Global Creative Leader: Education & Learning
2019, Vol. 9, No. 4, 67-88.
http://dx.doi.org/10.34226/gcl.2019.9.4.67

Network Analysis for the Evaluation of Software Education

Park, Geum-Ju[1] (Dankook University)

< ABSTRACT >

The purpose of this study was to analyze students' needs by conducting a network analysis on the evaluation results from related software education curriculums. The study tested, compared and performed several analyses; core key words analysis, co-occurrence frequency, network analysis, and centrality analysis on specific periods of curriculums in which the software program and textbooks were changed between the first semester of 2017 and the first semester of 2018. As a result of the study, the software curriculums were determined to have been based on the way of design thinking. More specifically, the study found that the curricula was implemented to improve creative thinking, to maintain and to promote team activities, to encourage motivation for participating in software education, to estimate appropriate difficulties and quantity of tasks, to strengthen learning based using scratch, to provide more engaging in active team activities, and to provide feedback on learning outcomes. Future research proposed an improvement plan of software education that could reflect each majors' characteristics through network analysis by college majors.

Key Words : Network analysis, text analysis, coding education, software education, college students

1) Corresponding Author: Park, Geum-Ju, Professor, Dankook University, 119 Dandae-ro, Dongnam-gu, Cheonan-si, Chungnam, Korea, 31116 / E-mail: 12171176@dankook.ac.kr

Global Creative Leader: Education & Learning
2019, Vol. 9, No. 4. 67-88.
http://dx.doi.org/10.34226/gcl.2019.9.4.67

대학 소프트웨어 교육의 서술형 강의평가에 대한 네트워크 분석

박금주[1] (단국대학교)

〈 요 약 〉

본 연구의 목적은 대학 소프트웨어 교육의 강의평가에 대해 네트워크 분석을 실시하여 수강생의 요구사항을 분석하는 데 있다. 소프트웨어 교육에 활용한 소프트웨어 프로그램 교체와 교재의 변화가 병행되었던 2017년 1학기와 2018년 1학기에 대해 핵심단어 출현빈도와 공출현빈도 산출, 네트워크 분석, 텍스트 중심성 산출, 네트워크 비교를 실시하였다. 연구결과, A대학의 소프트웨어 교육은 디자인 싱킹을 기반으로 창의적 사고 프로그램을 시행하고 조별활동 중심의 교육을 유지하면서, 소프트웨어 교육에 대한 동기부여, 적절한 난이도와 분량의 과제 부여, 스크래치 활용법에 대한 학습 강화, 적극적인 팀활동 참여 유도, 학습결과에 대한 피드백 제공이 요구되었다. 향후 연구에서는 전공 계열별 네트워크 분석을 통해 계열별 특성을 반영한 소프트웨어 교육 개선안을 제시하고자 한다.

주요어 : 네트워크 분석, 텍스트 분석, 코딩교육, 소프트웨어 교육, 대학생

1) 교신저자: 박금주, 교수, (31116) 충남 천안시 동남구 단대로 119, 단국대학교 / E-mail: 12171176@dankook.ac.kr
논문투고: 2019. 8. 21 / 심사일자: 2019. 8. 27 / 게재확정일자: 2019. 9. 10

Ⅰ. 서 론

미래 사회는 컴퓨터를 기반으로 소프트웨어가 사회를 움직이는 역할을 담당할 것이다. 현재 우리의 생활환경에서도 컴퓨터가 탑재되어 있는 기기를 다루거나 컴퓨터를 다루는 일이 보편화 되어 있다. 컴퓨터의 활용을 뛰어 넘어 컴퓨터 언어를 익혀 프로그램을 작성하고 적용할 줄 아는 능력을 필요로 하는 시대에 살고 있으며, 향후 우리 사회에서 컴퓨터의 활용은 더욱 늘어날 것이다. 이러한 능력은 소프트웨어 교육을 통해 갖출 수 있으며 교육과정에서 문제해결능력, 창의력 등을 향상시킬 수 있다.

세계 주요 국가도 소프트웨어 교육을 실시하고 있다. 미국은 모든 학생에게 소프트웨어를 가르치는 ‘Computer Science for All’ 계획에 따라 소프트웨어 교육을 실시하고 있다. 영국은 5~16세 대상 교육 프로그램에 소프트웨어 과목을 필수로 선정하였다. ‘코딩을 못하면 국가의 미래가 없다’라고 말하며 전 국민에게 소프트웨어 교육을 전파하고자 노력하고 있다. 일본 역시 2012년부터 고등학교에서 ‘정보’ 필수 과목으로 지정하여 프로그래밍 교육을 시행하고 있다. 이스라엘은 1994년부터 소프트웨어 과목을 정규과목에 포함시켜 교육하고 있다(대한민국교육부, 2015; 홍정민, 2017). 이렇게 세계 주요 국가들은 소프트웨어 교육의 중요성을 인식하고 우리나라 보다 앞서 소프트웨어 인재양성을 위해 발 빠르게 움직이고 있다.

우리나라는 국제전기통신연합(International Telecommunication Union: ITU)이 해마다 발표하는 정보통신기술발전지수분야에서 10년간 1,2위를 기록하고 있지만(김지선, 2018.6.10), 세계주요 국가보다 소프트웨어 교육에 있어서는 뒤떨어지고 있다. 고등교육기관인 대학은 산업현장의 인력을 양성하는 기관으로서 우리 사회가 필요로 하는 인재를 양성하는 역할을 담당하고 있지만, 소프트웨어 교육에 있어서는 3년 내외의 짧은 기간 동안 교육내용이나 방법에 변화를 주며 시행착오를 겪고 있다(유홍준, 2015; 이민석, 2016; 김완섭, 2017a; 한옥영, 김재현, 2017; 박금주, 최영준, 2018). 또한 각 대학의 교양대학에서 교양선택으로 운영하던 소프트웨어 관련 교과를 교양필수로 전환하여 실시한 것은 최근 1~3년간의 변화로, 수강생 대상 수요조사를 실시하고, 창의적 사고와 문제해결력 향상을 목표로 소프트웨어 교육이 이루어질 수 있도록 노력하고 있다.

선행연구 중 대학의 구체적인 소프트웨어 교육을 살펴볼 수 있는 사례 연구나 교과개설을 위한 연구는 각 대학 사례를 통해 확인할 수 있고(성정숙, 김수환, 김현철, 2015; 유홍준, 2015; 서응교, 2017; 서주영, 신승훈, 구은희, 2017; 송연옥, 2017; 신희성, 서응교, 오경선, 정혜진, 박소현, 2018), 수강생 대상 설문을 실시하여 요구를 파악하는 연구(최정원, 이영준,

2014; 김수환, 2015; 성정숙 외, 2015; 이민석, 2016; 김주연, 2017; 나정은, 2017a; 나정은, 2017b; 김완섭, 2017a; 김완섭, 2017b; 서주영 외, 2017; 오미자, 2017)는 선택형 문항 중심의 설문을 활용하였다. 선행연구에서 활용한 설문은 선택형 문항 중심으로 구성하여 수강생의 다양한 의견 수렴에는 한계가 있다. 반면, 텍스트로 되어 있는 서술형 문항에서는 수강생의 다양한 의견을 살펴볼 수 있다. 또한 전체 강의평가의 평균에 미치지 못하는 소프트웨어 교육 강의평가의 결과에 대한 원인을 찾기 위해서는 서술형 문항의 답변을 살펴볼 필요가 있다.

이에 본 연구에서는 A대학의 소프트웨어 교육에 대한 서술형 강의평가 내용에 네트워크 분석을 적용하여 수강생의 요구사항을 파악하고 소프트웨어 교육의 개선 방안을 제시하고자 한다. 네트워크 분석은 텍스트 자료를 수치화하고 시각화가 가능하며(민혜리, 윤한솔, 2017), 사회, 문화 차원의 분석에서 단어와 단어 간 복잡한 관계를 분석하는 범위로 확장되어 활용되고 있다(김용학, 김영진, 2016). 서술형 강의평가 내용 분석을 위해, 첫째, 서술형 강의평가결과를 입수하여 네트워크 분석이 가능한 형태의 텍스트 파일로 다듬는 전처리 과정을 수행하였다. 둘째, 단어의 출현빈도를 산출하고 빈도가 높은 단어를 중심으로 핵심단어를 선정하여 핵심단어를 중심으로 공출현빈도를 도출하였다. 셋째, 공출현빈도와 핵심단어 출현빈도를 활용해 네트워크 분석을 실시하고 텍스트 중심성 분석을 실시하였다. 넷째, 소프트웨어 교육 운영의 변화 전후를 살펴볼 수 있는 2017년 1학기와 2018년 1학기에 대해 소프트웨어 교육의 좋은 점과 개선점에 대한 네트워크를 비교・분석하였다.

Ⅱ. 대학의 소프트웨어 교육

문제해결능력, 창의력이 화두가 된 현재 우리 사회에서, 컴퓨팅 사고력을 향상시키기 위한 교육은 미래 사회를 이끌어 갈 세대를 위한 필수 교육으로 인식되고 있다. 소프트웨어 교육은 컴퓨팅 사고력을 향상시킬 수 있다. 컴퓨팅 사고력(computational thinking)은 데이터 수집・분석・표현, 주어진 문제에 대해 해결가능한 단위로 분해하는 문제 분해, 문제해결을 위해 필요한 요소를 단순화할 수 있는 추상화, 문제해결 과정을 표현할 수 있는 알고리즘 및 프로시저, 컴퓨터에서 수행 가능하도록 표현하는 자동화, 문제해결 모델을 실행 가능하도록 방법을 구안할 수 있는 시뮬레이션, 동시에 여러 작업이 함께 수행할 수 있도록 제작하는 병렬화로 구성된다(Wing, 2008; 국가과학기술자문회의, 2017; 이재호, 2017). 이와 같이 컴퓨팅 사고력을 구성하는 능력은 소프트웨어 교육을 통해 문제해결을 위한 과정에서 수정・보완사항의 재적용, 반복수행을 거치면서 향상시킬 수 있다(최성경, 2019).

각 대학은 소프트웨어 교육을 교양교육으로 시행하고 있다. 선행연구에 소개된 각 대학의 소프트웨어 교육 현황을 살펴보면 다음과 같다. A대학의 소프트웨어 교육은 2016년 1학기부터 시작되었으며 2016년 2학기까지는 교수자 재량에 의한 강의 운영으로, 활용하는 프로그램이나 교수방법에 제한이 없었다. 이론 강의와 실습이 함께 이루어지는 강의가 주를 이루었고 활용 프로그램으로는 파이선, 앱인벤터, 스크래치를 활용하였다.

2017년 1학기부터는 공통된 교재, 활용 프로그램 지정, 새로운 교수법을 도입하였다. SAP사의 전사적 자원 관리(ERP: Enterprise Resource Planning) 프로그램을 활용하였으며, 전체 강좌에서 활용할 수 있는 공통된 교재를 사용하였다. 교재는 소프트웨어를 다루기 전에 디자인 싱킹(Design Thinking)을 통해 창의적 사고를 할 수 있게 하고, 후반부에서는 전사적 자원 관리 프로그램을 설치하고 활용하는 방법을 다루었다. 교수법은 강의식, 팀기반학습(TBL: Team-Based Learning), 플립러닝(flipped learning)을 활용하였다. 디자인 싱킹에 대한 내용은 강의식으로 진행하였고, 전사적 자원 관리 프로그램을 다룰 때 필요한 용어와 기본적인 사항은 동영상으로 제작해, 대학 내 e러닝 시스템을 통해 강의 전 학습하게 하고, 오프라인 강의에서 프로그래밍을 하면서 질의와 응답을 통해 의문점을 해결하는 플립러닝을 활용하였다. 교재는 팀별 활동이 가능하도록 제작되었고, 교육은 인문, 사회, 예·체능 계열의 비이공계 학생을 대상으로 하였다.

2018년 1학기에는 사회, 인문, 공학, 자연, 예·체능별로 전공별 특성을 반영한 교재를 사용하였으며 디자인 싱킹에 대한 내용이 강화되고, 활용 프로그램은 전사적 자원 관리 프로그램에서 스크래치로 교체하였다. 강의는 8주간 창의적 사고와 디자인 싱킹을 다루고 5주간 코딩 실습으로 구성하였다. 소프트웨어 교육인 ‘창의적 사고와 코딩’을 4차 산업혁명 기초 과목 중 소프트웨어 관련 교양필수과목으로 지정하여 1학년 전체 학생을 대상으로 교육하고, 1학기는 디자인 싱킹과 프로젝트 수행 중심, 2학기는 알고리즘을 중심으로 실시하였다.

B대학은 2016년부터 온·오프라인 교육을 병행해 앱인벤터, 스크래치, 엔트리, 파이선을 활용한 소프트웨어 교육을 실시하고 있다. 교재는 컴퓨팅 사고를 활용한 문제해결을 중심으로 구성되어 있으며 실습에서는 스크래치와 파이선을 주로 사용하고 있다(김완섭, 2017a).

C대학도 2016년부터 ‘컴퓨터 사고와 SW코딩’이라는 과목명으로 전 계열의 1학년 학생을 대상으로 소프트웨어 교육을 실시하고 있다. 강의 전 문제해결 활동을 위한 동영상 강의를 미리 학습하고 오프라인 강의에서 토론이나 실습, 문제풀이가 이루어졌다. ‘문제해결과 알고리즘’은 각 계열별로 특화된 교육으로 선수지식과 경험을 바탕으로 문제해결을 할 수 있도록 구성되어 있다. 활용 도구로는 인문계와 예·체능계열은 엔트리를 사용하고 자연계열은 엔트리와 플레이 봇을 활용하고 있다(유홍준, 2015).

D대학은 인문대학의 1학년생을 대상으로 2개 학기에 걸쳐 기초필수과목으로 수강하도록 하였다. 첫 학기는 이론 중심으로 컴퓨터와 소프트웨어의 정의로 시작하여 문제해결 절차와 일상생활에서 접할 수 있는 소프트웨어 관련 주제를 다루었다. 두 번째 학기에는 R프로그램을 활용해 실습 중심의 수업으로 진행하여 데이터를 처리하고 분석하는 역량을 갖추도록 하는 데 중점을 두었다(서주영 외, 2017).

지금까지 살펴본 대학의 소프트웨어 교육은 1학년 학생을 대상으로 실시하고 있으며 소프트웨어를 활용한 코딩에 앞서, 컴퓨팅 사고를 활용한 문제해결력 향상 프로그램이 선행되었다. 또한 비전공자 학생이 포함된 1학년 전체 학생이 교육에 참여하므로 쉽게 익혀 활용할 수 있는 프로그램을 사용하고, 온·오프라인 교육을 병행하는 블렌디드 러닝을 활용하고 있다.

Ⅲ. 대학의 소프트웨어 교육에 대한 요구분석

대학 소프트웨어 교육 관련 선행연구에서 수강생을 대상으로 시행된 요구분석은 <표 1>과 같이 소프트웨어 교육에 대한 인식, 소프트웨어 교육의 어려움, 소프트웨어 교육의 효과성으로 나누어 볼 수 있다.

첫째, 소프트웨어 교육에 대한 인식은 성정숙 외(2015), 이민석(2016), 김완섭(2017a, 2017b), 나정은(2017a, 2017b), 서주영 외(2017), 오미자(2017), 김주연(2018)의 연구에서 살펴볼 수 있다. 성정숙 외(2015)는 인문계열의 소프트웨어 교육 초보 학습자의 특성 분석 연구에서 스크래치에 대한 흥미 정도, 프로그래밍 능력 정도, 스크래치 활용 용이성, 도움 정도, 추가 수강 여부, 향상된 능력에 대해 조사하였다. 수강생들은 수업 후반으로 가면서 난이도가 증가함과 동시에 흥미와 재미가 하락하는 경향을 보였으나, 개인 과제를 수행하며 성취감이 향상되는 것으로 나타났다.

이민석(2016)은 신입생 대상 설문을 통해 대학 입학 전 컴퓨터 언어 학습 경험, 소프트웨어 배움 의지, 활용 프로그램에 대한 흥미 정도, 소프트웨어에 대한 태도(긍정적, 해 볼만한 것, 나와 무관)에 대해 조사하였다. 강의 초반에 실시된 설문에서 스크래치를 사용해 본 경험이 가장 높게 나타났으며 소프트웨어 교육에 대한 경험은 없지만 한 번은 배우고 싶다는 의견이 가장 높게 나타났다.

김완섭(2017a)은 비전공자 대상 소프트웨어 필수 교과 개설을 위한 연구에서 소프트웨어 교육의 필요 여부, 교양필수 도입에 대한 생각을 조사하였다. 약 60%의 학생이 소프트웨어 교육이 필요하다고 응답하였으며 활용 프로그램으로는 스크래치와 파이선이 적합하다고 하

였다. 후속연구(김완섭, 2017b)인 신입생 대상 소프트웨어 교육 인식 조사에서는 교양필수 과목으로 적합한지, 융합적 소양 함양에 도움이 되는지, 향후 취업과 진로에 도움 여부를 조사하였다. 교양필수로 적합하다고 응답한 학생은 60% 이상이었으며, 융합교육과 취업역량의 효과성에 대해서도 긍정적으로 인식하고 있었다.

나정은(2017a)은 교양교과로서의 소프트웨어 교육의 요구를 파악하기 위해 소프트웨어 과목 수강 의향, 예상되는 도움 정도, 교과 성격(공통기초, 교양필수, 교양선택)에 대해 조사하였다. 연구 참여자들은 65%가 수강할 의향이 있음을 밝혔고 공통기초 과목으로 개설되길 희망하였으며, 학년 및 전공과 무관하게 소프트웨어 교육의 필요성을 인식하고 있었다. 후속 연구에서는 개인적 중요도, 전공공부 도움정도, 커리어 도움 정도를 조사하였다(나정은, 2017b).

서주영 외(2017)는 인문대학생의 소프트웨어 기초 교육에 대한 인식을 조사하였다. 소프트 교육의 유용성, 이론/실습 교육의 적정성, 소프트웨어의 사회적 중요성에 대한 인식을 조사하였다. 학생들은 대학 입학 전 소프트웨어를 접해 본 경험이 거의 없고 친숙한 주제를 활용한 실습위주의 교육을 선호하였다. 오미자(2017)는 소프트웨어 교육에 대한 비전공자의 인식 연구에서 소프트웨어 교육 경험 여부, 흥미 여부를 조사하였다. 비전공자 학생의 74%가 소프트웨어 교육에 대한 경험이 없었고, 87%의 학생은 프로그래밍에 대한 어려움이 있었으며 약 70%의 학생이 소프트웨어 교육이 필요 없다고 응답하였다. 김주연(2018)은 미래사회에서 소프트웨어 교육의 필요 여부, 신입생 대상 교육 필요 여부를 조사하였다. 소프트웨어 교육의 필요성에서 긍정적 답변과 부정적 답변의 비율이 약 6:4로 나타났다.

둘째, 소프트웨어 교육의 어려움은 최정원, 이영준(2014), 김수환(2015), 오미자(2017)의 연구에서 살펴볼 수 있다. 최정원, 이영준(2014)은 학습자들이 어려워하는 개념과 원리에 대해 조사하였다. 프로그램의 함수 개념과 사용 방법에 대한 이해 부족, 문제해결 과정에서 포함되어야 하는 패턴을 인지하는 데 어려움이 있음을 확인하였다. 김수환(2015)은 컴퓨터 비전공자들의 어려움을 분석하였다. 비전공 학생들은 활용 프로그램의 변수활용과 프로그래밍 관련 개념을 가장 어려워하였고, 소프트웨어 교육에 대한 재미와 흥미가 컴퓨팅 사고력, 프로그래밍 능력과 유의미한 상관관계를 보여 학습자의 어려움을 감소시키는 요소로 활용할 수 있음을 확인하였다. 오미자(2017)는 소프트웨어 교육의 어려운 정도, 난이도에 대한 의견, 필요성 유무, 필요한 이유/필요하지 않은 이유를 조사하였다. 학생들은 처음 경험하는 프로그래밍이 생소했고 프로그래밍 문법이 어려웠으며 코딩 과정에서 발생하는 오류 수정에 어려움이 있다고 하였다.

셋째, 소프트웨어 교육의 효과성은 나정은(2017b), 서주영 외(2017), 오미자(2017), 김주연(2018)의 연구에서 살펴볼 수 있다. 나정은(2017b)은 컴퓨터 실력 향상 정도, 지식 및 스킬

향상 정도를 조사하였다. 소프트웨어 기초 과목을 수강한 연구 참여자들은 자신의 학업에 필요한 과목으로 수강하여, 인문사회계열 학생의 경우 이공계열 학생과의 성적 비교에서 뒤처지지 않았고 컴퓨터에 대한 지식 향상에서 인문사회계열 학생들이 더 많은 향상이 있었다고 하였다. 서주영 외(2017)는 향상된 학습능력, 프로그래밍 능력 향상 정도를 조사하였다. 향상된 능력은 컴퓨터 활용능력, 논리적 사고력 및 통찰력, 문제해결능력 순이었다.

오미자(2017)는 학습동기와 성취 향상 정도, 컴퓨터 언어에 흥미를 가지는 데 도움 정도, 컴퓨터에 대한 관심 및 이해 증가 도움 정도로 구성된 흥미영역, 문제해결력 향상, 논리적 사고 및 통찰력 향상 도움 정도, 자기주도적 학습 능력 향상 도움 정도로 구성된 사고력 영역, 새로운 지식이나 경험 습득 정도, 다른 교과나 사회에 도움 정도로 구성된 성취도 영역으로 나누어 조사하였다. 소프트웨어 교육에 대한 흥미 여부와 필요성 여부에 따라 흥미, 사고력, 성취도 향상에 차이가 있었다. 김주연(2018)은 논리적 사고력 향상에 도움 정도, 블렌디드 러닝 기반 교육의 효과에 대해 조사하였다. 소프트웨어 교육은 논리적 사고력 향상에 도움이 되었으며 블렌디드 러닝은 학업성취도와 학업만족도 측면에서 오프라인 수업과 거의 유사한 수준을 보였다.

<표 1> 선행연구에 대한 요구분석 주제별 분류

연구자	요구분석 주제		
	SW교육에 대한 인식	SW교육의 어려움	SW교육의 효과성
최정원,이영준 (2014)		학습자가 어려워하는 SW원리	
김수환(2015)		비전공자의 어려움	
성정숙 외 (2015)	•스크래치 흥미정도 •프로그래밍 능력정도 •추가수강여부		
이민석(2016)	•SW교육에 대한 배움 의지 •SW교육에 대한 흥미정도		
김완섭(2017a)	•SW교육필요여부 •교양필수 도입에 대한 생각		
김완섭(2017b)	•교양필수과목 적합여부 •융합적 소양 함양 도움 여부 •취업과 진로 도움 여부		
나정은(2017a)	•수강의향 •예상도움정도 •교과성격		

나정은(2017b)	•개인적 중요도 •전공 도움정도 •커리어 도움정도		•컴퓨팅 사고력 향상 정도 •지식 및 스킬 향상 정도
서주영 외 (2017)	•유용성 •이론/실습의 적정성 •사회적 중요성		•향상된 학습능력 •프로그래밍 능력 향상 정도
오미자(2017)	•경험여부 •흥미여부	•어려운 정도 •난이도에 대한 의견	•흥미향상 •사고력향상 •성취도향상
김주연(2018)	•미래사회 필요 여부 •신입생 대상 필요 여부		•논리적 사고력 향상 도움 정도 •블렌디드 러닝 효과

지금까지 살펴본 대학 소프트웨어 교육에 대한 요구분석에서, 소프트웨어 교육에 대한 인식은 배움 의지, 흥미, 교육의 필요성 인식, 도움 정도를 중심으로 구성되어 있고, 소프트웨어 교육의 어려움은 비전공자로서 교양교과로 수강하면서 경험한 어려운 점에 대한 것이다. 소프트웨어 교육의 효과성은 사고력, 학습능력, 성취도 향상을 중심으로 구성되어 있다. 대학 소프트웨어 교육이 시작된 시점의 선행연구는 소프트웨어 교육에 대한 인식에 중심을 두었다면, 최근의 연구는 효과성 연구에 중점을 두고 있다.

Ⅳ. 연구방법

1. 자료수집

A대학 소프트웨어 교육의 강의평가결과에 대한 네트워크 분석을 위해 학사팀으로부터 2017년 1학기와 2018년 1학기의 강의평가결과를 입수하였다. 2017년 1학기 수강생 1,271명 중 936명, 2018년 1학기 수강생 4,034명 중 2,449명이 답변한 서술형 문항의 답변을 분석대상으로 하였다.

2017년 1학기는 A대학의 소프트웨어교육이 교재의 수정·보완 후 첫 학기로, 교재의 변화 후 두 번째 학기인 2018년 1학기와 강의평가결과를 비교해 개선점을 도출하고자 하였다.

엑셀파일 형태의 강의평가 자료를 텍스트 파일로 변환하고, 네트워크 분석이 가능한 텍스트로 다듬기 위해 전처리 작업을 시행하였다. R 프로그램을 활용해 한 가지 뜻이지만 여러 단어로 표현된 단어를 한 개의 단어로 통일하고 오타를 수정하였다. 단어 통일의 예를 살펴보면 모둠활동, 팀활동, 조별활동으로 표현된 단어들을 조별활동으로 통일하였다.

2. 분석방법

A대학의 강의평가는 객관식 문항과 서술형 문항으로 구성되어 있다. 객관식 문항은 강의계획서의 체계적 구성, 수업내용, 교재 및 수업자료, 교수법, 학생과의 상호작용, 수업관리, 평가, 지식과 관심 향상 정도, 강의에 대한 전반적 만족도로 구성되어 있다. 서술형 문항은 강의의 좋았던 점과 개선해야 할 점으로 구성되어 있다.

2017년 1학기와 2018년 1학기의 두 학기에 대해, 소프트웨어 교육의 강의평가결과를 전체 강의평가결과와 비교해 보면, 다음 <표 2>와 같다. 2017년 1학기와 2018년 1학기 전체 강좌의 강의평가 평점을 비교하면 거의 비슷한 분포를 보이고 있다. 반면, 소프트웨어 교육의 평점은 2017년 1학기보다 2018년 1학기의 평점 평균이 전체 항목에서 증가세를 보였지만, 여전히 전체 강좌 강의평가 평점보다는 낮은 분포를 보이고 있다. 2017년 1학기에 진행된 소프트웨어 교육은 디자인 싱킹 기반의 창의적 사고를 중심으로 시행하였다. 두 번째 시행인 2018년 1학기는 활용 프로그램을 교체하고 창의적 문제해결력을 강화하고자 디자인 싱킹에 대한 내용을 보강하였으며, 공통으로 활용하던 교재는 각 전공 특성을 반영한 교재를 개발하여 활용하고 있다. 그러나 이러한 변화에도 불구하고 소프트웨어 교육의 평균 평점은 전체 강좌와 비교해 강의평가 전체 항목에서 모두 낮은 분포를 보여, 그 원인을 찾고자 텍스트의 내용 분석이 가능한 네트워크 분석을 활용하였다.

<표 2> 2017년 1학기와 2018년 1학기 강의평가 평균 평점

강의평가 내용	2017-1		2018-1	
	전체강좌	SW교육	전체강좌	SW교육
1. 수업내용과 수업목표와의 부합 정도	4.15	3.87	4.15	3.92
2. 교재 및 수업자료의 도움 정도	4.11	3.76	4.12	3.87
3. 교수자는 학습자의 수준 고려하여 수업 진행	4.09	3.70	4.09	3.87
4. 수업에 적절한 교수법 활용	4.12	3.78	4.12	3.89
5. 학생과의 상호작용(질문 및 답변, 피드백, 상담)	4.14	3.86	4.14	3.92
6. 수업과 관련된 지식, 관심 향상	4.08	3.67	4.10	3.81
7. 전반적인 만족	4.09	3.70	4.10	3.83
전체 평균	4.11	3.76	4.12	3.87

강의평가 서술형 문항에 대해 한국어 텍스트 분석 프로그램인 KrKwic을 활용하였다. 전체 텍스트에 대해 단어 출현빈도를 계산하고, 출현빈도가 높은 단어를 중심으로 핵심단어를 도출하였다. 핵심단어를 활용해 KrTitle 프로그램에서 공출현빈도를 도출하였다(박한우,

Loet Leydesdorff, 2004). 공출현빈도는 한 문장에 함께 출현한 핵심단어의 빈도로 자료는 메트릭스 형태이다. 이러한 과정에서 산출된 핵심단어 출현빈도와 공출현빈도를 활용해 UCINET 프로그램에서 핵심단어 간 네트워크를 형성하였다. 핵심단어 간 네트워크 관계를 수치로 살펴볼 수 있는, 텍스트 중심성 분석은 연결중심성, 근접중심성, 매개중심성을 시행하였다. 연결중심성은 한 노드가 다른 노드와 얼마나 연결되어 있는지, 근접중심성은 노드와 노드의 인접 정도, 매개중심성은 노드와 노드를 연결하는 중간에 위치한 중간 노드가 다른 노드들과 얼마나 연결되어 있는가를 살펴볼 수 있다.

2017년 1학기와 2018년 1학기의 두 학기 간 소프트웨어 교육의 좋은 점과 개선점을 비교・분석하기 위해, 좋은 점과 개선점이 각각 결합된 매트릭스 형태의 자료를 활용해 네트워크 분석을 실시하였다.

V. 연구결과

1. 핵심단어 출현빈도

각 학기별 강의평가결과를 좋은 점, 개선점으로 각각 나누고 출현단어의 빈도를 산출하였다. 출현빈도가 높은 단어를 중심으로, 출현빈도가 5회 내외인 단어, 각 학기의 텍스트 분량을 감안하여 핵심단어를 선정하였다. 또한 강의, 수업, 교수님, 학생, 좋다 등과 같이 공통적으로 등장하는 단어를 제외하였다. <표 3>과 같이, 2017년 1학기의 좋은 점은 코딩, 친절, 재미, 조별(활동), 설명 등의 순으로, 개선점은 어렵다, 코딩, 생각, 컴퓨터, 힘들다 등의 순으로 나타났다. 2018년 1학기의 좋은 점은 코딩, 친절, 재미, 설명, 컴퓨터 등의 순으로, 개선점은 과제, 조별(활동), 코딩, 어렵다, 시험 등의 순으로 나타났다. 소프트웨어 교육의 좋은 점에서는 친절, 재미, 새롭다, 열정, 도움, 소통, 창의 등과 같은 긍정적 단어들이 도출되었고, 개선점에서는 어렵다, 힘들다, 아쉽다, 부족, 부담, 불편 등의 단어들이 등장해 소프트웨어 교육에 대한 문제점과 개선사항에 대한 의견을 살펴볼 수 있었다.

각 학기별 강의평가결과의 좋은 점과 개선점에서 도출한 핵심단어는 학생, 교수님, 강의와 같이 일반 교과 강의평가에서도 공통적으로 도출되는 단어들을 제외하고, 소프트웨어 교육의 특징을 살펴볼 수 있는 단어들을 중심으로 핵심단어를 도출하였다.

<표 3> 강의평가결과의 좋은 점과 개선점의 핵심단어 출현빈도

2017년 1학기				2018년 1학기			
좋은 점		개선점		좋은 점		개선점	
핵심 단어	출현 빈도	핵심 단어	출현 빈도	핵심 단어	출현 빈도	핵심 단어	출현 빈도
코딩	154	어렵다	103	코딩	166	과제	118
친절	64	코딩	103	친절	106	조별(활동)	107
재미	58	생각	57	재미	93	코딩	76
조별(활동)	54	컴퓨터	52	설명	75	어렵다	72
설명	50	힘들다	45	컴퓨터	67	시험	62
새롭다	46	이해	33	조별(활동)	66	컴퓨터	47
컴퓨터	40	과제	32	쉽다	50	생각	39
실습	31	필요(성)	31	새롭다	48	이해	35
쉽다	31	조별(활동)	31	열정	47	설명	33
어렵다	30	시험	30	스크래치	45	발표	32
이해	27	프로그램	30	이해	44	스크래치	22
열정	24	실습	28	관심	40	필요(성)	22
도움	23	설명	27	창의	37	조원(평가)	20
소통	19	비전공자	24	실습	34	프로그램	17
어플리케이션	17	아쉽다	14	과제	34	이론	15
과제	17	부족	14	도움	29	참여	14
경험	15	흥미(유발)	14	사고	28	진도	14
피드백	14	창의(력)	13	프로그램	27	보강	13
활용	13	도움	11	소통	25	평가	13
창의	12	발표	11	흥미	23	흥미	11
비전공자	11	피피티	11	질문	23	아쉽다	10
편안(한)분위기	11	조원(팀원)	10	피드백	21	집중	9
발표	11	(학생수준)고려	10	자유	19	피드백	9
자율	11	SAP	10	배려	17	불편	9
흥미	11	집중(집중력)	9	편안(한분위기	16	실습	8
눈높이	10	난이도	8	발표	15	질문	8
유익	9	이러닝	6	경험	12	난이도	7
플립드러닝	6	피드백	6	눈높이	10	교재	6
배려	6	플립드러닝	6	답변	10	부담	6
질문	6	참여(도)	5	피피티	9	이러닝	6

2. 네트워크 분석

네트워크 분석에 앞서 핵심단어 출현빈도를 활용해 공출현빈도를 도출하였다. 공출현빈도는 강의평가 각 문장에 출현빈도가 높은 핵심단어가 동시에 나타나는 빈도이다. [그림 1]과 [그림 2]는 각각 2017년 1학기와 2018년 1학기 강의평가 결과에 대한 좋은 점과 개선점의 공출현빈도로, 전체 30개 핵심단어의 중 각각 10개 핵심 단어의 공출현빈도를 제시하였다.

좋은점	코딩	친절	재미	조별	설명	새롭다	컴퓨터	실습	쉽다	어렵다
코딩	0	5	11	1	4	9	3	9	8	8
친절	5	0	3	2	14	0	2	2	2	1
재미	11	3	0	5	3	4	4	5	3	1
조별	1	2	5	0	3	0	0	0	1	1
설명	4	14	3	3	0	0	1	1	13	1
새롭다	9	0	4	0	0	0	0	0	1	0
컴퓨터	3	2	4	0	1	0	0	4	0	1
실습	9	2	5	0	1	0	4	0	1	0
쉽다	8	2	3	1	13	1	0	1	0	1
어렵다	8	1	1	1	1	0	1	0	1	0

개선점	어렵다	코딩	생각	컴퓨터	힘들다	이해	과제	필요	조별	시험
어렵다	0	21	3	3	2	8	2	4	0	0
코딩	21	0	30	12	20	17	9	11	7	3
생각	3	30	0	15	4	10	17	6	4	13
컴퓨터	3	12	15	0	6	0	1	5	0	2
힘들다	2	20	4	6	0	5	8	1	9	2
이해	8	17	10	0	5	0	7	1	2	0
과제	2	9	17	1	8	7	0	3	20	14
필요	4	11	6	5	1	1	3	0	1	0
조별	0	7	4	0	9	2	20	1	0	4
시험	0	3	13	2	2	0	14	0	4	0

〔그림 1〕 2017년 1학기 좋은 점(왼쪽), 개선점(오른쪽)의 공출현빈도

좋은점	코딩	친절	재미	설명	컴퓨터	조별	쉽다	새롭다	열정	스크래치
코딩	0	6	12	10	7	11	0	7	3	2
친절	6	0	3	15	3	3	0	1	1	5
재미	12	3	0	1	2	4	0	2	0	11
설명	10	15	1	0	1	1	2	0	1	0
컴퓨터	7	3	2	1	0	2	0	3	1	1
조별	11	3	4	1	2	0	0	1	2	2
쉽다	0	0	0	2	0	0	0	0	0	0
새롭다	7	1	2	0	3	1	0	0	0	2
열정	3	1	0	1	1	2	0	0	0	1
스크래치	2	5	11	0	1	2	0	2	1	0

개선점	과제	조별	코딩	어렵다	시험	컴퓨터	생각	이해	설명	발표
과제	0	115	8	3	25	2	44	7	10	20
조별	115	0	7	1	11	0	35	2	0	14
코딩	8	7	0	0	5	9	7	6	2	8
어렵다	3	1	0	0	3	1	0	3	2	0
시험	25	11	5	3	0	0	3	0	1	10
컴퓨터	2	0	9	1	0	0	3	0	2	0
생각	44	35	7	0	3	3	0	1	2	8
이해	7	2	6	3	0	0	1	0	7	5
설명	10	0	2	2	1	2	2	7	0	2
발표	20	14	8	0	10	0	8	5	2	0

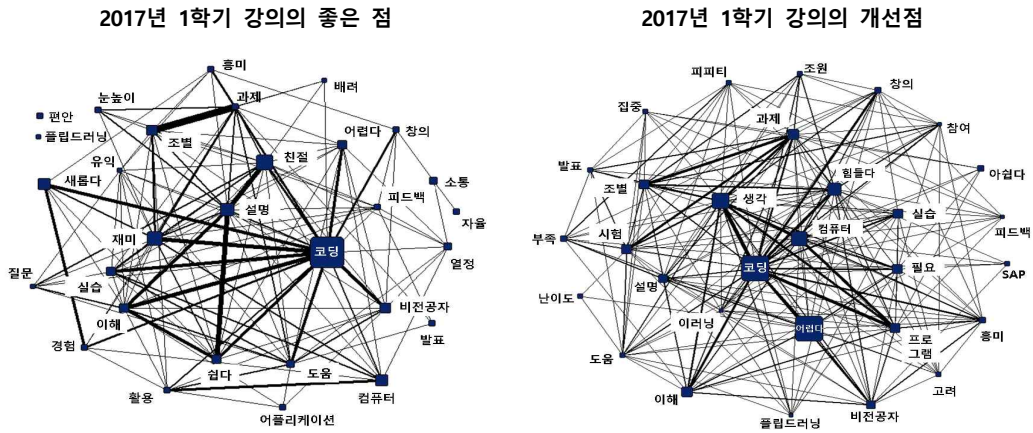
〔그림 2〕 2018년 1학기 좋은 점(왼쪽), 개선점(오른쪽)의 공출현빈도

이렇게 산출된 공출현빈도와 핵심단어 출현빈도를 활용해 네트워크 분석을 실시하였다. [그림 3]과 [그림 4]는 2017년 1학기와 2018년 1학기 강의의 좋은 점, 개선점에 대한 네트워크 분석이다. 2017년 1학기 강의의 좋은 점 네트워크는 ‘코딩’이 가장 많이 언급된 핵심단어이고, ‘조별(활동)’과 ‘과제’가 함께 언급되는 문장이 가장 많은 것으로 나타났다. 조별(활동)은 재미, 설명, 눈높이, 코딩, 흥미 등과 함께 언급되고 있으며 과제는 흥미, 코딩, 재미, 배려, 설명, 도움, 눈높이와 함께 언급되고 있다. 2017년 1학기 강의의 개선점 네트워크는 ‘어렵다’가 가장 많이 언급된 핵심단어이고, ‘코딩’과 ‘어렵다’가 함께 언급되는 문장이 가장 많은 것으로 나타났다. 코딩은 어렵다, 생각, 프로그램, 실습, 이해 등과 함께 언급되고 있으며 어렵다는 코딩, 비전공자, 이해, 부족 등과 함께 언급되고 있다.

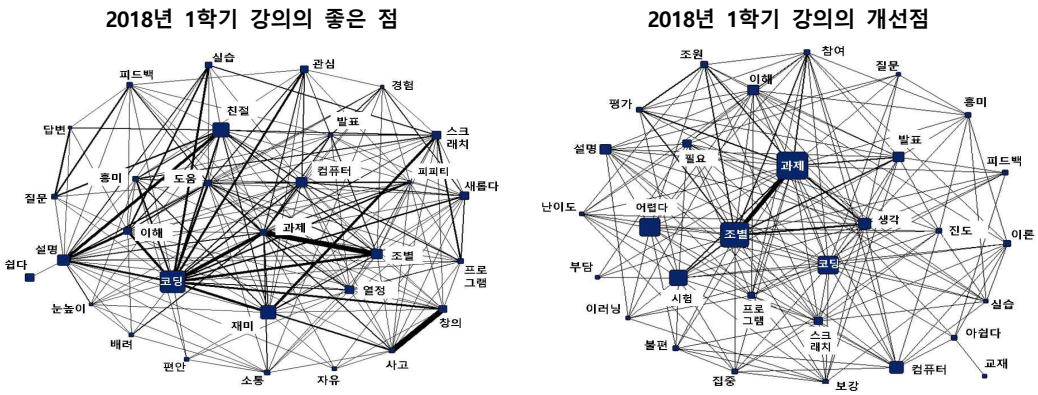
2018년 1학기 강의의 좋은 점 네트워크는 ‘코딩’이 가장 많이 언급된 핵심단어이고,

'조별(활동)' 과 '과제' 가 함께 언급되는 문장이 가장 많은 것으로 나타났다. 조별(활동)은 과제, 코딩, 재미 등과 함께 언급되고 있으며 과제는 조별(활동), 코딩, 재미, 컴퓨터, 도움 등과 함께 언급되고 있다. 2018년 1학기 강의의 개선점 네트워크는 '과제' 가 가장 많이 언급된 핵심단어이고, '조별(활동)' 과 '과제' 가 함께 언급되는 문장이 가장 많은 것으로 나타났다. 조별(활동)은 과제, 생각, 시험, 평가, 조원 등과 함께 언급되고 있으며 '과제' 는 조별(활동), 생각, 발표, 평가, 조원, 참여 등과 함께 언급되고 있다.

두 학기에 걸친 소프트웨어 교육의 강의평가결과에서 '조별(활동)' 과 '과제' 가 함께 언급되는 문장이 가장 많은 것으로 나타났다. 소프트웨어 교육의 좋은 점과 개선점에서 공통적으로 조별(활동)과 과제가 함께 언급되는 빈도가 높은 것은 조별활동에 대한 학생들의 관점 차이가 큼을 예상할 수 있다.



〔그림 3〕 2017년 1학기 강의평가결과에 대한 네트워크 분석



〔그림 4〕 2018년 1학기 강의평가결과에 대한 네트워크 분석

지금까지 살펴본 네트워크 분석을 수치화된 텍스트 중심성으로 살펴보면, <표 4>와 같이, 2017년 1학기 강의의 좋은 점에서 연결중심성은 코딩, 설명, 재미, 과제, 친절, 이해의 순으로, 근접중심성은 코딩, 친절, 설명, 도움, 이해의 순, 매개중심성은 코딩, 친절, 열정, 소통, 도움의 순으로 나타났다. 2017년 1학기 강의의 개선점에서 연결중심성은 코딩, 생각, 과제, 프로그램, 힘들다의 순으로, 근접중심성은 코딩, 생각, 힘들다, 과제, 컴퓨터의 순, 매개중심성은 코딩, 힘들다, 생각, 과제, 컴퓨터의 순으로 나타났다.

<표 5>와 같이, 2018년 1학기 강의의 좋은 점에서 연결중심성은 코딩, 과제, 조별(활동), 설명, 도움 등의 순으로, 근접중심성은 코딩, 과제, 도움, 설명, 컴퓨터 등의 순으로, 매개중심성은 설명, 코딩, 과제, 도움, 친절 등의 순으로 나타났다. 2018년 1학기 강의의 개선점에서 연결중심성은 과제, 조별(활동), 생각, 코딩, 발표 등의 순으로, 근접중심성은 과제, 코딩, 조별(활동), 생각, 시험 등의 순, 매개중심성은 과제, 코딩, 아쉽다, 조별(활동), 생각 등의 순으로 나타났다.

<표 4> 2017년 1학기 강의의 좋은 점, 개선점에 대한 텍스트 중심성

핵심단어	좋은 점			핵심단어	개선점		
	연결	근접	매개		연결	근접	매개
코딩	112	94	50.644	어렵다	64	42	12.598
친절	49	97	34.365	코딩	245	32	28.076
재미	51	102	12.322	생각	176	32	23.264
조별(활동)	44	103	11.740	컴퓨터	91	37	17.664
설명	65	100	13.957	힘들다	98	32	26.994
새롭다	24	112	1.600	이해	79	43	2.337
컴퓨터	25	107	5.023	과제	119	35	19.023
실습	31	105	3.643	필요(성)	49	39	12.552
쉽다	42	106	3.293	조별(활동)	71	41	7.076
어렵다	20	104	4.316	시험	70	41	15.674
이해	48	102	15.225	프로그램	105	43	4.322
열정	8	106	33.365	실습	38	44	3.025
도움	34	101	22.088	설명	46	41	5.427
소통	3	127	26.250	비전공자	58	43	3.682
어플리케이션	5	116	0.000	아쉽다	9	53	0.458
과제	49	104	16.142	부족	23	46	3.490
경험	18	111	1.640	흥미(유발)	51	43	2.350
피드백	12	105	14.431	창의(력)	40	45	4.669
활용	22	108	6.547	도움	34	41	13.388
창의	9	110	16.872	발표	29	49	0.893
비전공자	22	104	12.320	피피티	25	47	1.972
편안	0	0	0.000	조원(팀원)	29	49	0.378
발표	4	116	0.333	(학생수준)고려	14	47	1.311

핵심단어	좋은 점			핵심단어	개선점		
	연결	근접	매개		연결	근접	매개
자율	1	153	0.000	SAP	5	54	0.000
흥미	11	109	5.301	집중(집중력)	13	49	1.937
눈높이	10	114	0.000	난이도	4	59	0.100
유익	17	106	3.563	이러닝	34	41	4.277
플립드러닝	0	0	0.000	피드백	11	49	0.911
배려	5	114	0.000	플립드러닝	15	49	0.617
질문	15	110	0.000	참여(도)	31	46	2.535

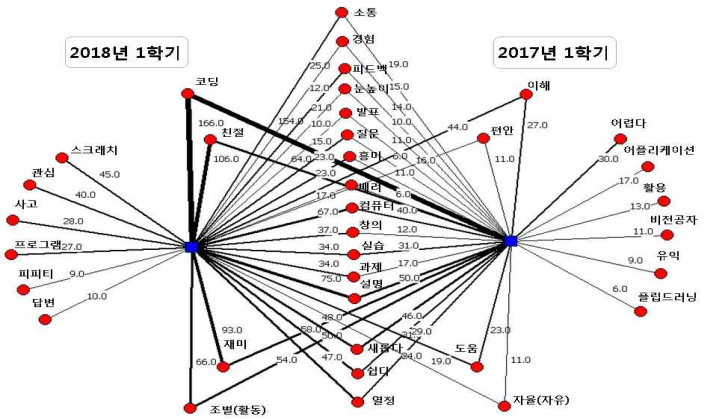
<표 5> 2018년 1학기 강의의 좋은 점, 개선점에 대한 텍스트 중심성

핵심단어	좋은 점			핵심단어	개선점		
	연결	근접	매개		연결	근접	매개
코딩	163	33	30.056	과제	313	34	51.286
친절	67	39	15.563	조별(활동)	246	39	20.891
재미	54	39	13.371	코딩	107	35	38.464
설명	68	36	54.961	어렵다	21	47	5.896
컴퓨터	50	37	9.900	시험	83	42	16.656
조별(활동)	72	41	4.500	컴퓨터	38	44	12.364
쉽다	2	64	0.000	생각	130	39	16.806
새롭다	31	44	4.578	이해	48	43	10.741
열정	22	42	7.950	설명	40	46	5.110
스크래치	49	42	4.483	발표	92	45	7.251
이해	52	41	14.526	스크래치	35	43	8.596
관심	32	48	1.265	필요(성)	30	44	9.007
창의	52	46	2.178	조원(평가)	70	49	1.748
실습	28	50	0.237	프로그램	32	45	6.901
과제	106	34	21.793	이론	22	48	7.119
도움	68	35	16.517	참여	69	44	7.773
사고	56	44	3.842	진도	20	48	3.827
프로그램	30	45	1.188	보강	34	48	4.183
소통	16	47	6.051	평가	56	51	0.316
흥미	27	44	6.327	흥미	17	50	2.677
질문	26	51	0.467	아쉽다	11	52	28.811
피드백	23	47	4.507	집중	26	48	2.440
자유	7	52	0.606	피드백	22	51	2.549
배려	9	52	0.383	불편	11	51	0.943
편안(한) 분위기	5	55	0.604	실습	13	49	1.921
발표	28	45	2.350	질문	10	53	0.486
경험	21	45	2.474	난이도	15	49	1.461
눈높이	12	49	1.470	교재	1	80	0.000
답변	13	54	0.000	부담	9	55	0.000
피피티	33	41	3.854	이러닝	11	52	0.777

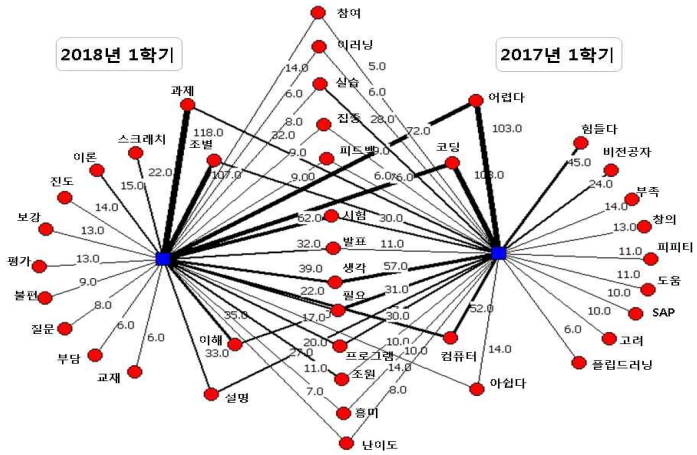
3. 네트워크 비교 분석

네트워크 비교를 살펴보면, [그림 5]와 같이 2017년 1학기와 2018년 1학기 강의의 좋은 점에 대한 비교 네트워크에서 공통적으로 코딩, 친절, 이해, 편안(한) 분위기, 재미 등과 같은 핵심단어가 언급되었다. 2018년 1학기의 좋은 점에서는 스크래치, 관심, 사고, 프로그램, 피피티, 답변의 단어들이 새롭게 도출되었다.

[그림 6]과 같이 강의의 개선점에 대한 비교 네트워크에서 공통적으로, 과제, 조별(활동), 어렵다, 코딩, 아쉽다 등의 단어가 언급되었다. 2018년 1학기 강의의 개선점에서는 스크래치, 이론, 진도, 보강, 평가, 불편, 질문, 부담, 교재의 단어들이 새롭게 도출되었다.



〔그림 5〕 2017년 1학기와 2018년 1학기 좋은 점 비교 네트워크



〔그림 6〕 2017년 1학기와 2018년 1학기 개선점 비교 네트워크

4. 서술형 강의평가 내용

2018년 1학기의 소프트웨어 교육은 활용 소프트웨어의 교체, 각 전공 특성을 반영한 교재 활용 등으로 변화를 주었지만 여전히 공통적으로 확인된 개선 요구사항과 새롭게 등장한 개선 요구사항이 있다.

새롭게 등장한 개선 요구사항을 서술형 강의평가결과에서 살펴보면 다음과 같다. 학생들은 수업이 흥미롭지 않다고 생각했고, 수업시간에 실습한 내용보다 과다한 과제에 대해 개선을 요구하였다. 또한, 스크래치를 처음 접해 수업내용이 이해되지 않았고, 익숙하지 않은 내용으로 인해 어려웠으며, 진도가 빠르다고 생각했다. 혼자 해결하기 어렵고 버거운 과제가 부여되고 있어 과제 분량을 줄여주었으면 한다는 의견도 있었다. 팀 단위 활동임에도 불구하고 일부 학생만 참여하여 무임승차 학생이 존재하므로, 전체 학생이 참여할 수 있는 학습 환경 조성이 필요하다고 했다. 교수자에 대해서는 발표에 대해 비판보다는 피드백을 원하였고 공정한 평가를 요구하였다. 컴퓨터 실습을 위해 컴퓨터가 책상에 비치된 상태에서 수업이 진행되어, 컴퓨터 실습 외 소셜 네트워크 서비스(Social Network Services)를 이용하거나 다른 작업을 할 수 있어 수업 집중의 어려움이 있고, 수업 보강 시 정확한 공지가 필요하다고 하였다.

강의의 좋은 점에 대해, 학생들은 부담스럽지 않은 과제와 수업방식에 만족하였고 질문과 발표에 대한 피드백이 있어 수업내용에 대한 이해가 쉽다고 하였다. 새롭고 재미있는 스크래치를 활용한 실습위주의 수업에 대해서는 흥미를 가질 수 있게 해주는 교수자가 있고, 디자인 싱킹을 기반으로 한 창의적 사고 프로그램이 생각을 많이 할 수 있게 했다고 답하였다. 또한, 코딩 실습과 과제 해결에 교수자의 지도가 도움이 되었다고 하였다.

Ⅵ. 결론

A대학은 2016년 1학기부터 소프트웨어 교육을 시작하여, 2017년에는 공통 교재 개발 및 활용, 활용 프로그램 지정, 플립러닝과 팀기반 학습을 기반으로 실시하였다. 2018년에는 공통교재를 각 계열별 특성을 반영한 5개의 개별 교재로 개발하여 대체하고, 활용 프로그램도 사용이 쉬운 스크래치 프로그램을 활용하기 시작하였으며 플립러닝과 팀기반 학습은 동일하게 적용하였다. 이런 변화에도 불구하고 소프트웨어 교육의 강의만족도는 강의평가 전체 항목에서 전체 강의평가보다 낮은 분포를 보여, 그 원인을 찾고자 2017년 2학기와 2018년 1학기 서술형 강의평가 내용에 대한 네트워크 분석을 실시하였다.

네트워크 분석 결과는 다음과 같다. 2017년 1학기와 2018년 1학기의 소프트웨어 교육의 강의평가결과에서 ‘조별(활동)’과 ‘과제’가 함께 언급되는 경우가 많음을 확인할 수 있었고 서술형 강의평가결과에서 디자인 싱킹 기반의 창의적 사고 프로그램에 의해 생각을 많이 하게 되었다는 수강생들의 답변을 다수 확인할 수 있었다. 개선점에서도 ‘조별(활동)’과 ‘과제’의 관계가 두드러지는데, 과다한 조별활동과 과제로 인해 힘들다는 학생들의 의견을 확인할 수 있었고, 학생의 의견을 반영하지 않고 배려하지 않는 수업 진행을 확인할 수 있었다.

2017년 1학기와 2018년 1학기에 대해 소프트웨어 교육의 좋은 점과 개선점의 비교 네트워크를 살펴보면, 좋은 점에서는 공통적으로 코딩, 친절, 이해, 편안(한) 분위기, 재미 등이 도출되었고, 2018년 1학기에는 스크래치, 관심, 사고, 프로그램, 피피티, 답변의 단어들이 새롭게 도출되었다. 2018년 1학기 강의평가결과에서 새롭게 등장한, 핵심단어를 포함하고 있는 서술형 강의평가결과를 살펴보면, 부담스럽지 않은 과제와 수업방식, 질문과 발표에 대한 교수자의 피드백, 흥미로운 실습 위주의 수업, 창의적 사고를 할 수 있는 강의, 실습과 과제에 대한 교수자의 도움을 확인할 수 있었다. 개선점에서는 공통적으로 과제, 조별(활동), 어렵다, 코딩, 아쉽다 등의 단어가 도출되었고, 2018년 1학기에는 스크래치, 이론, 진도, 보강, 평가, 불편, 질문, 교재의 단어들이 새롭게 도출되었다. 2018년 1학기 강의평가결과에서 새롭게 등장한 핵심 단어를 포함하고 있는 서술형 강의평가결과를 살펴보면, 흥미롭지 않은 수업, 과다한 과제, 익숙하지 않고 어려운 스크래치 프로그램, 빠른 진도, 해결하기 어려운 개별 과제, 조별활동에서 전체 학생이 참여할 수 있는 학습 환경 조성, 교수자의 피드백 필요, 공정한 평가의 내용을 확인할 수 있었다.

지금까지 살펴본 분석결과를 종합해보면, 소프트웨어 교육은 창의적 사고 향상 프로그램을 기반으로 실습위주의 코딩을 조별활동 중심으로 운영·유지하면서, 소프트웨어 교육 필요성에 대한 동기부여 강화, 적절한 분량과 난이도의 과제 부여, 스크래치 활용법에 대한 학습 강화, 적극적인 조별활동 참여 문화 조성, 학습결과에 대한 피드백 제공이 요구된다.

이상의 연구결과를 바탕으로 한 제언은 다음과 같다. 디자인 싱킹을 바탕으로 학생 간 협업이 가능하도록 조별활동으로 운영하는 대학의 소프트웨어교육은 문제해결력과 의사소통능력, 협업능력 등의 향상을 기대할 수 있다. 대학의 소프트웨어교육은 1학년 대상으로 교양필수 교과목으로 지정하여 운영하면서 비이공계열 학생도 소프트웨어교육에 참여하고 있다. 이러한 비이공계열 학생들은 동기부여가 반드시 선행되어야 하고, 수강생의 학습 능력을 감안하여 활용 소프트웨어를 선정하여야 한다. 또한 활용 소프트웨어의 사용법에 대한 자세한 설명, 과제와 실습결과에 대한 피드백이 제공되어 학생들이 학습내용에 대한 흥미를 유지하고 포기하지 않도록 독려하는 교수자의 배려가 필요하다.

후속연구에서는 사회, 인문, 공학, 자연, 예·체능의 5개 계열별 교재를 개발하여 활용하고 있는 소프트웨어 교육의 강의평가결과에 대해, 계열별 네트워크 분석, 교수와 학생 인터뷰 실시를 통한 계열별 소프트웨어 교육 개선안을 제시하고자 한다.

참고문헌

국가과학기술자문회의(2017). **4차 산업혁명에 대비한 SW 융합인재 양성 방안** (정책연구-2017-01).

김수환(2015). Computational Thinking 교육에서 나타난 컴퓨터 비전공 학습자들의 어려움 분석, **한국컴퓨터교육학회논문지, 18**(3), 49-57.

김완섭(2017a). 비전공자를 위한 소프트웨어 필수교과 개설을 위한 연구. **한국교양교육학회 학술대회 자료집**, **24**(4), 110-115.

김완섭(2017b). 교양필수로서의 컴퓨팅적사고 과목에 대한 신입생들의 인식 연구-숭실대학교 2017-1학기 수강자 설문조사를 중심으로. **문화와 융합**, **39**(6), 141-170.

김용학, 김영진(2016). **사회 연결망 분석**. 서울: 박영사.

김주연(2018). SW 코딩교육에서의 블렌디드 러닝의 학습효과 분석. **융복합지식학회논문지, 6**(1), 1-7.

나정은(2017a). 교양교육으로서의 소프트웨어 교육 니즈 분석. **교양교육연구**, **11**(3), 63-89.

나정은(2017b). 학습자 관찰을 통한 컴퓨팅적 사고 학습효과 분석. **교양교육연구**, **11**(5), 349-378.

대한민국교육부(2015,7,24). 초등학교에서 대학까지, 소프트웨어(SW) 교육 청사진 나왔다! [인터넷 블로그]. https://if-blog.tistory.com/5341에서 검색

민혜리, 윤한솔(2017). 강의평가 주관식 응답에 나타난 교수와 학생의 의견 차이 비교 분석: 강의의 장·단점 의견에 대한 네트워크 및 내용분석. **학습자중심교과교육연구**, **17**(11), 307-320.

박금주, 최영준(2018). 비전공자를 위한 소프트웨어 교육 방향의 탐색. **교육문화연구**, **24**(3), 273-292.

박한우, Loet Leydesdorff(2004). 한국어의 내용분석을 위한 KrKwic 프로그램의 이해와 적용: Daum.net에서 제공된 지역혁신에 관한 뉴스를 대상으로. **한국자료분석학회**, **6**(5), 1377-1387.

서응교(2017). 플립러닝과 디자인씽킹에 기반을 둔 창의적사고 강화와 코딩교육을 위한 강좌 개발. **학습자중심교과교육연구**, **17**(16), 173-199.

서주영, 신승훈, 구은희(2017). 소프트웨어 기초 교육에서 수업 방식에 의한 인문대학생의 디지털 마인드 변화 분석. **디지털융복합연구**, **15**(9), 55-64.

송연옥(2017). 대학에서의 성찰기반 코딩교육 수업설계와 적용: H대학교 'SW와컴퓨팅사고' 수업사례를 중심으로. **교육공학연구**, **33**(3), 709-736.

성정숙, 김수환, 김현철(2015). 인문계열 학생을 위한 SW교육에서의 초보 학습자 특성 분석.

한국컴퓨터교육학회 논문지, **18**(3), 25-35.

신희성, 서응교, 오경선, 정혜진, 박소현(2018). 비전공자 SW기초 교육을 위한 디자인씽킹 기반의 SW교육 프로그램 개발 사례 연구. **한국컴퓨터교육학회**, **22**(2). 125-128.

오미자(2017). 스크래치 프로그램을 활용한 프로그래밍 교육에 대한 비전공자의 인식 연구. **컴퓨터교육학회논문지**, **20**(1), 1-11.

유홍준(2015). 성균관대학교 소프트웨어 교양교육 실시 방안. **한국교양교육학회 학술대회 자료집**, **2015**(11), 157-160.

이민석(2016). 비이공계 소프트웨어 교육의 도전 과제. **한국교양교육학회 학술대회 자료집**, **2016**(11), 135-140.

이재호(2017). **언플러그드 SW코딩 교육을 위한 생활 속 SW 코딩의 발견 1: 신호등 SW프로그램의 발견**, 파주: 정일.

최성경(2019). **컴퓨팅 사고 중심의 멀티플 프로젝트 기반 코딩교육(CT-MPB) 모형 개발**. 박사학위논문, 경희대학교.

최정원, 이영준(2014). 프로그래밍 학습에서 학습자의 어려움 분석. **컴퓨터교육학회논문지, 17**(5), 89-98.

한옥영, 김재현(2017). 비전공자 소프트웨어 교육을 통한 컴퓨팅 사고력 향상에 대한 연구. **한국컴퓨터교육학회 동계 학술발표 논문지**, **21**(1), 139-141.

홍정민(2017). **4차 산업혁명 시대의 미래 교육 에듀테크.** 서울: 책밥.

김지선(2018.6.10). SW정책연구소, SW국가경쟁력지수 개발, 韓 SW경쟁력 객관화한다. 전자신문 CIOBIZ.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society*, *366*, 3717-3725.