Contents

1.0 Introduction

This page is intentionally blank.

2.0 Literature Review

The following literature review presents the necessary background to better understand Large Language Models (LLMs) and their impacts on how novice programmers learn to code. We begin by exploring key research around LLMs and how they have impacted disciplines since their inception. Specifically, we provide evidence of LLMs as a disruptive innovation in the field of computer programming. LLMs have transformed not only how people code, but the ways in which programmers think about programming itself. This has implications for not only how code will be written in the future, but what skills are necessary for becoming a programmer and the methods by which those skills are taught to novices.

In addition, this literature review will explore the research of LLM use by teachers, students and administration within higher education. The research embodying the strengths, drawbacks, and unique challenges of LLMs within this context is discussed. This is necessary to differentiate between the overarching issues of LLMs in education and those specific to the topic of computer programming education.

Research will show LLMs are transforming how skilled professionals write code. LLMs are being used predominantly by teachers and students in computer programming education as well, which has forced academics to rethink not only how to teach computer programming to novices, but what should be taught in the first place. The literature will identify this through the positive and negative impacts LLMs are having programming education.

The academic literature on computer science education discusses the well-known challenges of novice users learning to program. These challenges are the impacts of cognitive load on learning, getting learners to focus on computational literacy over syntax and technology, the role of self-efficacy and motivation on success, and the importance of metacognition for building problem-solving skills. This literature review will revisit each of these challenges through the lens of how LLMs are impacting them both positively and negatively.

2.1 Rise of the Large Language Model (LLM)

Natural Language Processing (NLP) has undergone significant changes in the past few years. With the introduction of transformer-based neural network architecture, Natural Language Processing took a giant step forward in performance and accuracy (Gillioz et al., 2020).

The Generative Pre-Trained Transformer (GPT) reasonably predicts the next word in a sequence using the input and previously generated output. While predicting the next token in the sequence is the extent of their ability (Shanahan, 2024), transformer-based language models trained on large data have demonstrated highly effective reasoning capabilities.

Open AI's foundational paper, "Language Models are Few-Shot Learners", demonstrated these transformer-based language models can produce human-level performance when they are trained on large copra (Brown et al., 2020). This was a pivotal discovery because at the time since prior to this paper transformer-based models were trained to be relatively task specific (Zhao et al., 2023).

The paper from Brown et al., 2020 led to significant advancements in research with respect to understanding the capabilities of LLMs. Wei et al. (2022) discovered reasoning capabilities of LLM's can be improved through a technique called chain-of-thought prompting. By including few-shot examples that break down complex reasoning into steps, the LLM can use those shots provided as an example of how to explain a complex process.

(Halevy et al., 2009)seminal paper, "The unreasonable effectiveness of data", explains as the training data set size increases, so does the model accuracy. In addition, the specific selected model algorithm becomes less relevant as training data set size increases. (Wei, Bosma, et al., 2022) documented a similar effect with large language models. Larger-sized models exhibited emergent abilities not found in their smaller counterparts. Examples of emergent abilities seen in the larger models include complex arithmetic and reading comprehension.

2.2 LLMs as a Disruptive Innovation for software development

(Christensen et al., 2018) divide technological innovations into two distinct types. Sustaining innovations improve existing products and services, while disruptive innovations provide a unique set of features to an initial set of customers. From the perspective of companies that offer generative AI such as Google,

Anthropic, Open AI, and Microsoft, Horn considers generative AI to be a sustaining innovation (Horn, 2024). In their comprehensive literature review of AI as a disruptive innovation, (Păvăloaia & Necula, 2023) consider the application of Generative AI to be a disruptive innovation across different sectors such as healthcare, agriculture business and education.

The rise of AI-assistant programming tools in industry is evidence of this disruption. There are a growing set of tools available in the cloud: Github Copilot, Amazon CodeWhisperer, Gemini Code assist, Claude Code, Open AI Codex v2, Tabnine, and Codeium, in addition to self-hosted options like FauxPilot, Tabby, and CodeLLama. While each provides a unique set of features for differentiation, they all have primary functions like code completion, code generation, code explanations, and discussion. The primary value-add touted by these tools is developers will be able to write code in less time, improving productivity. Talk about "vibe coding" and Lovable, Bolt, Replit and Cursor.

3.0 Methods

3.1 Introduction

The objective of this research is to study the impact of generative AI on an individual's learning. Specifically, this research studied the impacts of the Large-Language Model (LLM) used by students enrolled in an introductory Python programming course. Their learning was studied through the lenses of academic performance on a summative assessment at the mid-term of the course in addition to scores on a computational literacy instrument. LLM use was classified into activities that either support or avoid learning, which will then be quantified by for each participant. This chapter explains the methodology that was used to address the research questions of this study.

3.1.1 Research Questions

RQ1: How does large-language model use influence student learning performance?

- This was measured by first classifying chat logs into counts of const/unconst then

- performing a multi-variate regression of the key influencers Constructive/Unconstructive => E1

RQ2: When the prompt is adjusted to include assignment instructions (in-context learning) what is the impact on student learning performance?

- This was measured by first classifying chat logs into counts of const/unconst then

- Checking control / treatment against E1 for any statistical significance.

RQ3: What is the relationship between large language model use and computational literacy?

- This was measured by first classifying chat logs into counts of const/unconst then

- performing a multi-variate regression of the key influencers Constructive/Unconstructive => E1

3.2 Study Design

A diagram of the study design
Figure1: An overview of the study design.

3.2.1 Overview

The study took place over a six-week period in the Spring 2025 semester of an introductory Python programming course, IST256 (https://ist256.com/spring2025/syllabus/). Taught within the School of Information Studies at Syracuse University, the course teaches programming fundamentals from the informatics perspective and is intended for non-computer science majors. There were 173 students enrolled in the course. The study only focused on the first 6 weeks because these units cover basic computational literacy constructs as applied to the Python programming language. These include instruction sequencing, variables, branching, iteration, and composition (functions). The course schedule can be found here: (https://ist256.com/spring2025/syllabus/#course-schedule). The study was exempt from IRB under section §46.104 section 1, which covers research conducted in an established educational setting. A university IRB Exemption application has been filed and obtained for category 1 for research in established or commonly accepted educational settings. The IRB# is 24-346. While all elements of the study design were part of the course, students could elect to opt-out of including their data in the study.

The study begins with a diagnostic instrument C1 to get the baseline of computational literacy for each participant. This happened within the first week of class before any instruction. At this time students were introduced to the course-provided AI, (https://ai.ist256.com), which was a Large Language Model (GPT4o-mini) configured with a system prompt. Students were encouraged to use the AI as a virtual tutor asking it for help with Python questions and course-related assignments. When asking a specific question about an assignment, students were instructed to switch the LLM context by selecting the assignment in question from a drop-down menu.

Figure 2: Context-Selection from the IST256 AI Tutor

For the control group T1 this action did nothing - it does not add any additional context. For the treatment group T2 the action copied the assignment or lab instructions into the conversational context.

Figure 3: The treatment group (T2) is aware of the selected content.

Because the chatbot was self-hosted, all student interactions and AI responses were captured. D1 chatbot trace data is a dataset of those collected interactions throughout the six weeks of use. After the six week period, there were three observations. E1 was the midterm exam in the course covering the content from the first six weeks, C2 was a re-issue of the same C1 diagnostic as a means to measure improvement of computational literacy. Observation Q1 was a questionnaire that asked for simple demographic data about each participant. Q1 was issued at the end of the course along with course evaluations.

3.2.1.1 How the Study Addressed the Research Questions All three research questions explored the influence of Large-Language Model use on learning and computational literacy. Therefore an analysis of the chatbot trace data D1, was crucial to answering the research questions. Categorical content analysis [CITE - foundation content analysis] was used to classify student AI interactions into two categories of learning-supportive and learning-avoidance activities. These activities were deductively coded based on observations of other researchers in literature in addition to my six years of experience teaching the course.[CITE deductive]. The resulting categories were quantified and grouped to establish a quantized profile of AI use by each participant. These category counts were the independent variables of this study.

To answer RQ1, a multivariate regression was performed with the category counts as the independent variables and E1 as the dependent variable. RQ2 introduces the control / treatment independent variable to the multivariate regression from RQ1. Finally for RQ3 the dependent variable is changed to C2.

3.2.2 Participant Funnel

Participants were students recruited from a Spring 2025 section of IST256. There were 173 students enrolled in the course. The research was exempt from IRB under section §46.104 section 1, which covers research conducted in an established educational setting. A university IRB Exemption application was filed and approved for category 1 for research in established or commonly accepted educational settings under the Syracuse University IRB number #24-346.

Of the 173 participants only 126 consented to the study. Five of the students who consented did not complete the observations necessary for the dependent variables: C1, C2 or E1, leaving 121 participants.

3.2.2.1 Population 1: Chatbot participants There are two populations under analysis. The first population is the set of participants who used the AI chatbot and therefore have the trace data D1 necessary to study their AI interactions. There were 48 individuals in the control

group and 39 in the treatment group for a total participant population of 87. *Figure 4: The participant funnel for chatbot use. 87 participants.*

3.2.2.1 Population 2: Chatbot participants with Survey Responses A survey Q1 was issued to students with the goal of identifying possible covariates. Besides demographic questions around the year of study, major, and gender students were also asked about their programming experience prior to the IST256 course. Ten chatbot users did not complete this survey thus reducing the participant size down to 77 whenever survey responses were needed in the analysis. Among the 77 participants, 41 were in the control group and 36 were in the treatment group.

Figure 5: The participant funnel when accounting for survey responses.

3.2.3 Computational Literacy Instrument (C1/C2)

No consensus models exist for developing computational thinking (Shute et al., 2017), therefore no commonly accepted instrument for measuring computational thinking exists. Brennan & Resnick, (2012) assert three key dimensions of the computational thinking framework: computational concepts, computational practices, and computational perspectives. Computational concepts are programming structures like loops and conditions, debugging and remixing are examples of computational practices, and activities such as expressing and framing problems are examples of computational perspectives. The psychometrically validated CT-Test instrument developed by Román-González et al., (2017) measures computational concepts only. The computational concepts evaluated through the instrument such as loops, branching, arrays, functions, instruction sequencing aligned closely with the computational concepts taught in the first 6 weeks of content in the IST256 course.

The CT-Test consists of twenty-eight multiple-choice questions, and students are given 45 minutes to complete the test. The test questions were delivered online one question at a time in a random order using Syracuse University's Blackboard learning management system. Like the author's original instrument, students were permitted to skip questions and revisit them later. The second test C2 had the same questions as C1 only the order of the questions was re-randomized for each student. Students received a score out of 28 on the test but no indication of what questions they got correct / incorrect.

The following figure is an example question from the CT-Test. The entire instrument can be found in appendix A.

Figure 6: A sample question fro the CT-Test

3.2.4 Chatbot Design

The AI chatbot website used in the experiment (https://ai.ist256.com) was programmed in Python using the Streamlit framework (https://streamlit.io/). It was released as open source under the Apache 2.0 license and published to Github (https://github.com/cent-ischool/ist256-chatapp). The application was deployed to a Kubernetes cluster in the Syracuse University data center. Users were required to authenticate with their Syracuse University credentials to use the application. Their chat interaction trace data was stored in a Postgresql database in the same data center. A python script was written to extract the data from the database.

3.2.4.1 LLM Selection Acceptance criteria for choosing an LLM for the AI chatbot came down to accuracy and response time. I wanted the accuracy and response time to be on-par with the hosted platforms at that time: ChatGPT ((https://chatgpt.com/) and Claude (https://claude.ai/), so that participants would not churn based on those criteria.

Open-source models such Dolphin3, Llama3 and Qwen2.5 were considered initially. Unfortunately their response times fell off significantly at scale, likely due to these models being hosted on time-shared Syracuse-University GPU hardware. Their accuracy was found to be suitable for the content of the course, however.

Ultimately, the large language model selected Open AI's GPT4o-mini. The model, which strikes a balance between price, response time and performance, had an adequate understanding of introductory Python, and since it was hosted in the Azure cloud it handled concurrent text generations. GPT4o was also evaluated, but since it performed similarly, at 16 times the price of 4o-mini, the latter seemed an obvious choice.

GPT4o-mini was a very adequate model. With correct prompting, it was able to generate correct solutions to all programming assignments and labs in the course. It also generated correct responses to 43 of 45 questions on the midterm exam (E1), placing it in the 98th percentile.

3.2.4.2 Random Assignment Once authenticated to the AI chatbot website, participants were assigned to the control or treatment groups based on the MD5 hash of their student ID being odd or even. This method assured each student was always assigned to the same group at each login session. The algorithm is provided here:

```
hash = md5(student_id)
number = int(hash)
in_treatment = mod(number, 2) # odd or even
```

3.2.4.3 Control Group (T1) Every request to the AI Tutor included a system prompt. The system prompt was used to place some guardrails on the AI and control the generated output. The system prompt's objectives were set to:

(1) generate Python code in the style which is taught in the course, (2) explain all code generated written, (3) adopt a persona of a helpful and friendly tutor for a college level introductory Python course, and (4) not answer questions that are not course related.

Several system prompt iterations were evaluated before the following was determined to be most suitable for meeting the objectives while minimizing the number of input tokens.This was the base model configuration for the control group (T1).

Figure 7: The system prompt for the control group T1.

3.2.4.4 Treatment Group (T2) The treatment group consisted of the control group LLM and configurations (T1) plus a user prompt injected into the conversation based on contextual selection. For example, if the participant selected **03-HW-Conditionals** as the context (assignment), the content of the homework assignment notebook **lessons/03-Conditionals/HW-Conditionals.ipynb**, was loaded into the chat conversation. This was the same assignment to be completed by the student and contains the instructions, suggested approach, sections where code must be written and any sample code. The following prompt template is used to add the information to the conversation:

A black screen with white text AI-generated content may be incorrect.

Figure 8: Treatment T2 context prompt template.

Subsequently, the AI chatbot responds that it is ready to assist with the assignment. The AI response demonstrated context awareness.

A black background with white text AI-generated content may be incorrect.

Figure 9: AI response to context selection from T2.

When a participant asked the AI to complete a specific section of the assignment, the AI responded with working code and with an explanation of it. This is a key difference between T1 and T1. The control group T1 could generate the working code as well, but the participant must elicit the prompt to generate the results. This could be achieved by copying the assignment instructions into the prompt, which was all T2 was doing essentially.

The purpose of the control / treatment group was to help understand the impact of context-awareness on learning. For example, did the context-awareness improve user engagement with AI, or simply make question-answering more convenient?

3.2.5 Midterm Exam (E1)

The purpose of the midterm exam was to measure students' understanding of the fundamental concepts in the course. The exam covered:

1. Conceptual ideas, e.g. Definite and indefinite loops

2. Application of the concepts, e.g. when does one use a definite vs indefinite loop?

3. Concept in Python language, e.g. how does one use while or for to write loops? Is the loop you see a definite or indefinite loop?

4. Application in Python language, e.g. here is code with a while loop. What is the output upon execution?

The exam contained 45 multiple choice questions and students were given 45 minutes to complete it. The exam was closed book with the exception of a single page of notes as a study guide. Exam and academic integrity were preserved by issuing the exam in class and on paper. There were54 versions of the same exam, with each version having both questions and answers randomized. Versions A-D were issued in class, version E was reserved for students at the testing center.

The Cronbach's Alpha of the exam versions was between 0.83 and 0.88, indicating high reliability the instrument reflects students' subject knowledge.

Exam Version (E1)	N	Cronbach's Alpha
A	40	0.87
B	40	0.86
C	40	0.83
D	39	0.87
E	9	0.88
	168	

Table 1: Cronbach's Alpha of E1 midterm exam

3.2.6 Questionnaire (Q1)

A debriefing questionnaire (Q1) was issued after the experiment. The purpose of this questionnaire was to:

1. Collect basic demographics from the participant pool. Such as gender, major, and class rank.

2. Identify participants with prior computer programming experience.

3. Identify participants who used AI other than the AI provided.

4. Collect student perceptions of programming and AI use.

The questionnaire was delivered using the Syracuse University Qualtrics survey system and can be found in Appendix B. Only demographic data and responses to prior programming experience were used in the data analysis.

3.2.7 Chatbot Trace Data (D1)

As explained earlier the AI chatbot recorded participant interactions and AI responses into a Postgresql database. The data included when they interacted with the AI, the context they selected, if they were in the control or treatment group. Here's a data dictionary of the fields in the trace data:

Field	Data Type	Purpose
Id	Sequential integer	A unique number for each item in the trace data, the highe
Session Id	Universally Unique ID (UUID)	A globally unique identifier to record the session. Filtering
Participant ID	Text	The participant number. Unique for each participant, so th
Timestamp	ISO8601 timestamp	The UTC timestamp user's prompt or AI response as text
Model	Text	The AI Model used. This is always gpt-4o-mini
Rag	Boolean	True is the treatment group T1 whereas False is the contro
Context	Text	The user selected context at the time of the user prompt o
Role	Text	Values "user" or "assistant" Indicator of whether the row v
Content	Text	In the case of Role == "user", the Content is the prompt.

Table 2: Fields in the D1 Chatbot Trace Data

The trace data could be coded in a variety of ways, such as based on the type of question asked, or the specifically of the prompt. Ultimately this trace data was used to classify users' interactions with the AI by session Id, then quantify those session-based interactions.

3.3 Data Analysis

3.3.1 Overview

[describe the process at a high level, details are below]

Categorical content analysis

Build a codebook

Take a sample of 50

Hand classify the 50

Evaluate which model does the most consistent job classifying compared to human.

Classify all 1024 observations

Summarize classifications so the unit of analysis is participant.

[old stuff]

There were 1024 participant sessions in the D1 chatbot trace data. Each session was coded using an LLM based on whether the session demonstrated a clear indication of task completion or activity where the participant sought to learn.

There was also a category for inconclusive sessions, where the session did not clearly demonstrate either. The categorical content analysis was performed based on deductive coding. Themes were identified among findings in academic literature in addition to my seven years of experience teaching IST256. The codebook outlined four key behaviors. The task completion category included activities such as superficial learning through answer seeking, and overreliance by cognitive offloading for task completion. The understanding category included scaffolding activities where the AI helps the learner, and questioning where participants ask questions to clarify a concept.

Since it was not practical to manually code all 1024 participant sessions, AI-assisted content analysis was used. The codebook was turned into a prompt and then

The final unit analysis was a dataset by participant with a count of sessions, count of sessions coded as task completion sessions and count of sessions coded as understanding.

Krippendorf's alpha for inter coder reliability among the three AI's used to classify the data.

3.3.2 Tools

What did you use to perform the analysis? Python libraries used, etc. . . .

3.3.3 Building The Codebook

To perform the categorical content analysis of the 1024 D1 chatbot trace data sessions, first a codebook was built deductively. An initial set of codes were established based on observations of other researchers in literature in combination with my seven years of experience teaching IST256. A sub-sample of 50 sessions were selected at random from the D1 dataset to identify initial classifications. I classified each session according to the codebook in *Table 3*. In addition, I summarized the classified activity as an example which could be incorporated into the AI prompt used by the LLM.

P articipant Behavior	**D efinition**	**Example**	**Classified Activity**	• *Supported Citations**
Seeking Answers (Superficial Learning)	The participant makes a direct ask of the AI to complete their assignment or lab. There is no co nversation.	"Can you code the solution for assignment ABC?" "How do I complete section XYZ in the lab?"	Task Completion	- Hassan Et Al 2025, - Fin ney-Ainsley 2022
O verreliance (Cognitive offloading for task completion)	Participant asks the AI to complete a task and does not further engage. Thus there is no clear intention to understand their work for them. Thus a deep connection to the material is not formed.	"Can you convert this algorithm into code?" "Can you provide an approach to start this as signment?" "Fix this error for me."	Task Completion	- Prather Et Al 2024, -Margulieux 2024
Scaffolding (AI as Expert)	AI provides support / expertise to the learner to a learner to help them achieve a task they couldn't complete on their own.	"What does this error mean?" "Can you help me locate the error in my code?" "Can you help me t roubleshoot my code?" .	Learning	- Prather 2019, - Hassan 2025, - Prather 2024 (weird)

P articipant Behavior	**D efinition**	**Example**	**Classified Activity**	• *Supported Citations**
Questioning (AI as Tutor)	Asking the AI to explain a concept or provide code examples of a concept.	Can you provide an example of a nested loop? Why would I use a nested IF verses elif? Can you help me understand di ctionaries? When would I use a list versus a dictionary?	Learning	- Fin ney-Ainsley 2022, - Cambaz and Zhang 2024
Unable to Determine	The transcript does not provide enough information to classify the par ticipant's behavior. Off topic question not about the subject matter.	"When is the exam?" "Can you tell me the weather? " What did I ask you last week?	I nconclusive	

Table 3: Codebook of participant behaviors

For example, in one of *Participant 119*'s sessions this was this only question asked the AI chatbot:

> PARTICIPANT_119: "Write a function, score_sentiment() which given 3 inputs: positive words, negative words and some text, will return an integer score of sentiment as output. Assume all inputs are lower case and do not have any punctuations."
> AI_ASSISTANT: (writes code and explains what it does as the bot

was designed)

The participant pasted in part of the assignment instructions asking the AI generate code. I classified this session as Task Completion, under the behavior Seeking Answers.

For another example, here is *Participant 107's* session. I classified this interaction as Learning under the behavior Questioning, as it is obvious the participant is seeking to understand Python concepts.

> PARTICIPANT_107: Why false?
> a=0.1
> b=0.2
> bool(a+b==0.3)
> AI_ASSISTANT: (provides an explanation with examples)
>
> PARTICIPANT_107: explain 7%2
> AI_ASSISTANT: (provides an explanation with examples)
>
> PARTICIPANT_107: explain == vs =
> AI_ASSISTANT: (provides an explanation with examples)

An AI prompt was formulated from the codebook and the various examples of interactions I observed within the observations. The AI prompt is included in Appendix C.

3.3.4 Justification of Model Selection for Classification

Using a Large-Language Model to classify the 1024 sessions raised two concerns. First was inter-coder reliability. I wanted the LLM classifier to match my classifications as closely as possible. Secondly was internal reliability, LLMs are prone to hallucinations so I had to ensure whatever model I chose was consistent across multiple classifications of the same data. The best model was the most similar to my classification and the most internally consistent.

Four different models were chosen as candidates: x-ai/grok-4-fast, openai/gpt-5-mini, google/gemini-2.5-flash, and anthropic/claude-sonnet-4. Each model candidate came from a different AI provider and were considered the best for classification type tasks at the time of the study.

Each of the four models classified the same 50 session sub-sample I classified initially. The same AI prompt from Appendix C was used across all four models. This procedure was repeated three times to establish a measure of consistency among model runs. Krippendorff's alpha was calculated to measure the model's agreement with my classifications and with itself across the three runs.

Model	Agreement with Human	Internal Consistency (3 runs)
x-ai/grok-4-fast	0.873131	0.885031
openai/gpt-5-mini	0.788552	0.914743
google/gemini-2.5-flash	0.833052	1.000000
anthropic/claude-sonnet-4	0.873131	1.000000

Table 4: The Krippendorff's Alpha of model vs human and internal consistency of model against itself.

As *table 4* indicates, x-ai/grok-4-fast and anthropic/claude-sonnet-4 classifications were in closest agreement to my classifications. Only google/gemini-2.5-flash, and anthropic/claude-sonnet-4 classified the data consistently across the three runs. For these reasons anthropic/claude-sonnet-4 was chosen as the model to complete the classification task.

Incidentally, the Krippendorff's alpha across all models + human for the three runs was 0.76108, 0.75115 and 0.77207.

3.3.5 Categorical Content Analysis

Next we run the classifier using anthropic/claude-sonnet-4 across the 1024 points of data. (show a bar chart of the categories).

Next as part of the categorical content analysis count each category so that the unit of analysis is at the participant level. This forms our independent variables.

Bujilding the dependent variables by counting up the categories. This is the independent variable. Unit of analysis is participant level so we count up the number of interactions that are "Task Completion" vs "Learning"

3.3.4 Addressing the Research Questions

Now explain the approach to answering the research questions. Multi-variate regressions....

4.0 Results

TODO

5.0 Summary

This page is intentionally blank.

References

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., & Henighan, T. (2020). *Language Models are Few-Shot Learners*.

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., & Henighan, T. (2020). *Language Models are Few-Shot Learners*.

Christensen, C. M., McDonald, R., Altman, E. J., & Palmer, J. E. (2018). Disruptive Innovation: An Intellectual History and Directions for Future Research. *Journal of Management Studies*, *55*(7), 1043–1078. https://doi.org/10.1111/joms.12349

Gillioz, A., Casas, J., Mugellini, E., & Khaled, O. A. (2020). *Overview of the Transformer-based Models for NLP Tasks*. 179–183. https://doi.org/10.15439/2020F20

Halevy, A., Norvig, P., & Pereira, F. (2009). The Unreasonable Effectiveness of Data. *IEEE Intelligent Systems*, *24*(2), 8–12. https://doi.org/10.1109/MIS.2009.36

Horn, M. B. (2024, June 3). *What does Disruptive Innovation Theory have to say about AI? - Christensen Institute*. https://www.christenseninstitute.org/blog/what-does-disruptive-innovation-say-about-ai/

Păvăloaia, V.-D., & Necula, S.-C. (2023). Artificial Intelligence as a Disruptive Technology—A Systematic Literature Review. *Electronics*, *12*(5), 1102. https://doi.org/10.3390/electronics12051102

Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*, *72*, 678–691. https://doi.org/10.1016/j.chb.2016.08.047

Shanahan, M. (2024). Talking about Large Language Models. *Communications of the ACM*, *67*(2), 68–79. https://doi.org/10.1145/3624724

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158. https://doi.org/10.1016/j.edurev.2017.09.003

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., & Le, Q. V. (2022). *Finetuned Language Models Are Zero-Shot Learners* (No. arXiv:2109.01652). arXiv. http://arxiv.org/abs/2109.01652

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., & Zhou, D. (2022). *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.*

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., … Wen, J.-R. (2023). *A Survey of Large Language Models* (No. arXiv:2303.18223). arXiv. http://arxiv.org/abs/2303.18223
stensen, C. M., McDonald, R., Altman, E. J., & Palmer, J. E. (2018). Disruptive Innovation: An Intellectual History and Directions for Future Research. *Journal of Management Studies*, *55*(7), 1043–1078. https://doi.org/10.1111/joms.12349