A Dataset and Benchmark for Automatically Answering and Generating Machine Learning Final Exams

Sarah Zhang
EECS
MIT
sazhang@mit.edu

Reece Shuttleworth
EECS
MIT
rshuttle@mit.edu

Derek Austin
CS
Columbia University
da2986@columbia.edu

Yann Hicke
CS
Cornell University
ylh8@cornell.edu

Leonard Tang
Mathematics
Harvard University
leonardtang@college.harvard.edu

Sathwik Karnik
EECS
MIT
skarnik@mit.edu

Darnell Granberry
EECS
MIT
darnellg@mit.edu

Iddo Drori
EECS and CS
MIT and Columbia University
idrori@mit.edu,idrori@cs.columbia.edu

Abstract

Can a machine learn machine learning? We propose to answer this question using the same criteria we use to answer a similar question: can a human learn machine learning? We automatically answer MIT final exams in Introduction to Machine Learning at a human level. The course is a large undergraduate class with around five hundred students each semester. Recently, program synthesis and few-shot learning solved university-level problem set questions in mathematics and STEM courses at a human level. In this work, we solve questions from final exams that differ from problem sets in several ways: the questions are longer, have multiple parts, are more complicated, and span a broader set of topics. We provide a new dataset and benchmark of questions from eight MIT Introduction to Machine Learning final exams between Fall 2017 and Spring 2022 and provide code for automatically answering these questions and generating new questions. We perform ablation studies comparing zero-shot learning with few-shot learning, chain-of-thought prompting, GPT-3 pre-trained on text and Codex fine-tuned on code on a range of machine learning topics and find that few-shot learning methods perform best. We make our data and code publicly available for the machine learning community.

1 Introduction

Can a machine learn machine learning? This work presents a new dataset of machine learning final exams with around 500 questions and a benchmark of baselines using transformers and their respective grade performance, demonstrating that the best baseline performs at a human level.

In university-level STEM courses, students complete assignments (including problem sets and labs) and exams throughout the course. Recent work has opened the door for a machine to solve course problem sets [4] using language models and few-shot learning. However, final exams remain

Preprint. Under review.

Figure 1: 500 MIT students taking the final exam in Introduction to Machine Learning in Fall 2021 (no student faces or identifying information appears in the image to maintain anonymity). The final exam is three hours long and is taken in an indoor track-and-field stadium that accommodates many students.

challenging, and this work is the first to present a structured dataset of machine learning finals and a benchmark of baseline methods for answering them. Final exams are different from problem sets because they serve as a benchmark of cumulative understanding of material learned over a semester and evaluate the students' depth and breadth of expertise. Further, questions on final exams are longer, have multiple parts, span a broader set of topics, and are more complicated and nuanced.

All the above holds for MIT's Introduction to Machine Learning (Intro to ML), a rigorous undergraduate course with around 500 students each semester, as shown in Figure 1, making it one of MIT's largest undergraduate courses offered.

Intro to ML is a core class in MIT's computer science program. The course description [1] is:

> "Introduction to Machine Learning introduces principles, algorithms, and applications of machine learning from the point of view of modeling and prediction; formulation of learning problems; representation, over-fitting, generalization; clustering, classification, probabilistic modeling; and methods such as support vector machines, hidden Markov models, and neural networks."

The prerequisites for the course are Python Programming and Multivariate Calculus, with Introduction to Algorithms and Linear Algebra recommended. The class typically consists of weekly exercises, labs, quizzes, homework, a midterm, and a final exam. There were no final exams in Fall 2020 and Spring 2020 due to COVID-19.

Intro to ML final exams differ from problem sets (psets) in several ways, and the experience of solving each varies. First, Intro to ML finals are long, containing around nine questions with seven parts. Final exam questions are also multifaceted and multi-stepped: different parts of a single question require applying different concepts and problem-solving skills, and parts may build upon each other. While weekly problem sets focus on a single topic, finals span topics from the entire semester. Further, Intro to ML final questions are often story-based problems that may require mathematical modeling. Due to the time constraint of these exams, finals are also designed to test core understanding and application of course material over rote calculations. Thus, asking a machine to answer questions from finals allows for testing whether the model is able to learn a breadth and depth of topics beyond problem sets.

In this work, we present a new dataset curated from the eight most recent final exams of MIT's Intro to ML course, totaling nearly 500 questions spanning twelve topics in machine learning: regression, classifiers, logistic regression, features, loss functions, neural networks, convolutional neural networks (CNNs), Markov decision processes (MDPs), recurrent neural networks (RNNs), reinforcement learning, clustering, and decision trees. Our dataset covers the finals given for the

[1]MIT Registrars Office Catalog for Course 6: Electrical Engineering and Computer Science, 6.3900 (6.036) Introduction to Machine Learning http://student.mit.edu/catalog/m6c.html

semesters Fall 2017, Spring 2018, Fall 2018, Spring 2019, Fall 2019, Spring 2021, Fall 2021, and Spring 2022. Due to the COVID-19 pandemic, there we no finals in the course during 2020.

Table 1: Spring 2021, Question 10.a: question part, solution, Codex zero-shot answer, program output, GPT-3 Zero-Shot answer, and GPT-3 Zero-Shot with CoT prompt answer.

<table>
<tr><td>Topic</td><td>Classifiers</td></tr>
<tr><td>Question Part</td><td>Given a set of data $\mathcal{D}_{\text{train}} = \{(x^i, y^i)\}$, a weighted nearest neighbor regressor has the form

$$h(x;\theta) = \frac{\sum_{(x^i,y^i)\in\mathcal{D}_{\text{train}}} f(x, x^i, \theta) y^i}{\sum_{(x^i,y^i)\in\mathcal{D}_{\text{train}}} f(x, x^i, \theta)}. \tag{1}$$

A typical choice for f is $f(x, x', \theta) = e^{-\theta\|x-x'\|^2}$ where θ is a scalar and $\|x - x'\|^2 = \sum_{j=1}^{d}(x_j - x'_j)^2$. Assume our training data $\mathcal{D}_{\text{train}} = ((1, 1), (2, 2), (3, 6))$. What is $h(10, 0)$? That is, letting $\theta = 0$, what is our prediction for $x = 10$?</td></tr>
<tr><td>Solution</td><td>3</td></tr>
<tr><td>Codex Zero-Shot Answer</td><td><pre><code>import numpy as np

def f(x, x_i, theta):
  return np.exp(-theta * np.linalg.norm(x - x_i)**2)

def h(x, theta):
  D = np.array([(1, 1), (2, 2), (3, 6)])
  numerator = 0
  denominator = 0
  for x_i, y_i in D:
    numerator += f(x, x_i, theta) * y_i
    denominator += f(x, x_i, theta)
  return numerator / denominator

print(h(10, 0))
</code></pre></td></tr>
<tr><td>Program Output</td><td>3.0</td></tr>
<tr><td>GPT-3 Zero-Shot Answer</td><td>$h(10;0) = \frac{f(10,1,0)y^1+f(10,2,0)y^2+f(10,3,0)y^3}{f(10,1,0)+f(10,2,0)+f(10,3,0)} = \frac{1\cdot 1+1\cdot 2+1\cdot 6}{1+1+1} = \frac{9}{3} = 3$</td></tr>
<tr><td>GPT-3 ZS with CoT Answer</td><td>$f(10,(1,1),0) = e^{-0\|10-(1,1)\|^2} = e^{-0\|9\|^2} = e^{-0\cdot 81} = 1$
$f(10,(2,2),0) = e^{-0\|10-(2,2)\|^2} = e^{-0\|8\|^2} = e^{-0\cdot 64} = 1$
$f(10,(3,6),0) = e^{-0\|10-(3,6)\|^2} = e^{-0\|7\|^2} = e^{-0\cdot 49} = 1$
$h(10,0) = \frac{\sum_{(x^i,y^i)\in\mathcal{D}_{\text{train}}} f(x,x^i,\theta)y^i}{\sum_{(x^i,y^i)\in\mathcal{D}_{\text{train}}} f(x,x^i,\theta)} = \frac{1\cdot 1+1\cdot 2+1\cdot 6}{1+1+1} = \frac{9}{3} = 3$</td></tr>
</table>

We prevent overfitting by including finals from 2021 and 2022 that are unavailable online. The Spring 2022 final exam was given after GPT-3 and Codex were last updated, verifying that the models are not trained on this data. The final exam questions have many parts, each posing a new problem, and each question in the dataset corresponds to one part. The questions in the finals are varied in topics and solution types. Solutions are multi-modal, with primarily open-ended questions and some true/false and multiple-choice questions on theory, math, and code implementations.

We make the dataset publicly available and welcome others to use it to aid in developing and assessing new language models and methods. Due to the diversity of Intro to ML final questions, our dataset uniquely assesses advanced problem-solving and reasoning skills in machine learning, math, and natural language processing. This dataset opens the door to achieving breakthroughs in machine learning performance in machine learning final exams.

Semester	Mean Human Grade	Mean Human Non-Image Grade	Mean Machine Non-Image Grade
Spring 2021	75.84	80.77	62.09
Fall 2021	74.38	60.88	58.94
Spring 2022	69.07	70.82	68.86
Mean	73.10	70.82	63.29

Table 2: Mean human and machine grades on Introduction to Machine Learning final exams by semester. Spring 2021, Fall 2021, and Spring 2022 final exams were unavailable online when GPT-3 and Codex were trained, ensuring that the model is not overfitting.

In addition to the dataset, we present a benchmark using several baseline methods. We apply zero-shot and few-shot learning to GPT-3 and Codex, adding chain-of-thought prompting for GPT-3. We find that few-shot learning methods perform best. As shown in Table 2 the best performing methods pass the final exams, and their grade is comparable with human grades of MIT students on the same machine learning finals evaluated by the same human graders.

Following our previous work [4] that showed that questions generated by these models are indistinguishable from human-written questions, we also generate new final exam questions.

1.1 Related Work

There is a conception that humans are generalists, whereas machines are specialists. However, large language models based on transformers such as GPT-3 [1], Gopher [7], and Palm [3], also called foundation models, are generalist learners. Specifically, in our setting, while humans care about the number of topics in an exam and therefore find finals more difficult than problem sets, foundation models effortlessly scale to many topics without re-training. Language models may be pre-trained on text and fine-tuned on specific datasets such as code [2] which allows generating programs from text.

There are several ways to improve the mathematical reasoning ability of language models: (1) using chain-of-thought (CoT) prompting [5, 11], (2) using the top-k ranking solutions [6] and merging them by voting [10] or least-to-most prompting [12], and (3) using program synthesis and few-shot learning to generate code that answers questions [4].

Solving university-level course questions is a relatively new endeavor. The first work to tackle university-level machine learning course problem set questions [9] used a transformer and GNN architecture and heavily relied on data augmentation. This resulted in overfitting and did not scale up to other types of questions or courses. Probability and statistics course problem-set questions have been answered [8] by probabilistic program synthesis with human performance. Problem-set questions from the core math courses [4] have been automatically solved using few-shot learning and program synthesis at a human level.

2 Dataset

We present a new dataset of 496 questions from the eight most recent final exams of MIT's Introduction to Machine Learning course. The dataset spans questions on the 12 machine learning topics covered in the course: (1) regression, (2) classifiers, (3) logistic regression, (4) features, (5) loss functions, (6) neural networks, (7) convolutional neural networks (CNNs), (8) Markov decision processes (MDPs), (9) recurrent neural networks (RNNs), (10) reinforcement learning, (11) clustering, and (12) decision trees. We make our data and code publicly available.[2]

The breakdown of questions, parts, points, and non-image points by each semester and topic are shown in Table 3 and Table 4. Each question in a final exam consists of multiple parts. Questions are written by providing set-up and context information first, followed by the question parts (which may come with additional information). Set-up and context information may contain (1) story elements (ex., character names, and motivations), (2) relevant definitions and equations, and (3) data points. We format questions in the dataset by concatenating the question context, any context or solutions

[2]Data and code: https://github.com/idrori/mlfinalsQ

Semester	Questions	Parts
Fall 2017	10	61
Spring 2018	9	42
Fall 2018	9	59
Spring 2019	9	58
Fall 2019	8	61
Spring 2021	12	70
Fall 2021	8	86
Spring 2022	9	59
Mean	9.25	62
Total	74	496

Table 3: The number of questions and parts in the final for each semester of Introduction to Machine Learning. Spring 2020 and Fall 2020 did not have final exams due to COVID-19.

Topic	Questions	Parts
Regression	6	58
Classifiers	10	60
Logistic Regression	1	8
Features	3.5	21
Neural Networks	12.5	76
Loss Functions	2	14
CNNs	8	63
MDPs	9	75
RNNs	7	33
Reinforcement Learning	8	50
Clustering	1	8
Decision Trees	6	32
Mean	6.73	45.09
Total	74	496

Table 4: The number of questions and parts in the final for each topic of Introduction to Machine Learning.

from prior parts of the question required for answering the part, and the part's context and question. We split the questions into their corresponding parts. Questions consist of English text, mathematical notation, and images. Mathematical notation is represented in the dataset by LaTeX and images by screenshots from pdfs files.

The types of question answers are diverse. A few are multiple-choice or true/false questions. Most are open-ended, for which the evaluation requires modeling the problem, mathematical manipulation, or code writing. Many questions require providing an explanation and justification.

To curate the dataset, we transcribe questions into text. The six final exams of Fall 2017, Spring 2018, Fall 2018, Spring 2019, Fall 2019, and Spring 2022 were available in pdf format, so we used mathpix.com to automatically extract transcriptions, including both text and LaTeX. The final exams of Spring 2021 and Fall 2021 were available in LaTeX. We extract questions and solutions for all parts of all types of questions, including those that rely on images.

We curated the five exams between 2017 and 2019 from publicly available pdf files. Spring 2020 and Fall 2020 do not have final exams due to COVID-19. The three exams between 2021 and 2022 were unavailable online; therefore, the model does not overfit their solutions. The aggregate average grades were available to the students and did not contain any personally identifiable information. Three duplicate questions were originally on the final exam of Fall 2017 (questions 1, 3, 6) and appear again in the final exam of Spring 2022.

Method	Prompt
GPT-3 Zero-Shot	<question>
GPT-3 Few-Shot	Q: <similar question> A: <similar question's answer> Q: <question> A:
GPT-3 Zero-Shot with CoT	Q: <question> A: Let's think step by step.
GPT-3 Few-Shot with CoT	Q: <similar question> A: <similar question's answer> Q: <question> A: Let's think step by step.
Codex Zero-Shot	""" Write a program that answers the following question: <question> """
Codex Few-Shot	""" Write a program that answers the following question: <similar question> """ <similar question's correct code> """ Write a program that answers the following question: <question> """

Table 5: Input prompt for each of six baseline methods (1) GPT-3 Zero-Shot, (2) GPT-3 Few-Shot, (3) GPT-3 Zero-Shot with CoT, (4) GPT-3 Few-Shot with CoT, (5) Codex Zero-Shot, and (6) Codex Few-Shot.

3 Benchmark

3.1 Baselines

We provide a benchmark by comparing six baselines for answering the final exam questions: (1) GPT-3 with zero-shot learning, (2) GPT-3 with few-shot learning, (3) GPT-3 with zero-shot learning and chain-of-thought (CoT) prompting, (4) GPT-3 with few-shot learning and chain-of-thought (CoT) prompting, (5) Codex with zero-shot learning, and (6) Codex with few-shot learning.

Table 5 shows the prompt used for each approach. GPT-3 zero-shot uses the question as-is, whereas GPT-3 zero-shot with CoT uses the suffix "Let's think step by step." after the question to encourage multi-step output. Codex zero-shot uses the prefix "Write a program that answers" before the question within Python comments denoted by triple quotes """ to encourage Codex to write code. GPT-3 few-shot finds the closest questions in the embedding space, measured by cosine similarity, and uses these questions and their corresponding answers before the new question as examples in the prompt. Codex few-shot finds the closest questions in the embedding space also as measured by cosine similarity and uses these questions and their corresponding code as examples.

For students, a good study technique is to use previous final exams to review and practice for their upcoming final. We model this method by few-shot learning using the question–answer pairs (for GPT-3) or question–code (for Codex) with the closest question embeddings from previous finals. We implement this by considering all the exam questions, marking each question by its semester and year, and using only previous semesters' questions for few-shot learning. The Fall 2017 and Spring 2022 contain three questions that are the same. We handle these duplicate questions the same way humans do by allowing few-shot learning in Spring 2022 based on successful Fall 2017 zero-shot answers. In the same way, humans benefit by practicing for an exam by doing previous exams. Since it is the first final exam, we do not perform few-shot learning on the Fall 2017 exam.

Semester	GPT-3 ZS	GPT-3 FS	GPT-3 ZS CoT	GPT-3 FS CoT	Codex ZS	Codex FS
Fall 2017	**38.21**	NA	22.86	NA	21.43	NA
Spring 2018	44.35	60.48	38.71	**70.97**	32.26	67.74
Fall 2018	51.99	52.18	61.63	**64.17**	49.78	54.00
Spring 2019	43.45	54.23	41.07	**58.81**	15.54	41.55
Fall 2019	53.17	**62.70**	28.97	56.35	25.40	59.13
Spring 2021	44.33	55.81	53.45	60.21	33.62	**62.09**
Fall 2021	58.94	**58.94**	50.35	54.90	18.11	42.00
Spring 2022	42.78	**68.86**	32.03	53.48	51.01	65.46

Table 6: We benchmark different baselines for each semester, excluding question parts that rely on images. We compare the average grade of GPT-3 with zero-shot (ZS), GPT-3 with few-shot (FS) learning, GPT-3 with ZS, and chain-of-thought (CoT) prompting, GPT-3 with FS and CoT prompting, Codex with ZS, and Codex with FS. Fall 2017 is the first semester, so few-shot learning results based on previous semesters are unavailable (NA). Spring 2020 and Fall 2020 did not have final exams due to COVID-19. Spring 2021, Fall 2021, and Spring 2022 final exams were unavailable online when GPT-3 and Codex were trained, ensuring that the model is not overfitting.

3.2 Grading

We grade the answers and aim to keep all factors equal in grading human and machine answers. Human and machine answers are graded based on the number of points allocated to each question part, giving full, partial, or no credit for each answer. We approximate partial credit by assigning half-credit. The course staff graded the student final exams, which included graduate TAs and instructors. Two of the same graduate TAs and the instructor that graded the student answers also graded the machine answers. The grading instructions are the same for grading student answers as grading machine answers.

3.3 Performance

Table 6 shows the machine grades by semester and Table 7 shows the machine grades by topic, excluding question parts that rely on images. We compare the average grade of GPT-3 with zero-shot (ZS), GPT-3 with ZS and chain-of-thought (CoT) prompting, GPT-3 with few-shot (FS) learning, GPT-3 with FS and CoT prompting, Codex with ZS, and Codex with FS. Fall 2017 is the first semester, so few-shot learning results based on previous semesters are unavailable (NA). Spring 2020 and Fall 2020 did not have final exams due to COVID-19. Spring 2021, Fall 2021, and Spring 2022 final exams were unavailable online when GPT-3 and Codex were trained, ensuring that the model is not overfitting content it has seen previously. The results consistently demonstrate that few-shot learning methods perform best across semesters and topics, as marked in bold.

3.4 Limitations

Our dataset consists of all question parts and their solutions, including images. However, our baseline methods do not handle questions that rely on an image containing the information required to solve the question since GPT-3 and Codex do not handle images. Tables 8 and 9 show the breakdown of the number of question parts and points of questions that do not rely on image information for answering the question. On average, 28.42% of the question parts, which make up 35.63% of the points in final exams, are questions that rely on image information. The points attributed to the non-image parts are tallied, recorded, and used to calculate non-image percentage grades.

3.5 Generating New Questions

We use the dataset of exam questions to generate new questions automatically. We use questions from our dataset as prompts to create new high-quality questions not present in our dataset. We create a list of various questions in our curated dataset and use the resulting list to prompt GPT-3 to create a new question. The supplementary material demonstrates the results of this process for each topic in the course. The Appendix consists of new generated questions and the closest question from our dataset

Topic	GPT-3 ZS	GPT-3 FS	GPT-3 ZS CoT	GPT-3 FS CoT	Codex ZS	Codex FS
Regression	31.71	50.00	25.61	40.85	40.24	**50.00**
Classifiers	38.18	46.21	26.28	42.35	18.88	**53.74**
Logistic Regression	50.00	60.00	77.50	**77.50**	55.00	70.00
Features	58.65	75.96	53.85	77.31	68.85	**81.54**
Loss Functions	NA	NA	NA	NA	NA	NA
Neural Networks	48.34	60.23	44.54	**68.42**	37.82	63.45
CNNs	37.50	**53.58**	28.36	47.81	13.38	36.77
MDPs	49.19	52.01	46.03	**54.23**	24.38	38.03
RNNs	61.46	**71.88**	57.29	66.14	12.50	40.63
Reinforcement Learning	36.09	42.99	36.67	**50.11**	28.79	45.11
Clustering	100.00	100.00	100.00	**100.00**	50.00	50.00
Decision Trees	54.70	**71.80**	32.48	51.28	46.15	54.70

Table 7: We benchmark different baselines for each course topic, excluding question parts that rely on images. We compare the grade of GPT-3 with zero-shot (ZS), GPT-3 with few-shot (FS) learning, GPT-3 with zero-shot and chain-of-thought (CoT) prompting, GPT-3 with FS and CoT, Codex with zero-shot, and Codex with few-shot learning. The question parts on loss functions rely on image information and are therefore unavailable (marked NA).

Semester	Non-Image Parts / All Parts	Non-Image Points / All Points
Fall 2017	49 / 61	70 / 100
Spring 2018	27 / 42	62 / 100
Fall 2018	29 / 59	62 / 100
Spring 2019	41 / 58	70 / 100
Fall 2019	47 / 61	63 / 100
Spring 2021	50 / 70	61 / 100
Fall 2021	56 / 86	48 / 100
Spring 2022	46 / 59	69 / 100
Total	345 / 496	505 / 800

Table 8: The number of question parts that do not rely on images and the number of points that do not rely on images in Introduction to Machine Learning finals for each semester. Spring 2020 and Fall 2020 did not have final exams due to COVID-19.

Topic	Non-Image Parts / All Parts	Non-Image Points / All Points
Regression	33 / 58	41 / 67
Classifiers	52 / 60	101 / 123
Logistic Regression	8 / 8	10 / 10
Features	18 / 21	26 / 38
Loss Functions	0 / 14	0 / 18
Neural Networks	56 / 76	87 / 128
CNNs	52 / 63	57 / 76
MDPs	38 / 72	39 / 112
RNNs	27 / 33	48 / 67
Reinforcement Learning	42 / 50	55 / 79
Clustering	1 / 8	2 / 12
Decision Trees	18 / 32	39 / 72
Total	345 / 496	505 / 800

Table 9: The number of questions parts, and the number of points that do not rely on images, and number of points that do not rely on images in Introduction to Machine Learning finals for each topic of the course.

as measured by the cosine similarity of the embedding of each question. These new questions are diverse and qualitatively similar to questions on previous MIT final exams. This provides an efficient way for course TAs and instructors to generate new final questions.

3.6 Implementation Details

We use the latest OpenAI GPT-3 and Codex models and do not re-train these very large language models. We fix all the hyperparameters of the models so that the answers are deterministic and reproducible. Specifically, we set both top P, which controls diversity, and sampling temperature, which controls randomness, to 0. The frequency and presence penalties are also set to 0, and we do not halt on any stop sequences. We allow diversity for generating new questions by setting the top P and temperature to 0.1. We run Codex with an upper bound of generating programs with 1024 tokens. We use the OpenAI text-davinci-002 and code-davinci-002 engines for generating text and programs. For few-shot-learning and question generation, we use the text-similarity-babbage-001 engine to embed the questions and find the closest questions in the dataset by cosine similarity. The running time for answering or generating each question part is a few seconds.

4 Conclusions

We present a dataset and benchmark for answering and generating university-level final exams in machine learning. The dataset is the first of its kind, and the benchmark compares state-of-the-art language model approaches. Machine performance and human performance are evaluated by the same graders and grading instructions. A comparison of baselines shows that few-shot learning methods perform best across semesters and topics. A limitation of our work is that our benchmark does not consider questions that rely on images for their solution. Potential negative societal impacts of the work may be applications that are used for helping answer finals during the exams. Potential positive societal impacts include improving students learning for final exams, helping course staff generate questions for finals, and comparing levels of difficulty of exams across semesters and schools. We hope this dataset and benchmark serve the machine learning community and advance the state-of-the-art in the field. Future work will extend the benchmark to handle questions that rely on images and extend the dataset to finals exams in many STEM courses and universities.

References

[1] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901, 2020.

[2] Mark Chen et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[3] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[4] Iddo Drori, Sarah Zhang, Reece Shuttleworth, Leonard Tang, Albert Lu, Elizabeth Ke, Kevin Liu, Linda Chen, Sunny Tran, Newman Cheng, Roman Wang, Nikhil Singh, Taylor L. Patti, Jayson Lynch, Avi Shporer, Nakul Verma, Eugene Wu, and Gilbert Strang. A neural network solves, explains, and generates university math problems by program synthesis and few-shot learning at human level. *arXiv preprint arXiv:2112.15594*, 2022.

[5] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

[6] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *arXiv preprint arXiv:2203.07814*, 2022.

[7] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[8] Leonard Tang, Elizabeth Ke, Nikhil Singh, Bo Feng, Derek Austin, Nakul Verma, and Iddo Drori. Solving probability and statistics problems by probabilistic program synthesis at human level and predicting solvability. In *Proceedings of the International Conference on Artificial Intelligence in Education (AIED)*, 2022.

[9] Sunny Tran, Pranav Krishna, Ishan Pakuwal, Prabhakar Kafle, Nikhil Singh, Jayson Lynch, and Iddo Drori. Solving machine learning problems. In *Proceedings of the Asian Conference on Machine Learning (ACML)*, pages 470–485, 2021.

[10] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[11] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[12] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes]
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 3.4.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 4.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See our repository
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 3.6.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 3.6.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See our repository
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See Section 2.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See Section 2.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

1. Submission introducing new datasets must include the following in the supplementary materials:
   (a) **Documentation and intended uses.** Documentation can be found in the README files of the GitHub repository https://github.com/idrori/mlfinalsQ. The authors intend for the dataset to be used by the machine learning and broader research community to improve
   (b) **URL to dataset.** The full dataset can be accessed and downloaded at https://github.com/idrori/mlfinalsQ.
   (c) **Author statement.** The authors bear all responsibility in case of violation of rights, etc., and confirm the data license.
   (d) **Hosting, licensing, and maintenance plan.** The data and code is hosted and maintained on GitHub under an MIT license.
2. To ensure accessibility, the supplementary materials for datasets must include the following:
   (a) **Links to access the dataset and its metadata.** The dataset and its metadata is accessible at https://github.com/idrori/mlfinalsQ.
   (b) **Reading the dataset.** Questions in the dataset are presented in json file format, with the following fields:

Field	Description
Semester	The semester the question's final was given in (ex. Fall 2017)
Question Number	The number of the question from the final (ex. 1, 2...)
Part Number	The question part label the question has (ex. a, b.i)
Points	The number of points the question is worth
Topic	The primary machine learning topic that the question targets Topics are regression, classifiers, logistic regression, features, loss functions, neural networks, CNNs, MDPs, RNNs, reinforcement learning, clustering, and decision trees
Type	Text if the question only relies on text, Image if the question relies on an image
Question	The original question text as presented from the source
Solution	The solution to the question

Table 10: Dataset json fields and their descriptions.

   The dataset is also available to download as a CSV file with the fields described in 10 as column headers.
   (c) **Long-term preservation.** We ensure the longevity of this dataset by keeping it publicly available in a GitHub repository.
   (d) **Explicit license.** The code and data is licensed under the MIT license in the repository.
   (e) **Structured Metadata.** `https://github.com/idrori/mlfinalsQ/blob/main/data/schema_dataset.json`
   (f) **Dataset Identification.** Our data and code are maintained in a GitHub repository, allowing for easy access, Our dataset thus does not have a DOI.
3. **Reproducibility.** Data, code, and evaluation procedures for reproducing the benchmark results in this paper are available at https://github.com/idrori/mlfinalsQ. The code provided allows Note that grading machine outputs were done manually, so none of the provided code will produce those.

B Appendix

Table 11: Generating new questions: example of a new question for each topic automatically generated and the closest question in the dataset based on cosine similarity of the questions embeddings.

Topic	Question	Similarity
Regression	Generated Question: "We're given a data set $D = \left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n}$, where $x^{(i)} \in R^d$ and $y^{(i)} \in R$. Let X be a $d \times n$ matrix in which the $x^{(i)}$ are the columns and let Y be a $1 \times n$ vector containing the values of $y^{(i)}$. Using the ordinary least-squares formula, we can compute $W_{ols} = \left(XX^T\right)^{-1} XY^T$ Using ridge regression, we can compute $W_{\text{ridge}} = \left(XX^T + \lambda I\right)^{-1} XY^T$ We decide to try to use these methods to initialize a single-unit neural network with a linear activation function. Assume that XX^T is neither singular nor equal to the identity matrix, and that neither W_{ols} nor W_{ridge} is equal to $(0, 0, \ldots, 0)$. Consider a neuron initialized with W_{ridge}. Provide an objective function $J(W)$ that depends on the data, such that batch gradient descent to minimize J will have no effect on the weights, or argue that one does not exist."	0.945
	Closest Question: "We're given a data set $D = \left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n}$, where $x^{(i)} \in R^d$ and $y^{(i)} \in R$. Let X be a $d \times n$ matrix in which the $x^{(i)}$ are the columns and let Y be a $1 \times n$ vector containing the values of $y^{(i)}$. Using the analytical regression (ordinary least-squares) formula, we can compute $$W_{ols} = \left(XX^T\right)^{-1} XY^T$$ Using ridge regression, we can compute $W_{\text{ridge}} = \left(XX^T + \lambda I\right)^{-1} XY^T$ We decide to try to use these methods to initialize a single-unit "neural network" with a linear activation function and no offset: $h(x; W) = W^T x$. Assume that XX^T is invertible and not equal to the identity matrix, and that neither W_{ols} nor W_{ridge} is equal to $(0, 0, \ldots, 0)$. Note also that we are not using an explicit offset/bias term. Rory has solved many problems from this particular domain before and the solution has typically been close to $W^* = (1, \ldots, 1)^T$. Define an objective function $J(W)$ that we could minimize in order to obtain good estimates for Rory's next problem, even with very little data."	

Table 11: Generating new questions: example of a new question for each topic automatically generated and the closest question in the dataset based on cosine similarity of the questions embeddings.

Topic	Question	Similarity
Classifiers	Generated Question: “Consider a binary classification problem with two classes, +1 and -1. Assume that the training data is linearly separable. Consider the following two models: Model 1: $g(x) = \text{sgn}(w^T x)$ Model 2: $g(x) = \text{sgn}(w^T x + b)$ Assume that the training data is linearly separable. Which model is more likely to overfit? Closest Question: “In some cases, we will have a validation set in addition to training and test sets. Assume the validation set is approximately the same size as the test set. This validation set is often used to tune hyperparameters such as λ. Imagine we have trained a classifier using regularization, with λ chosen based on performance on the training set. Which will have the highest accuracy the training set, the validation set or the test set?"	0.782
Logistic Regression	Generated Question: “In logistic regression, we use the sigmoid function to "squish" the output of our linear function to be between 0 and 1. What is the output of the sigmoid function when the input is 0?" Closest Question: “Consider the simplest of all neural networks, consisting of a single unit with a sigmoid activation function: $h(x; w = \sigma(w_0 + w_1 x)$ where $\sigma(z) = (1 + exp(-z))^{-1}$ Let’s start with a classifier defined by $w_0 = -1$ and $w_1 = 1$. Which range of input values x are classified as positive? Which as negative?"	0.810
Features	Generated Question: “You are working on a project to predict how successful a given Kickstarter campaign will be, based on information about the campaign (such as the amount of money pledged, the number of backers, the category of the project, the country where the project is based, and so forth). One of the features you want to use is the project’s title. What is a good way to encode the project’s title for use in a machine-learning predictor?" Closest Question: “Mac O’Larnin is considering selling an app on Frugal Play. You have a friend with inside info at Frugal, and they’re able to share data on how previous apps have performed on the store. Mac decides that he will learn a neural network with no hidden layer (i.e., consisting only of the output layer). He needs help in figuring out the precise formulation for machine learning. What is the best way to encode the app characteristic ’Suitable for people ages (2–4, 5–10, 11–15, 16 and over)’ as a feature for an input to the neural network? Choose from among the following: multiple unary features (one-hot encoding), multiple binary features (thermometer encoding), an integer or real-valued feature. Also give the exact function that maps each input to its corresponding feature(s)."	0.761

Table 11: Generating new questions: example of a new question for each topic automatically generated and the closest question in the dataset based on cosine similarity of the questions embeddings.

Topic	Question	Similarity
Loss Functions	Generated Question: “We have a data set of images, each of which is either a picture of a dog or a picture of a cat. We want to train a binary classifier to distinguish between the two classes. Which of the following is a better choice of loss function for this task: (a) The 0-1 loss, which gives a loss of 1 if the classifier predicts the wrong class and a loss of 0 otherwise. (b) The cross-entropy loss, which is given by -log(p) if the correct class is predicted and -log(1-p) otherwise, where p is the predicted probability of the correct class. (c) The hinge loss, which is given by max(0, 1-s) if the correct class is predicted and max(0, 1+s) otherwise, where s is the predicted score for the correct class." Closest Question: “Consider the following 2D dataset in (x,y) format: ((1,-1), +1), ((1,1), +1), ((1,2.5),+1), ((2,-2),-1), ((2,1),+1),((2,3),+1),((5,-1),-1),((5,-2),-1). Break ties in distance by choosing the point with smaller x_1 coordinate, and if still tied, by smaller x_2 coordinate. Compute the leave-one-out cross validation accuracy of the 3-nearest-neighbor learning algorithm on this dataset."	0.790
Neural Networks	Generated Question: “Consider a neural network with two hidden layers, each with two units. The input layer has two units, and the output layer has one unit. The hidden layers use the sigmoid activation function, and the output layer uses the linear activation function. The weights from the input layer to the first hidden layer are w11 = 1, w12 = 1, w21 = 1, and w22 = 1. The weights from the first hidden layer to the second hidden layer are w11 = 1, w12 = 1, w21 = 1, and w22 = 1. The weights from the second hidden layer to the output layer are w11 = 1, w21 = 1. The bias terms are all zero. What is the output of the neural network for the input x1 = 1, x2 = 1?" Closest Question: “A neural network is given as $Z^1 = X * W^1$, $A^1 = f1(Z^1)$, $Z^2 = W^2 * A^1$, $\hat{y} = f^2(Z^2)$. Specifically, the input X is a 4×1 column vector, $\hat{y}$ is a 1×1 scalar. W^2 is a 3×1 vector. We also know that, $Z^1 = (W^1)^T X$ and $Z^2 = (W^2)^T A^1$. What are the dimensions of Z^2?"	0.880
CNNs	Generated Question: “Suppose we have a 3x3 image and we use a 2x2 filter with stride 1. What are the dimensions of the output image?" Closest Question: “A neural network is given as $Z^1 = X * W^1$, $A^1 = f^1(Z^1)$, $Z^2 = W^2 * A^1$, $\hat{y} = f^2(Z^2)$. There is only one data point which is: $X = [1, 1, 1, 1]^T$ and $y = [1]$. If W^1 and W^2 are both matrices/vectors of all ones, what is the resulting Loss where the Loss = $(y - \hat{y})^2$?"	0.895

Table 11: Generating new questions: example of a new question for each topic automatically generated and the closest question in the dataset based on cosine similarity of the questions embeddings.

Topic	Question	Similarity
MDPs	Generated Question: "Consider an MDP with four states, called A, B, C, and D, and with two actions called Move and Stay. The discount factor $\gamma = 0.9$. Here is a reminder of the Q-learning update formula, based on experience tuple (s, a, r, s') : $$Q(s,a) := (1-\alpha)Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s', a')\right)$$ Let $\alpha = 1$. Assume we see the following state-action-reward sequence: A, Move, 0 B, Move, 0 C, Move, 1 A, Move, 0 B, Move, 0 C, Move, 1 With Q-values all starting at 0, we run the Q-learning algorithm on that state-action sequence. Provide the q-learning value for Q(C, Move)." Closest Question: "Consider an MDP with four states, called A, B, C, and D, and with two actions called Move and Stay. The discount factor $\gamma = 0.9$. Here is a reminder of the Q-learning update formula, based on experience tuple (s, a, r, s') : $$Q(s,a) := (1-\alpha)Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s', a')\right)$$ Let $\alpha = 1$. Assume we see the following state-action-reward sequence: A, Move, 0 B, Move, 0 C, Move, 1 A, Move, 0 With Q-values all starting at 0, we run the Q-learning algorithm on that state-action sequence. Provide the q-learning value for Q(A, move)."	0.988
RNNs	Generated Question: "Consider the following RNN: $s_t = \tanh(w_1 x_t + w_2 s_{t-1} + b)$, $y_t = w_3 s_t + b_2$. Assume $s_0 = 0$ and $b_2 = 0$. What values of w_1, w_2, w_3 and b would generate output sequence $[0, 0, 0, 1, 1, 1, 1]$ given input sequence $[0, 0, 0, 1, 0, 1, 0]$" Closest Question: "Ronnie makes a simple RNN with state dimension 1 and a *step* function for f_1, so that $s_t = \text{step}(w_1 x_t + w_2 s_{t-1} + b)$ where $\text{step}(z) = 1$ if $z > 0.0$ and equals 0 otherwise, and where the output $$y_t = s_t \ .$$ Assuming $s_0 = 1$, we want to generate output sequence $[0, 0, 0, 1, 1, 1, 1]$ given input sequence $[0, 0, 0, 1, 0, 1, 0]$. Rennie thinks this is not possible using Ronnie's architecture. Rennie makes an argument based on the relationships in the table above. Is Rennie right?"	0.907
Reinforcement Learning	Generated Question: "What is the tabular Q-learning update equation, based on experience tuple (s, a, r, s')?" Closest Question: "b is the bias for classifier network. What are dimensions of b for multi k-class classification?"	0.796

Table 11: Generating new questions: example of a new question for each topic automatically generated and the closest question in the dataset based on cosine similarity of the questions embeddings.

Topic	Question	Similarity
Clustering	Generated Question: "Suppose that we have a dataset with n data points, k clusters, and d features. After running the k-means algorithm, the within-cluster sum of squared errors (WCSS) is given by: $$\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i - \mu_{y_i}\|^2$$ where y_i is the cluster label of the ith data point, and μ_{y_i} is the cluster center associated with the ith data point. The within-cluster sum of squared errors (WCSS) is a measure of how well the clusters fit the data. Suppose that we have two datasets, X_1 and X_2, where X_1 has n_1 data points and X_2 has n_2 data points. We run the k-means algorithm on both datasets. We find that the WCSS for X_1 is smaller than the WCSS for X_2. Does this imply that the clusters for X_1 are better than the clusters for X_2? Why or why not?" Closest Question: "Consider the following 2D dataset in (x,y) format: ((1,-1), +1), ((1,1), +1), ((1,2.5),+1), ((2,-2),-1), ((2,1),+1),((2,3),+1),((5,-1),-1),((5,-2),-1). We will construct a tree using a greedy algorithm that recursively minimizes weighted average entropy. Recall that the weighted average entropy of a split into subsets A and B is: (fraction of points in $A) \cdot H\left(R_{j,s}^{A}\right) + ($ fraction of points in $B) \cdot H\left(R_{j,s}^{B}\right)$ where the entropy $H\left(R_m\right)$ of data in a region R_m is given by $H\left(R_m\right) = -\sum_k \hat{P}_{mk}\log_2 \hat{P}_{mk}$. The $\hat{P}_{mk}$ is the empirical probability, which is in this case the fraction of items in region m that are of class k. Some facts that might be useful to you: H(0) = 0, H(3/5) = 0.97, H(3/8) = 0.95, H(3/4) = 0.81, H(5/6) = 0.65, H(1) = 0. Draw the decision tree that would be constructed by our tree algorithm for this dataset. Clearly label the test in each node, which case (yes or no) each branch corresponds to, and the prediction that will be made at each leaf. Assume there is no pruning and that the algorithm runs until each leaf has only members of a single class."	0.767
Decision Trees	Generated Question: "The Gini score is a measure of how often a randomly chosen element would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. It ranges from 0 to 1, with 0 meaning that there are no mislabeled elements and 1 meaning that the subset is perfectly mixed. Explain whether or not it would be a good idea to use the Gini score as a scoring function for pruning decision trees." Closest Question: "There are different strategies for pruning decision trees. We assume that we grow a decision tree until there is one or a small number of elements in each leaf. Then, we prune by deleting individual leaves of the tree until the score of the tree starts to get worse. The question is how to score each possible pruning of the tree. Here is a definition of the score: The score is the percentage correct of the tree on a separate validation set. Explain whether or not it would be a good idea and give a reason why or why not."	0.867