즘 2는 범주 트리의 노드 확장을 구현한 의사코드를 나타낸다. 알고리즘 2의 입력은 벡터화된 텍스트 데이터들의 집합 V, 분류기 집합 C (즉, 알고리즘 1의 출력), 그리고 사전에 정의한 최소 분류 정확도 θ이다. 또한, 알고리즘 2의 출력은 최종 분류기 집합 C_{out}와 클래스 레이블들의 집합 N_{class}이다. 1번 줄과 2번 줄에서 C_{out}과 N_{class}를

각각 루트 노드의 분류기, 루트 노드의 자식 노드들만을 포함하도록 초기화 한다.

4번 줄에서 13번 줄까지는 그림 2에서 설명한 바와 같이 탐욕 알고리즘을 사용하여 가장 분류 정확도가 높은 노드로 확장을 진행한다. 이때, 노드 확장 후, 새로 계산한 분류 정확도 acc_{best}가 최소 분류 정확도 θ보다 큰 경우, C_{out}과 N_{cand}를 갱신한다 (14~17번 줄). 반면, 노드 확장 후, 분류 정확도 acc_{best}가 최소 분류 정확도 θ보다 작은 경우 노드 확장을 종료한다 (18~19번 줄).

Algorithm 2. Expanding nodes in category tree

input: a predefined minimum accuracy θ
a set of classifier for each node C
a set of vectorized text data V
output: a set of classifier C_{out}
a set of class labels N_{class}

1: C_{out} = {a classifier of root node}
2: N_{class} = {child nodes of a root node}
3: **while** (True)
4: acc_{best}= 0
5: C_{best} = $\varnothing$
6: n_{best} = null
7: **for** each node $n_i \in N_{class}$
8: $C_{current}$ = $C_{out} \cup \{c_{n_i}\}$
9: $acc_{current}$= RunClassification($C_{current}$, V)
10: **if** ($acc_{current} > acc_{best}$)
11: acc_{best}= $acc_{current}$
12: C_{best} = $C_{current}$
13: n_{best} = n_i
14: **if** ($acc_{best} > \theta$)
15: C_{out} = C_{best}
16: N_{class} = N_{class} -{n_{best}}
17: N_{class} = N_{class} $\cup$ {child nodes of n_{best}}
18: **else**
19: break
20: **return** C_{out}, N_{class}

그림 3은 알고리즘 2를 사용한 노드 확장의 예시를 보여준다. 그림 3(a)는 초기화 과정에 해당한다. 그림 3(b)에서는 노드 3으로 확장이 진행된 것을 가정한 경우이고, 그림 3(c)는 이어서 노드 6으로 확장이 진행된 것을 가정한 경우이다. 마지막으로 그림 3(d)에서 노드 2로 확장이 진행되지만, 이때 새롭게 구한 분류 정확도가 최소 분류 정확도보다 낮다고 가정하면(즉, $acc_{best} < \theta$), 이때 노드 확장을 멈추고, 이전 단계의 C_{out}과 N_{class}를 반환한다.

알고리즘 2를 통하여 분류기 집합 C_{out}과 클래스 레이블들의 집합 N_{class}을 구한 후, 분류는 다음과 같이 진행된다. 분류 대상이 되는 문서를 d_{target}이라 가정하자. 또한, 그림 3의 예를 가정하자 (즉, $C_{out} = c_1, c_3, c_6$, $N_{class} = n_2, n_7, n_{12}, n_{13}$). 이때, d_{target}이 C_{out}을 통해 단말 노드인 n_{12}로 분류되면, 사용자의 수동 분류를 없이 분류가 종료된다. 만일 d_{target}이 비단말 노드인 n_7로 분류되면, 사용자의 수동 분류를 통해 d_{target}을 단말 노드 (즉, n_{14} 또는 n_{15})까지 분류한다.

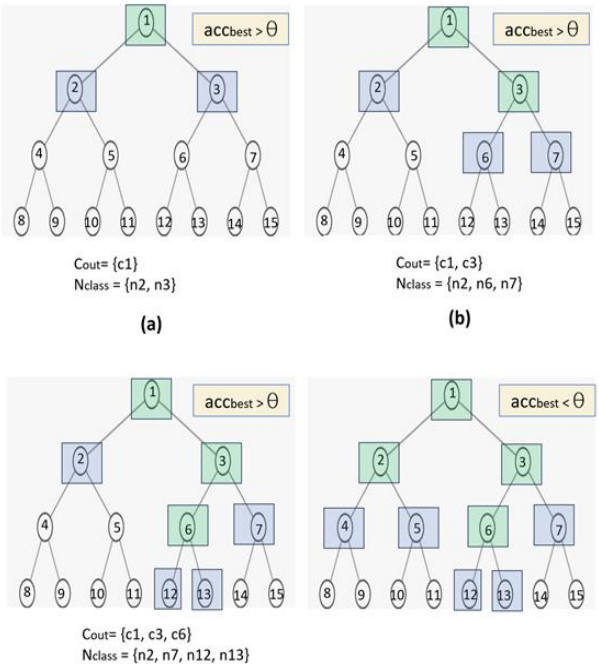


Fig. 3. Example of Node Expansion

IV. Experiments and Results

4.1 Experiment Setup

본 연구에서는 성능 평가를 위해 다음의 실데이터를 이용하였다.

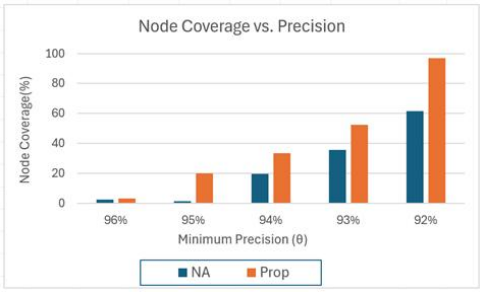
- DBpedia 데이터셋: DBpedia [27]에서 316,392개의 문서를 수집하였다. DBpedia에서 사용하는 범주 트리는 총 200개의 노드로 구성되어 있으며, 트리의 레벨은 7이다.
- ACM 데이터셋: ACM Digital Library [28]에서 29,698개의 논문 요약을 수집하였다. 논문은 최대 4레벨, 59개 노드로 구성된 범주 트리로 분류된다.

DBpedia 데이터셋을 사용하는 실험은 충분한 학습 데

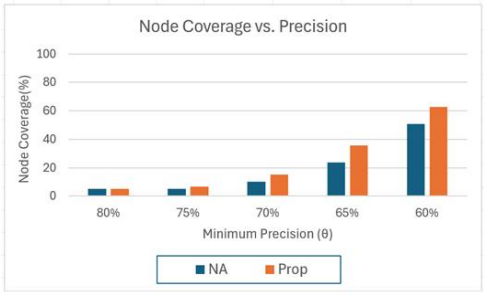
이터로 인해 높은 분류 정확도를 달성할 수 있는 환경에서 제안 기법의 성능을 평가하는 것을 목적으로 하고 있다. 반면, ACM 데이터셋을 사용하는 실험은 학습 데이터가 부족하여 낮은 분류 정확도가 예상되는 환경에서 제안 기법의 성능을 평가하는 것을 목적으로 한다.

실험에서는 본 논문에서 제안하는 기법(Prop)과 레벨별로 트리 노드를 확장하는 단순 기법(NA)의 성능을 비교하였다. 예를 들어, 단순 기법은 그림 2에 나타난 범주 트리의 경우 노드1, 노드2, 노드3,... 노드15와 같이 각 레벨을 순차적으로 확장하는 방식이다. 또한, 각 노드마다 필요한 다중 분류기로 SVM을 사용하였다.

4.2. Experimental Results



(a) DBpedia Dataset

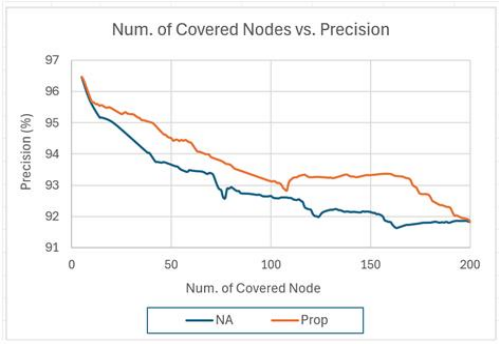


(b) ACM Dataset

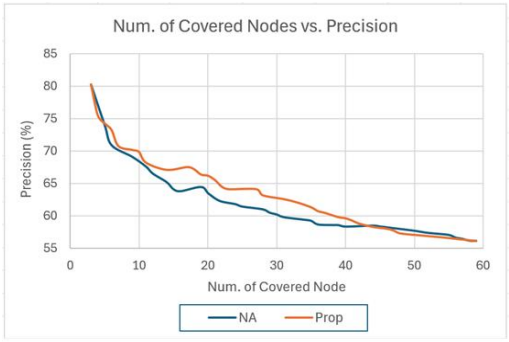
Fig. 4. Node coverage for varying required minimum precision (θ)

그림 4는 사전에 정의한 최소 분류 정확도(θ)에 따라 변하는 노드 커버리지(node coverage)를 보여준다. 노드 커버리지는 범주 트리의 전체 노드 중에서 각 기법이 최소 분류 정확도를 충족하며 노드를 최대로 확장할 수 있는 비율을 나타낸다. 노드 커버리지가 100%인 경우, 전체 노드가 확장 가능해지므로, 알고리즘을 통해 데이터를 전적으로 분류할 수 있다. 따라서 노드 커버리지가 높을수록 사용자의 수동 분류 작업이 감소한다.

그림 4의 실험 결과에 따르면, 제안 기법은 단순 노드 확장 방식에 비해 모든 최소 분류 정확도 수준에서 더 우수한 성능을 나타내고 있다. 이는 제안 기법이 각 노드의 확장 단계에서 분류 정확도를 고려하여 노드를 확장하는 전략을 채택하고 있기 때문이다. 이로 인해 범주 트리의 노드를 분류 정확도가 높은 방향으로 확장할 수 있다. 특히, 제안 기법은 높은 분류 정확도가 요구되는 DBpedia 데이터셋과 낮은 분류 정확도의 환경이 예상되는 ACM 데이터셋 모두에서 우수한 성능을 보이고 있다. 이러한 결과는 제안 기법이 다양한 환경에서 일관된 성능을 제공할 수 있음을 보여주며, 실제 응용 분야에서의 폭넓게 적용 가능함을 보여준다.



(a) DBpedia Dataset



(b) ACM Dataset

Fig. 5. The number of covered nodes in a category tree vs. classification precision

그림 5의 실험 결과는 범주 트리의 각 노드를 차례로 확장하며 분류 정확도를 측정한 결과를 보여준다. 실험에서는 DBpedia 데이터셋을 최대 200개의 노드까지, ACM 데이터셋을 최대 59개의 노드까지 확장하는 것이 가능하다. 그림 5에서 볼 수 있듯이, 대부분의 경우에서 제안 기법을 사용한 노드 확장 방식이 단순 레벨별 노드 확장 방식보다 우수한 성능을 나타내고 있다. 이 결과는 제안 기법이 노드를 확장할 때 더 높은 분류 정확도를 달성할 수 있는 방법론을 채택하고 있음을 입증한다. 특히, 제안 기법은 각 노드의 확장 시점에서 최적의 결정을 내리기 위해 노드별

분류 정확도를 고려하여, 분류 정확도가 높은 방향으로 노드 확장을 진행하기 때문이다. 이러한 접근 방식은 DBpedia와 같이 더 많은 노드가 있는 데이터셋에서 특히 효과적이며, ACM과 같이 상대적으로 노드 수가 적은 데이터셋에서도 그 효과를 확인할 수 있다. 이는 본 연구의 제안 기법이 범주 트리의 노드를 확장하는 과정에서 단순한 순차적 확장보다 훨씬 효과적인 방법을 제공함을 보여주며, 다양한 분류 환경에서의 적용 가능성을 보여준다.

본 장의 실험 결과는 제안 기법이 사용자가 설정한 최소 분류 정확도를 만족하면서, 단순 기법에 비해 범주 트리에서 더 많은 노드를 커버할 수 있다는 것을 증명한다. 또한, 제안 기법은 분류 정확도가 다양한 환경에서 일관된 성능을 제공할 수 있다는 것을 보여준다.

V. Conclusions

본 논문에서는 사용자의 요구에 부합하는 분류 정확도를 달성하면서 가능한 세분화된 분류를 가능하게 하는 노드 확장 기반의 새로운 알고리즘을 제안하였다. 제안 기법은 탐욕 알고리즘을 이용하여 분류 정확도가 높은 노드를 우선적으로 확장함으로써, 분류 정확도를 최대화하는 방향으로 범주 트리를 확장한다. 실데이터를 이용한 실험 결과는 본 논문의 제안 기법이 다양한 최소 분류 정확도에서 단순 기법보다 우수함을 입증하였다.

REFERENCES

[1] Z. Lv, H. Song, P. Basanta-Val, A. Steed, and M. Jo. Next-Generation Big Data Analytics: State of the Art, Challenges, and Future Research Topics. IEEE Transactions on Industrial Informatics, Vol. 13, 2017. DOI: 10.1109/TII.2017.2650204

[2] Z. Yang and Z. Ge. On paradigm of industrial big data analytics: From evolution to revolution. IEEE Transactions on Industrial Informatics, Vol. 18, 2022. DOI: 10.1109/TII.2022.3190394

[3] J. Wang, C. Xu, J. Zhang, and R. Zhong. Big data analytics for intelligent manufacturing systems: A review, Vol. 62, 2022. DOI: 10.1016/j.jmsy.2021.03.005

[4] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao. Deep Learning-based Text Classification: A Comprehensive Review. ACM Computing Surveys, Vol. 54, 2021. DOI: 10.1145/3439726

[5] C. Wang, P. Nulty, and D, Lillis. A Comparative Study on Word Embeddings in Deep Learning for Text Classification. Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval, 2020. DOI: 10.1145/3443279.3443304

[6] Q. Li, H. Peng, J. Li, C. Xia, R. Yang, L. Sun, P. S. Yu, and L. He. A Survey on Text Classification: From Traditional to Deep Learning. ACM Transactions on Intelligent Systems and Technology. Vol. 13, 2022. DOI: 10.1145/3495162

[7] S. Daud, M. Ullah, A. Rehman, T. Saba, R. Damasevicius, and A. Sattar. Topic Classification of Online News Articles Using Optimized Machine Learning Models. Computers, Vol. 12, 2022. DOI: 10.3390/computers12010016

[8] J Andersen. Why Do We Need Domain-Experts for End-to-End Text Classification? An Overview. Proceedings of the 15th International Conference on Agents and Artificial Intelligence, 2023. DOI: 10.5220/0011605900003393

[9] K. M. Almunirawi and A. Y. A. Maghari. A Comparative Study on Serial Decision Tree Classification Algorithms in Text Mining. International Journal of Intelligent Computing Research, Vol. 7, 2016. DOI: 10.20533/ijicr.2042.4655.2016.0093

[10] S. Wang, L. Jiang, and C. Li. Adapting naive Bayes tree for text classification. Knowledge and Information Systems, Vol. 44, 2015. DOI: 10.1007/s10115-014-0746-y

[11] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. Neurocomputing, Vol. 408, 2020. DOI: 10.1016/j.neucom.2019.10.118

[12] K. Li, J. Xie, X. Sun, Y. Ma, and H. Bai. Multi-class text categorization based on LDA and SVM. Procedia Engineering. Vol. 15, 2011. DOI: 10.1016/j.proeng.2011.08.366

[13] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits. Is bert really robust? Natural language attack on text classification and entailment. 2, 2019. DOI: 10.48550/arXiv.1907.11932

[14] R. Johnson and T. Zhang. Deep pyramid convolutional neural networks for text categorization. In Proceedings of the Annual Meeting of the Association for Computational Linguistics. 2017. DOI: 10.18653/v1/P17-1052

[15] C. N. Silla and A. A. Freitas. A survey of hierarchical classification across different application domains. Data Mining and Knowledge Discovery. Vol. 22, 2011. DOI: 10.1007/s10618-010-0175-9

[16] H. Peng, J. Li, Y. He, Y. Liu, M. Bao, L. Wang, Y. Song, and Q. Yang. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. Proceedings of WWW, 2018. DOI: 10.1145/3178876.3186005

[17] R. A. Stein, P. A. Jaques, and J. F. Valiati. An analysis of hierarchical text classification using word embeddings. Information Sciences. Vol. 471, 2019. DOI: 10.1016/j.ins.2018.09.001

[18] S. Dumais and H. Chen. Hierarchical classification of Web content. Proceedings of the ACM SIGIR conference on Research and development in information retrieval. 2000. DOI: 10.1145/345508.345593

[19] J. Zhou, C. Ma, D. Long, G. Xu, N. Ding, H. Zhang, P. Xie, and G. Liu. Hierarchy-Aware Global Model for Hierarchical Text Classification. Proceedings of the Annual Meeting of the Association for Computational Linguistics, July 2020. DOI: 10.18653/v1/2020.acl-main.104

[20] Y. Zhang, Z. Shen, Y. Dong, K. Wang, and J. Han. MATCH: Metadata-Aware Text Classification in A Large Hierarchy. Proceedings of the Web Conference 2021, New York, NY, USA, April 2021. DOI: 10.1145/3442381.3449979

[21] B. Wang, X. Hu, P. Li, and P.S. Yu. Cognitive structure learning model for hierarchical multi-label text classification. Knowledge-Based Systems. Vol. 218, 2021. DOI: 10.1016/j.knosys.2021.106876

[22] H. Peng, J. Li, S. Wang, L. Wang, Q. Gong, R. Yang, B. Li, P.S. Yu, and L. He. Hierarchical Taxonomy-Aware and Attentional Graph Capsule RCNNs for Large-Scale Multi-Label Text Classification. IEEE Transactions on Knowledge and Data Engineering. Vol. 33, 2021. DOI: 10.1109/TKDE.2019.2959991

[23] N. Aljedani, R. Alotaibi, and M. Taileb. HMATC: Hierarchical multi-label Arabic text classification model using machine learning. Egyptian Informatics Journal. Vol. 22, 2021. DOI: 10.1016/j.eij.2020.08.004

[24] B. Jang, I. Kim, and J.W. Kim. Word2vec convolutional neural networks for classification of news articles and tweets. Plos One, Vol. 14, no. 8, 2019. DOI: 10.1371/journal.pone.0220976

[25] J. Pennington, R. Socher, and C. Manning. GloVe: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 2014. DOI: 10.3115/v1/D14-1162

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. 2018. DOI: 10.48550/arXiv.1810.04805

[27] DBpedia. https://en.wikipedia.org/wiki/DBpedia

[28] ACM Digital Library. https://dl.acm.org/

Authors

Byungjin Lim received the B.S. degree from Sangmyung University in 2023, where he is currently pursuing the master's degree with the Department of Computer Science. His research mainly focuses on Artificial Intelligence.

Jong Wook Kim received the Ph.D. degree in Computer Science Department, Arizona State University, in 2009. He was a Software Engineer with the Query Optimization Group, Teradata, from 2010 to 2013.

Dr. Kim is currently an Associate Professor with the Department of Computer Science at Sangmyung University. His primary research interests include the area of data privacy, distributed databases, and query optimization.