The Journal of the Convergence on Culture Technology (JCCT)
Vol. 11, No. 1, pp.651-658, January 31, 2025. pISSN 2384-0358, eISSN 2384-0366

http://dx.doi.org/10.17703/JCCT.2025.11.1.651

JCCT 2025-1-68

생성형 언어모델을 활용하는 코딩수업에서의 학습평가

Assessment Approaches in Coding Class Supported by Generative Language Models

김기백*

Gibak Kim*

요 약 본 논문은 생성형 언어모델이 코딩 교육에서 미치는 영향을 분석하고, 이에 적합한 학습 평가 방식에 대해 고찰하고자 한다. 생성형 언어모델은 자연어로 주어진 지시문을 바탕으로 정확하고 효율적인 코드를 생성할 수 있는 능력을 가지고 있어 코딩 학습 및 교육의 도구로의 활용도가 증대되고 있다. 이러한 환경 변화에 따라 코딩 수업에서 학생들의 성취도 평가를 위한 코딩 시험에서도 생성형 언어모델의 활용을 허용하는 것이 피할 수 없는 상황으로 여겨지고 있다. 그러나 언어모델을 활용한 코딩 시험에서 학생들이 지시문을 단순히 프롬프트로 입력하여 높은 점수를 받는 경우가 발생하면서, 기존 평가 방식만으로는 학습자의 성취도를 변별하기 어려워지고 있다. 우리는 프로그래밍 문법에 대한 이해도와 생성형 언어모델을 활용하여 문제를 해결하는 능력 간의 상관관계를 분석하고자 하며 S대학의 코딩수업에서 기본적인 프로그래밍 문법을 평가하는 시험과 생성형 언어모델을 활용한 코딩시험 결과를 통해 두 가지 능력의 상관관계를 분석하였다. 이러한 결과를 바탕으로 학습자의 프로그래밍 문법 이해도와 생성형 언어모델을 활용하여 문제를 해결하는 능력을 종합적으로 평가하기 위한 새로운 평가 문항 개발의 필요성을 논의한다.

주요어: 생성형 언어모델, 코딩 교육, 평가 방법, 프롬프트 엔지니어링

Abstract This paper aims to analyze the impact of generative language models on coding education and explore appropriate assessment methods. Generative language models have the ability to generate accurate and efficient code based on natural language prompts, making them increasingly utilized as tools in coding education. However, in coding exams that allow the use of language models, students may simply input the exam instructions as prompts and still achieve high scores. As a result, traditional assessment methods have become inadequate in distinguishing students' actual coding proficiency. We aim to analyze the correlation between understanding programming syntax and the ability to solve problems using generative language models. To do this, we examined the results of both a test assessing basic programming syntax and a coding test that involved utilizing generative language models, administered in a coding course at S University. Additionally, the paper discusses the need to develop new assessment items that comprehensively evaluate students' understanding of programming syntax and problem-solving abilities.

Keywords: Generative Language Model, Coding Education, Assessment Approach, Prompt Engineering

*정회원, 숭실대학교 전기공학부 교수 (제1저자 및 교신저자)
접수일: 2024년 10월 2일, 수정완료일: 2024년 11월 10일
게재확정일: 2024년 12월 5일

Received: October 2, 2024 / Revised: November 10, 2024
Accepted: December 5, 2024
*Corresponding Author: imkgb27@ssu.ac.kr
School of Electrical Engineering, Soongsil Univ, Korea

Ⅰ. 서 론

인터넷이 활성화되기 이전의 코딩 방식은 주로 독립적이고 자율적인 학습과 문제 해결에 의존하였으나 인터넷이 활성화된 이후, 코딩 방식은 급격히 변화하였는데, 온라인 문서, 튜토리얼, 포럼, 그리고 오픈 소스 커뮤니티의 등장으로 프로그래머들은 더 빠르고 쉽게 정보를 얻을 수 있게 되었다. 문제 해결을 위한 코드 예제와 라이브러리를 검색하고, 동료 개발자들과 협업하며, 새로운 기술을 신속하게 습득할 수 있는 환경이 조성되었다.

현재 우리는 생성형 언어모델의 활성화된 시대를 맞이하고 있다. 생성형 언어모델은 자연어로 사용자와 상호작용하며 복잡한 프로그래밍 문제를 신속하게 해결해준다. 이러한 도구들을 코딩에 활용함으로써 더 높은 수준의 추상적 문제 해결과 창의적 사고에 집중하는 능력을 배양하는 동시에, AI의 도움 없이 기초 개념을 이해하고 코드를 작성하는 능력을 유지하는 것도 여전히 중요하다. 따라서 기초 프로그래밍 문법과 알고리즘 교육을 강화하면서도, 생성형 언어모델을 효과적으로 활용하는 방법을 가르치는 새로운 교육 패러다임을 도입해야 할 필요성이 있다.

생성형 언어모델을 활용하는 코딩 교육에서는 학습자에 대한 평가 방법도 기존의 평가 방식과는 달라져야 한다. 전통적인 코딩 교육에서는 주로 문법의 정확성과 알고리즘 구현 능력을 평가했다면, 생성형 언어모델을 사용하는 환경에서는 이러한 평가만으로는 학습자의 실제 역량을 제대로 파악하기 어려울 수 있다. 생성형 언어모델은 학습자에게 복잡한 코드를 자동으로 생성해 줄 수 있기 때문에, 이제는 단순한 코드 작성 능력보다는 문제를 분석하고 언어모델을 활용하여 문제를 효과적으로 해결하는 능력, 그리고 모델이 제공하는 코드를 비판적으로 검토하고 수정하는 능력이 중요해진다. 따라서 학습자의 사고력, 문제 해결 능력, 언어모델과의 상호작용 능력을 종합적으로 평가하는 새로운 평가 방법이 필요하다.

본 논문에서는 S대학교 코딩 수업에서 학습자의 평가를 위한 실시한 시험과 결과 분석을 다루고자 한다. 시험 문항의 유형을 소개하고 코딩 문법을 테스트하는 시험과 생성형 언어모델의 활용을 허용하는 시험 점수에 대한 상관관계를 분석하였다. 이 분석을 통하여 프로그래밍 문법을 숙지하는 능력과 생성형 언어모델을 다루는 능력은 차이점이 있음을 확인하고 이를 반영한 새로운 형태의 평가 문항의 필요성이 있음을 확인하고자 한다.

Ⅱ. 생성형 언어모델과 코딩

생성형 언어모델은 대규모의 텍스트 데이터를 학습하여, 주어진 프롬프트에 따라 자연스럽고 논리적인 텍스트를 생성하는 인공지능 모델을 말한다. 이러한 모델은 학습 과정에서 방대한 양의 소스 코드 데이터를 포함한 다양한 텍스트를 학습하기 때문에, 문법적 패턴이나 프로그래밍 언어의 문맥을 이해하고 생성할 수 있다[1].

프로그래밍 언어는 명확한 문법 규칙과 구조를 가지고 있어, 언어모델이 이를 학습하기에 용이하다. 코드의 패턴이나 알고리즘의 전형적인 흐름을 학습한 모델은 주어진 문제에 대한 해결책을 코드로 생성하는 데 강점을 보인다. 또한, 프로그래밍 언어는 자연어에 비해 문맥을 해석하는 데 더 명확한 규칙이 존재하므로, 생성형 언어모델은 이러한 규칙을 기반으로 오류를 줄이고 더 정확한 코드를 생성할 수 있다. 이 때문에 생성형 언어모델은 반복적인 코딩 작업, 코드 자동 완성, 디버깅, 코드 최적화 등에서 개발자들에게 유용한 도구로 사용되고 있으며, 코딩 교육에서도 활용도가 높아지고 있다.

생성형 언어모델에 대한 접근성이 용이해짐에 따라 교육 및 학술 연구에 활용하는 방법 및 효과에 대한 관심이 높아지고 있으며 이에 관련된 여러 연구가 진행되어 왔다[2-6]. ChatGPT를 교육에서 활용할 때, 고려해야 할 요소를 학습자의 인식을 중심으로 확인하고자 하는 연구와 교수자들을 대상으로 한 면담을 통해 교수자의 역할을 탐색하고자 하는 연구가 진행되었다[2,3]. ChatGPT를 활용한 학술연구 검색을 통해 국내외 연구동향을 살펴보는 연구도 진행되었다[4]. ChatGPT의 오남용에 대한 연구들도 진행되었는데 또한 대학 신입생들을 대상으로 한 설문조사를 통해 오남용 실태 및 해결책을 모색하는 연구와 표절검사시스템에 대한 현황을 조사하고 이에 대한 대학생들의

인식도를 조사하기도 하였다[5,6]. 이와 더불어 생성형 언어모델을 코딩에 활용하는 것에 관해서도 많은 연구자들의 관심을 끌고 있는데, 교양필수 기초코딩 교과목에서 ChatGPT 활용에 대한 효과성을 조사하는 연구와 프로그래밍 수업에서 ChatGPT를 활용하여 문제를 푸는 과정을 경험한 학생들을 대상으로 프로그래밍 환경 변화에 대한 전망을 조명하는 연구도 진행되었다[7, 8]. 또한 프로그래밍에 대한 개념 학습, 예제 실습, 연습 실습, 자유 실습에서 ChatGPT 활용을 위한 프롬프트 설계에 대한 연구도 진행되었다[9]. 앞선 연구 결과들로부터, 대체적으로 코딩 교육에서 ChatGPT와 같은 생성형 인공지능 도구가 프로그래밍 학습을 용이하게 한다는 장점과 더불어 AI챗봇에 의존적이게 되어 역량 계발에 부정적인 효과가 나타날 우려에 대해서도 지적하고 있다.

III. 코딩 학습 성취도 평가

코딩교육의 학습 목표는 기본 프로그래밍 문법에 대한 이해, 논리적이고 체계적으로 사고하는 능력, 데이터 구조에 대한 이해, 개발 환경 및 패키지 활용방법 숙지 등이라 할 수 있다. 이러한 능력을 평가하기 위해서 기존에 실시한 코딩 시험은 문법적 오류를 수정하라는 문제와 간단한 과업을 수행하는 코드를 작성하라는 유형의 문제들을 주로 출제하였다. 그림 1에는 기존 코딩 시험 문제의 예시로, 문법적 오류를 수정하라는 문제와 그래프를 그리거나 문제를 해결하는 코드를 작성하라는 문제가 제시되어 있다.

```
# Rectangle 이라는 클래스를 정의하는 코드의 오류를 수정하시오.
class Rectangle:
  def __init__(self, x1, y1, x2, y2):
    self.x1 = x1
    self.y1 = y1
    self.x2 = x2
    self.y2 = y2

# 리스트 안에서 인접한 두 숫자의 곱들의 모든 합을 구하는 코드를 작성하고자 한다
# 오류들을 수정하시오.
list1=[2, 4, -1, 3, 5, 6]
for i in range(len(list1))
  a=list1[i]*list1[i+1]
  sum += a
print(sum)
```

(a) 문법적 오류 수정 문제

(1) $y = \sin(0.2x) + \sin(2x)$**를 그려보시오. (x축의 범위는 0부터** 2π**까지)**

문제 7. $i - 2i^2 + 3i^3 - 4i^4 + \cdots + (-1)^{n+1} n \cdot i^n = 8 + 7i$**를 만족시키는 자연수** n**의 값을 출력하는 코드를 작성하시오(반드시 while문을 사용하시오).**

(b) 과업 수행 문제

그림 1. 기존 코딩 시험 문제 예시
Figure 1. Examples of conventional coding problems

2024학년도 1학기 수업에서는 기존 학기의 시험과는 달리 다음 두 가지 형태의 시험을 실시하였다. 첫 번째는 2~4 줄로 작성된 간단한 코드를 실행했을 때의 결과를 답안으로 작성하는 필기시험이고, 두 번째 시험은 주어진 과업을 수행하는 파이썬 코드를 작성하는 코딩시험이다. 표 1에 필기시험 문항 예시를 나타내었다. 생성형 언어모델을 활용하는 시대임에도 불구하고 프로그래밍 언어의 기본적인 문법은 숙지하여야 하므로 필기시험은 이를 평가하기 위한 최소한의 테스트라고 할 수 있다.

표 1. 필기시험 문항 예시
Table 1. Examples of questions for the written exam

import numpy as np a = np.array([[2,3,4],[4,5,6]]) print(a[:,-1])
res = 0 for i in range(5):   res += i print(res)
import numpy as np a = np.array([2,3,4]) print(a//2 == 1)
list = [1, 1, 2, 3, 5, 8, 13] print(list[list[4]])

코딩시험은 수강생들이 네트워크에 연결되어 있는 PC에 접속하여 구글 코랩 (Google Colaboratory) 환경에서 직접 코딩하여 결과물을 제출하도록 하였다. 구글 코랩은 구글 드라이브에서 사용할 수 있는 사용할 수 있는 앱으로서 구글 서버에 접속하여 텍스트 셀과 코드 셀로 이루어진 ipynb (Interactive Python Notebook) 환경에서 코드를 작성하고 실행하여 결과를 즉각적으로 확인할 수 있는 장점이 있다. 2023년 상반기에 Gemini를 기반으로 하는 생성형 언어모델을 사용하여 코드를 생성할 수 있는 기능이 추가되었다. 그림 2는 구글 코랩에서 추가된 코드 셀을 보여주고 있는데, “생성”부분을 클릭하면 프롬프트를 입력할 수 있는 창이 뜨고 여기에 작성하고자 하는 코드에 대한 설명을 입력하여 코드를 생성할 수 있다. 프롬프트를 입력하여 코드를 생성하는 예시를 그림 2에 나타내었다.

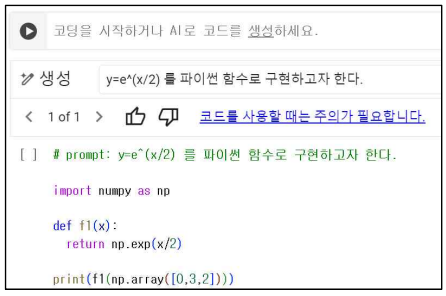
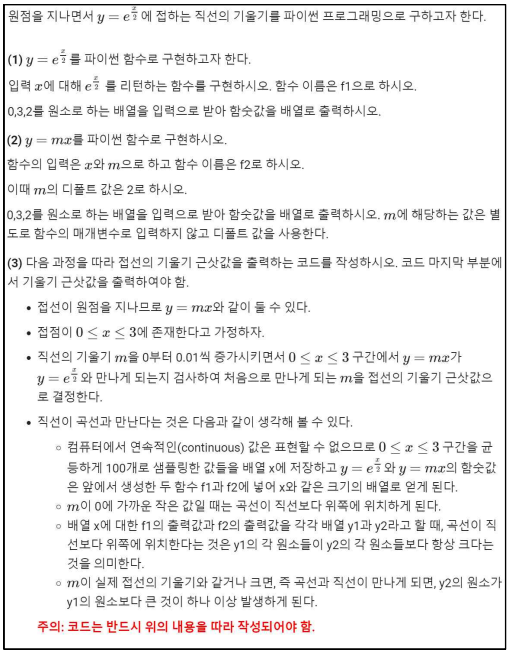


그림 2. 프롬프트 입력으로 코드 생성 예시
Figure 2. Example of code generation using prompt input

예시에서와 같이 자연어로 된 문장으로 원하는 코드를 생성할 수 있는데, 그 수준이 매우 뛰어나서, 많은 경우 사용자가 프롬프트 입력에 대한 별다른 고민없이 문제에 주어진 지문을 그대로 프롬프트에 입력하는 것만으로도 정답에 가까운 코드를 생성할 수 있다. 이는 프롬프트 입력에 대한 별다른 고민 없이도 결과를 얻을 수 있는 상황을 만들며, 이러한 단순한 접근 방식은 평가 시 학습자 간의 변별력을 희석시키는 결과를 초래할 수 있다. 따라서 생성형 언어모델을 사용하는 코딩 시험에서는 학습자 간 변별력을 확보할 수 있도록 복합적이고 창의적인 문제 출제가 중요해지는데, 다음 세 가지 요소를 고려하여 문제를 출제하였다. 첫째, 문제를 여러 단계로 나누어 각 단계를 해결해야 최종 답에 도달할 수 있게 하였다. 각 단계는 서로 의존적이며, 이전 단계를 제대로 이해하지 못하면 다음 단계를 해결하기 어렵도록 설계하였다. 둘째, 생성형 언어모델이 학습하지 못했을 가능성이 높은 창의적인 해결과정을 제시하였다. 셋째, 언어모델이 잘못된 결과를 도출할 가능성이 높은 문제를 출제하여, 학생들이 생성형 언어모델의 오류 가능성을 인식하고, 생성형 언어모델이 생성한 코드를 충분히 이해하여 잘못된 부분을 파악하고 수정하도록 하였다.

그림 3에 수록된 코딩 시험 문제 예시에서는 “원점을 지나면서 $y = e^{\frac{x}{2}}$ 에 접하는 직선의 기울기를 파이썬 코드로 구현하라”는 문항이 주어져 있다. 이 지시문을 복사하여 그대로 생성형 언어모델의 프롬프트로 입력하면 주어진

내용을 수행하는 코드를 작성해주므로 변별력 확보를 위해 특정 과정을 따라 코딩하라는 제약 조건을 추가로 부여하였다. 세 개의 과정으로 구분하였는데 (1)과 (2)는 비교적 간단한 내용으로서 지시문을 거의 그대로 생성형 언어모델에 프롬프트로 입력하여도 올바른 코드를 생성해주는 것을 확인할 수 있었다. 이와 달리 (3)의 과정은 접선을 구하기 위해 일반적이지 않고 창의적인 과정을 제시하여 지시문을 프롬프트에 입력하는 것만으로는 생성형 언어모델이 원하는 코드를 출력하기 어렵게 하였다. 올바른 결과를 얻기 위해서는 답을 얻는 과정을 충분히 이해하고 생성형 언어모델과 수 차례 대화 과정을 거쳐야 한다.


원점을 지나면서 $y = e^{\frac{x}{3}}$에 접하는 직선의 기울기를 파이썬 프로그래밍으로 구하고자 한다.

(1) $y = e^{\frac{x}{3}}$를 파이썬 함수로 구현하고자 한다.

입력 x에 대해 $e^{\frac{x}{3}}$ 를 리턴하는 함수를 구현하시오. 함수 이름은 f1으로 하시오.

0,3,2를 원소로 하는 배열을 입력으로 받아 함숫값을 배열로 출력하시오.

(2) $y = mx$를 파이썬 함수로 구현하시오.

함수의 입력은 x와 m으로 하고 함수 이름은 f2로 하시오.

이때 m의 디폴트 값은 2로 하시오.

0,3,2를 원소로 하는 배열을 입력으로 받아 함숫값을 배열로 출력하시오. m에 해당하는 값은 별도로 함수의 매개변수로 입력하지 않고 디폴트 값을 사용한다.

(3) 다음 과정을 따라 접선의 기울기 근삿값을 출력하는 코드를 작성하시오. 코드 마지막 부분에서 기울기 근삿값을 출력하여야 함.

- 접선이 원점을 지나므로 $y = mx$와 같이 둘 수 있다.
- 접점이 $0 \le x \le 3$에 존재한다고 가정하자.
- 직선의 기울기 m을 0부터 0.01씩 증가시키면서 $0 \le x \le 3$ 구간에서 $y = mx$가 $y = e^{\frac{x}{3}}$와 만나게 되는지 검사하여 처음으로 만나게 되는 m을 접선의 기울기 근삿값으로 결정한다.
- 직선이 곡선과 만난다는 것은 다음과 같이 생각해 볼 수 있다.
  - 컴퓨터에서 연속적인(continuous) 값은 표현할 수 없으므로 $0 \le x \le 3$ 구간을 균등하게 100개로 샘플링한 값들을 배열 x에 저장하고 $y = e^{\frac{x}{3}}$와 $y = mx$의 함숫값은 앞에서 생성한 두 함수 f1과 f2에 넣어 x와 같은 크기의 배열로 얻게 된다.
  - m이 0에 가까운 작은 값일 때는 곡선이 직선보다 위쪽에 위치하게 된다.
  - 배열 x에 대한 f1의 출력값과 f2의 출력값을 각각 배열 y1과 y2라고 할 때, 곡선이 직선보다 위쪽에 위치한다는 것은 y1의 각 원소들이 y2의 각 원소들보다 항상 크다는 것을 의미한다.
  - m이 실제 접선의 기울기와 같거나 크면, 즉 곡선과 직선이 만나게 되면, y2의 원소가 y1의 원소보다 큰 것이 하나 이상 발생하게 된다.

주의: 코드는 반드시 위의 내용을 따라 작성되어야 함.

그림 3. 생성형 언어모델 사용을 허용하는 문제 예시
Figure 3. Example of test statement allowing the use of generative language models

표 2에 소문항 (1)~(3)에 대한 수강생들의 정답률을 나타내었다. (1)과 (2)에 비해 (3)의 정답률이 현저히 떨어지는 것을 확인할 수 있다.

표 2. 문항별 정답률
Table 2. Percent correct per question

	(1)	(2)	(3)
정답률	87 %	85 %	41 %

Ⅳ. 평가 결과 분석

본 연구에서는 기본 문법을 평가하는 필기시험 점수와 생성형 언어모델을 활용하는 코딩 시험 점수 간의 상관관계를 분석하였다. 이를 위해 대조군으로 기존 코딩 시험(2023년)의 중간고사와 기말고사 점수 데이터를 사용하였다. 일반적으로 같은 유형의 시험 점수 간에는 어느 정도 높은 상관관계를 보인다고 가정할 수 있으며, 이와 비교하여 2024년 필기시험과 코딩 시험 점수의 상관관계를 비교 분석하였다. 선형 상관관계를 측정하는 피어슨 상관계수와 비선형 상관관계를 측정하는 스피어만 상관계수 및 켄달타우 상관계수를 모두 계산하여 보다 포괄적인 분석을 실

시하였다.

분석 결과 (표 3), 기존 코딩 시험(2023년)의 중간고사와 기말고사 점수 간의 피어슨(Pearson), 스피어만(Spearman), 켄달타우(Kendall's Tau) 상관계수는 각각 0.55, 0.56, 0.42로 나타났고, 기본 파이썬 문법을 평가하는 필기 시험과 생성형 언어모델 활용을 허용한 코딩 시험간의 상관계수는 피어슨 0.19, 스피어만 0.18, 켄달타우 0.13으로 나타나, 두 시험 점수 간의 상관관계가 매우 낮음을 알 수 있다. 이러한 결과는 생성형 언어모델을 활용한 코딩 시험이 기존의 코딩 시험과는 다른 평가 기준을 가지고 있으며, 학생들의 기초 프로그래밍 지식과 생성형 언어모델을 활용한 문제 해결 능력 간의 연관성이 낮음을 시사한다.

표 3. 상관계수들
Table 3. Correlation coefficients

	피어슨	스피어만	켄달타우
중간시험 - 기말시험	0.55	0.56	0.42
필기시험 - 코딩시험	0.19	0.18	0.13

그림 4와 그림 5에 각각 필기시험-코딩시험 점수와 대조군으로서 기존 코딩시험(2023년)의 중간고사-기말고사 점수에 대한 산포도를 나타내었다.

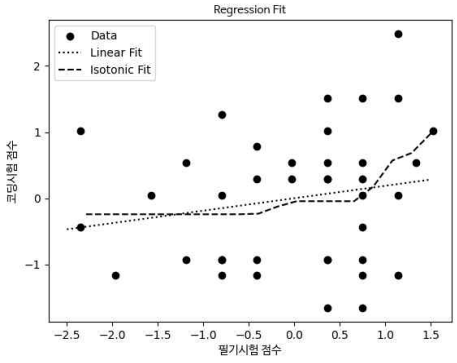


그림 4. 필기시험과 코딩시험 점수: 산포도와 선형회귀 및 등위회귀 결과
Figure 4. Scores for written test and coding test: scatter plot, linear regression, isotonic regression

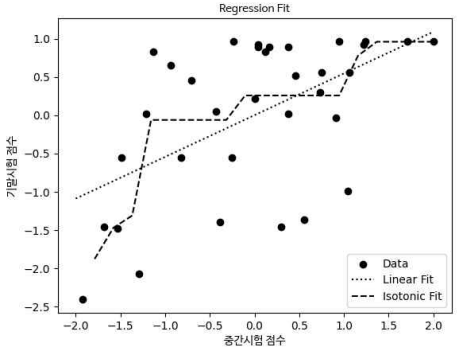


그림 5. 중간시험과 기말시험 점수: 산포도와 선형회귀 및 등위회귀 결과
Figure 5. Scores for midterm and final: scatter plot, linear regression, isotonic regression

산포도와 함께 선형회귀를 나타내는 직선과 등위회귀 곡선을 표시하였다. 등위회귀(Isotonic Regression)는 데이터

포인트 간의 순서 관계를 보존하면서 단조증가(또는 단조감소)하는 비선형 함수를 찾는 회귀 기법으로서 주로 변수 간의 단조 관계를 유지하면서 데이터의 경향을 모델링하는 데 사용된다[10].

표 4에는 회귀결과에 대한 결정계수(coefficient of determination: R-squared)도 나타내었는데, 결정계수는 회귀 모델의 성능을 평가하는 중요한 지표로, 모델이 데이터를 얼마나 잘 설명하는지에 대한 정보를 제공한다. 1에 가까울수록 설명성이 높다고 볼 수 있는데 표에 나타난 결과에 따르면 두 가지 점수분포 모두 선형회귀보다는 등위회귀 곡선이 데이터를 더 잘 설명하는 것으로 확인되었으며, 기존 코딩시험의 중간시험-기말시험 데이터보다 필기시험-코딩시험 데이터의 결정계수가 현저히 낮게 나타나 표 3의 상관계수에서 확인한 바와 같이 필기시험과 코딩시험 점수 간에는 관련성은 상당히 희박한 것으로 결론지을 수 있다. 이상의 분석을 통해 기본적인 문법을 이해하는 능력과 생성형 언어모델을 능숙하게 활용하는 능력은 상관관계가 낮음을 알 수 있다.

표 4. 결정계수
Table 4. Coefficient of determination (R-squared)

	선형회귀	등위회귀
중간시험 - 기말시험	0.30	0.55
필기시험 - 코딩시험	0.035	0.11

V. 결 론

생성형 언어모델들의 수준이 높아짐에 따라 여러 분야에서 활용되고 있으며 대학에서도 연구, 교육, 학습에 걸쳐 활용도가 증대되고 있다. 생성형 언어모델의 뛰어난 코딩 능력은 이제 프로그래밍의 필수 도구로 자리 잡고 있으며, 이로 인해 코딩 교육도 언어모델을 효과적으로 활용하는 방식으로 변화해야 할 것이다. 하지만 생성형 언어모델의 사용을 허용할 경우 프로그래밍에 대한 이해도가 낮아도 정답을 도출할 가능성이 높기 때문에 기존의 코딩 시험으로는 학습자의 성취도를 변별하기 어려워졌다. 따라서 코딩 교육에서는 학습자가 프로그래밍 언어의 기본적인 문법을 충분히 이해하도록 하면서도, 생성형 언어모델을 능숙하게 활용할 수 있는 능력을 배양하는 것이 중요하다. 이를 위해서는 학습 평가 방식을 재설계하여, 학습자가 문제를 해결하는 과정에서 언어모델을 어떻게 활용하고, 생성된 코드를 어떻게 분석하며, 목표를 달성하는지를 종합적으로 평가할 수 있어야 할 것이다.

우리는 이러한 평가 관점에서, 기본적인 문법을 테스트하는 전통적인 시험과 생성형 언어모델을 활용하여 코딩하는 능력을 테스트하는 시험의 차이점을 분석하였다. 이 분석을 통해, 언어모델을 활용하는 코딩 시험이 기존의 시험과는 다른 평가 기준을 요구한다는 점을 확인하였다. 또한, 생성형 언어모델을 활용하여 코딩하는 능력을 평가하기 위해서는 기존 코딩 시험과는 다른 문항 설계의 필요성을 강조하였다. 앞으로의 코딩 교육에서는 생성형 언어모델을 활용한 문제 해결 능력을 강화하는 동시에, 이를 공정하고 정확하게 평가할 수 있는 방법을 지속적으로 연구하고 발전시켜야 할 것이다.

References

[1] Sangyeop Yeo, Yuseung Ma, Hyungkook Jun, Taeho Kim, and SangCheol Kim, "Survey on Recent Large Language Models for Code Generation," *KISE Transactions on Computer Practices*, Vol. 30, No. 8, pp. 372-381, August 2024. https://doi.org/10.5626/KTCP.2024.30.8.372
[2] Hyeong-Jong Han, "A Qualitative Research on Exploring Consideration Factors for Educational Use of ChatGPT," *JCCT*, Vol. 9, No. 4, pp. 659-666, July 2023. http://dx.doi.org/10.17703/JCCT.2023.9.4.659
[3] Hyeong-Jong Han, "Exploring Instructor Roles in Operating and Integrating ChatGPT into Classroom in

Higher Education," *Journal of Digital Contents Society*, Vol, 25, No. 2, pp. 465-474, February 2024. http://dx.doi.org/10.9728/dcs.2024.25.2.465

[4] Rachel Ju, Yerin Choi, Ji Hoon Song, and Myunghyun Yoo, "Potential Impact of ChatGPT on Education and Academic Research: Review of Trends in Korea and Overseas," *Journal of Educational Technology*, Vol. 39, No. 4, pp. 1401-1447, 2023. http://dx.doi.org/10.17232/KSET.39.4.1401

[5] Pyong Ho Kim, Ji Won Yoon, and Ju Hyung Yoo, "College Students' Perspectives on ChatGPT Integration in Higher Education and Relevant Ethical Considerations," *IJACT*, Vol. 12, No. 1, pp. 234-241, March 2024. https://doi.org/10.17703/IJACT.2024.12.1.234

[6] Hyun-ju Kim and Jinyoung Lee, "A Study on the University Education Plan Using ChatGPT for University Students," *JCCT*, Vol. 10, No. 1, pp. 71-79, 2024. http://dx.doi.org/10.17703/JCCT.2024.10.1.71

[7] Wanseop Kim, "Analysis of the Educational Effects Regarding the Use of ChatGPT in Compulsory Basic Coding Subjects," *Journal of General Education*, Vol. 17, No. 5, pp. 113-123, October 2023. https://doi.org/10.46392/kjge.2023.17.5.113

[8] Ramazan Yilmaz and Fatma Gizem Karaoglan Yilmaz, "Augmented Intelligence in Programming Learning: Examining Student Views on the Use of ChatGPT for Programming Learning," *Computers in Human Behavior: Artificial Humans*, pp. 1-7, 2023. https://doi.org/10.1016/j.chbah.2023.100005

[9] Seul Ki Kim, "Exploring the Possibility of Using Generative Artificial Intelligence for Programming Education: Focusing on ChatGPT," *Proceedings of The Korea Association of Computer Education*, Vol. 27, No, 2, pp. 151-154, 2023.

[10] R. E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. D. Brunk, *Statistical Inference Under Order Restrictions: The Theory and Application of Isotonic Regression*. Wiley, 1972.