Chameleon: Plug-and-Play Compositional Reasoning with Large Language Models

Pan Lu[1], Baolin Peng[2], Hao Cheng[2], Michel Galley[2], Kai-Wei Chang[1], Ying Nian Wu[1], Song-Chun Zhu[1], Jianfeng Gao[2]

[1]University of California, Los Angeles [2]Microsoft Research, Redmond

https://chameleon-llm.github.io

NEURAL INFORMATION PROCESSING SYSTEMS

UCLA Microsoft

Tool-Augmented LLMs

Hugging Face · Bing · OpenAI · GitHub · Bard AI · Google · OpenAI DALL·E 3 · python

How to **compose** these numerous tools to tackle complex tasks?

Tool-Augmented LLMs (LLM Agents)!

Chameleon on ScienceQA and TabMWP

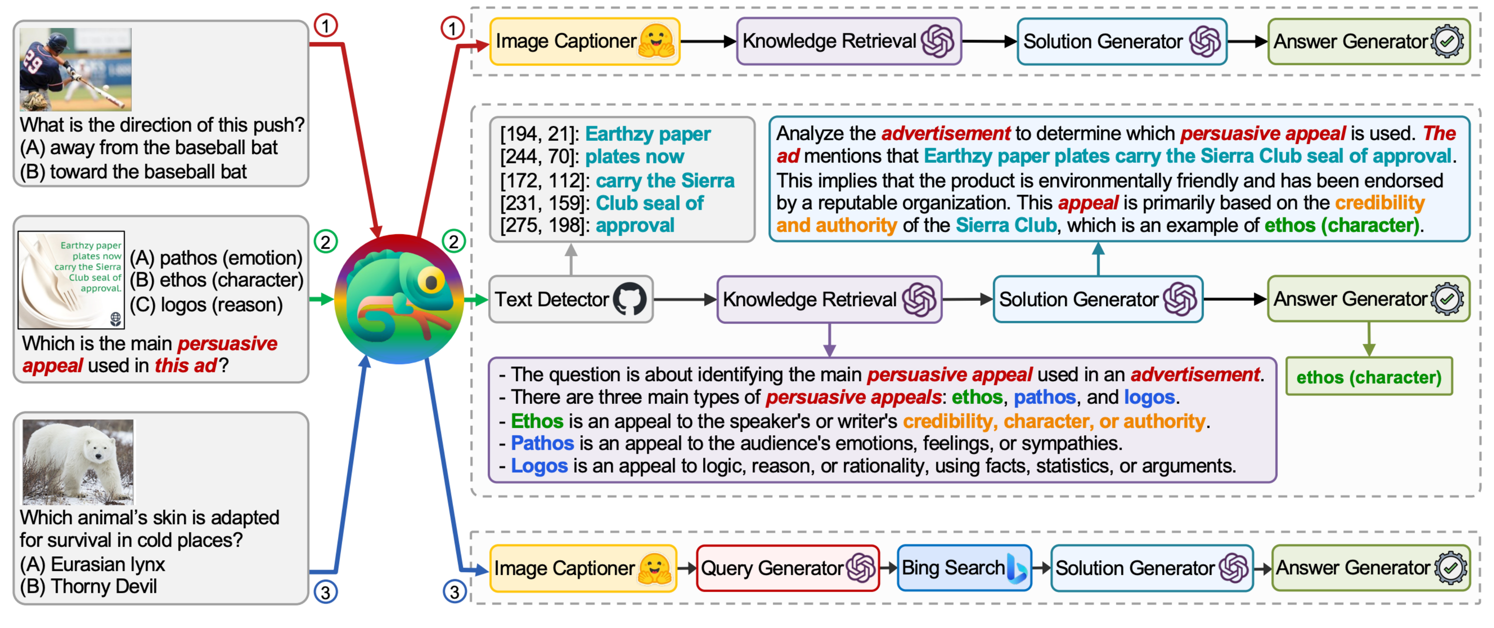


Figure 1: Examples from our Chameleon approach with GPT-4 on ScienceQA [28], a multi-modal question answering benchmark in scientific domains. Chameleon is adaptive to different queries by synthesizing programs to compose various tools and executing them sequentially to get final answers.

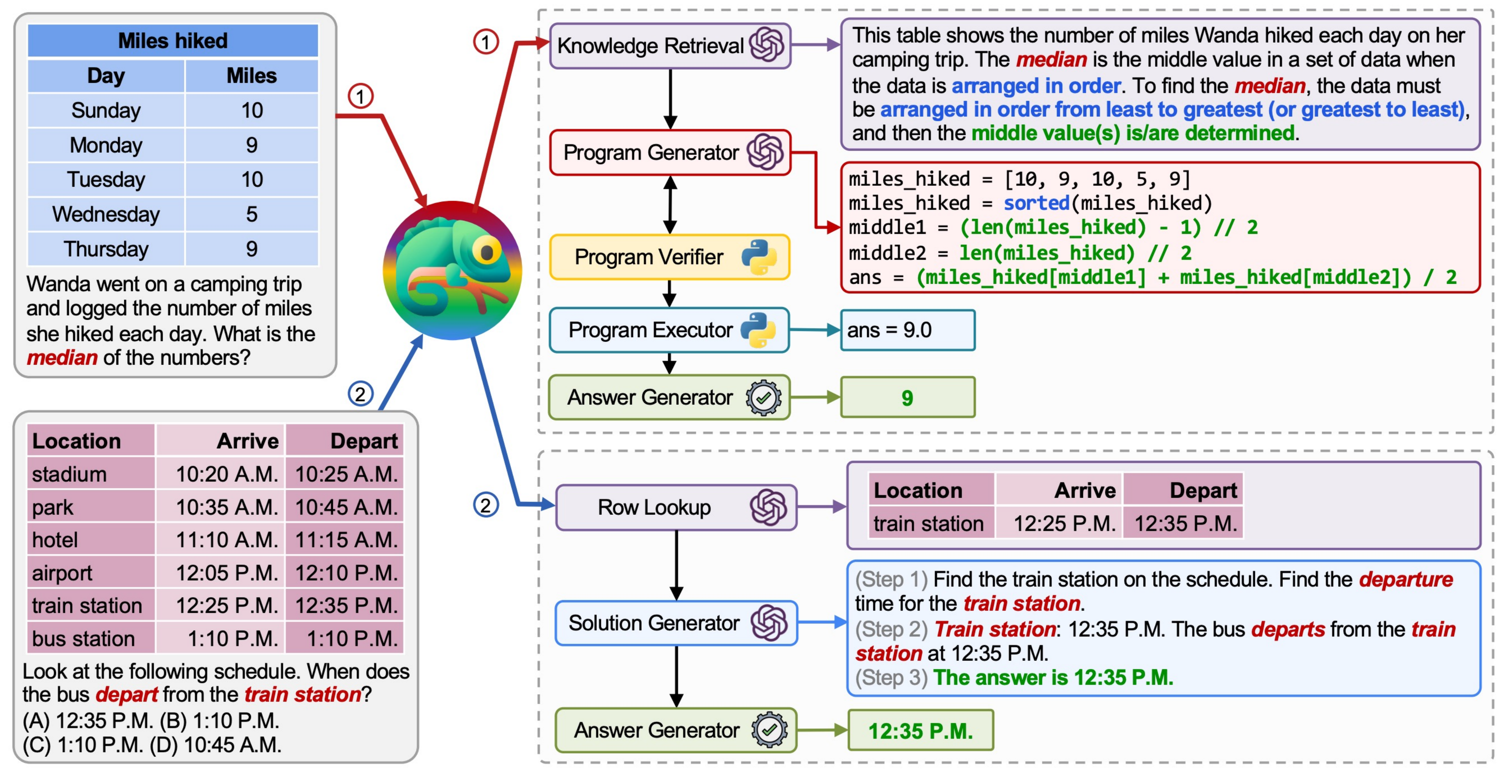
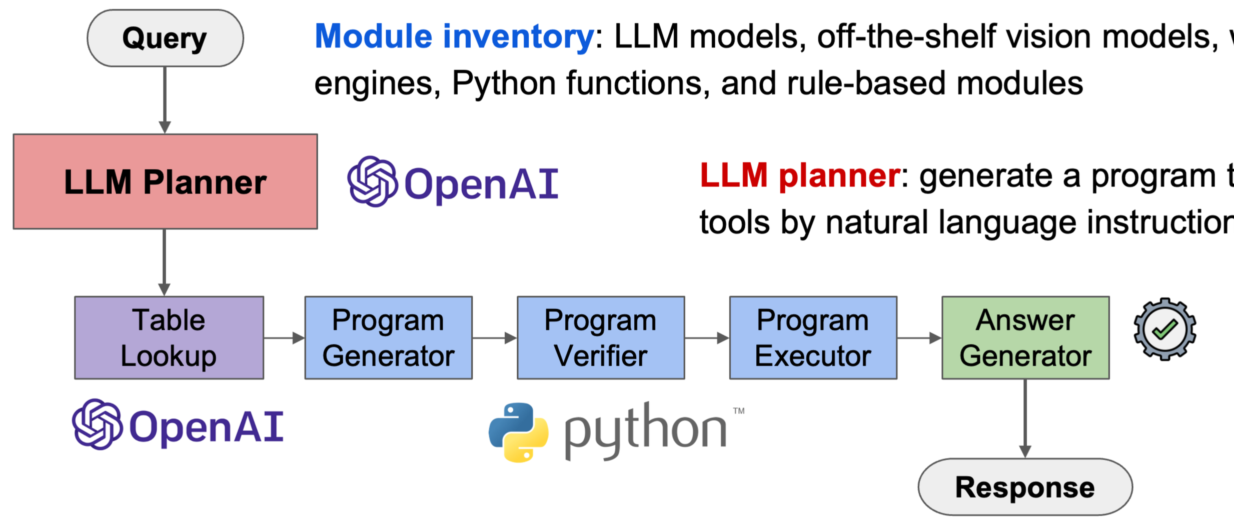


Figure 2: Two examples from our Chameleon approach with GPT-4 on TabMWP [29], a mathematical reasoning benchmark with tabular contexts. Chameleon demonstrates flexibility and efficiency in adapting to different queries that require various reasoning abilities.

Chameleon: LLM Planner



Module inventory: LLM models, off-the-shelf vision models, web search engines, Python functions, and rule-based modules

LLM planner: generate a program to compose tools by natural language instructions

Advantages

- Efficiently **extendable** to using new modules
- **Natural-language-like programs** are less error-prone, easy to debug, & user-friendly
- **Flexible to replace** the underlying LLM for the planner as well as each module

You need to act as a policy model, that given a question and a modular set, determines the sequence of modules that can be executed sequentially to solve the query.

The modules are defined as follows:

Query_Generator: This module generates a search engine query for the given question. Normally, we consider using "Query_Generator" when the question involves domain-specific knowledge.

Bing_Search: This module searches the web for relevant information to the question. Normally, we consider using "Bing_Search" when the question involves domain-specific knowledge.

Image_Captioner: This module generates a caption for the given image. Normally, we consider using "Image_Captioner" when the question involves the semantic understanding of the image, and the "has_image" field in the metadata is True.

Text_Detector: This module detects the text in the given image. Normally, we consider using "Text_Detector" when the question involves the unfolding of the text in the image, e.g., diagram, chart, table, map, etc., and the "has_image" field in the metadata is True.

Knowledge_Retrieval: This module retrieves background knowledge as the hint for the given question. Normally, we consider using "Knowledge_Retrieval" when the background knowledge is helpful to guide the solution.

Solution_Generator: This module generates a detailed solution to the question based on the information provided. Normally, "Solution_Generator" will incorporate the information from "Query_Generator", "Bing_Search", "Image_Captioner", "Text_Detector", and "Knowledge_Retrieval".

Answer_Generator: This module extracts the final answer in a short form from the solution or execution result.

Below are some examples that map the problem to the modules.

Question: Compare the average kinetic energies of the particles in each sample. Which sample has the higher temperature?

Context: The diagrams below show two pure samples of gas in identical closed, rigid containers. Each colored ball represents one gas particle. Both samples have the same number of particles.

Options: (A) neither; the samples have the same temperature (B) sample A (C) sample B

Metadata: 'pid': 19, 'has_image': True, 'grade': 8, 'subject': 'natural science', 'topic': 'physics', 'category': 'Particle motion and energy', 'skill': 'Identify how particle motion affects temperature and pressure'

Modules: ["Text_Detector", "Knowledge_Retrieval", "Solution_Generator", "Answer_Generator"]

Chameleon: Module Inventory

Provider	Modules
Hugging Face	Image Captioner
GitHub	Text Detector
OpenAI	Row Lookup, Column Lookup, Table Verbalizer
OpenAI	Query Generator, Knowledge Retrieval
Bing	Bing Search
python	Program Verifier, Program Executor
OpenAI	Program Generator
OpenAI	Solution Generator
(rule-based)	Answer Generator

Experiments on ScienceQA

Model	#Tuned Params	ALL	NAT	SOC	LAN	TXT	IMG	NO	G1-6	G7-12
Heuristic baselines										
Random Choice [28]	-	39.83	40.28	46.13	29.25	47.45	40.08	33.66	39.35	40.67
Human [28]	-	88.40	90.23	84.97	87.48	89.60	87.50	88.10	91.59	82.42
Fine-tuned models										
Patch-TRM [30]	90M	61.42	65.19	46.79	65.55	66.96	55.28	64.95	58.04	67.50
VisualBERT [23, 24]	111M	61.87	59.33	69.18	61.18	62.71	62.17	58.54	62.96	59.92
UnifiedQA [18]	223M	70.12	68.16	69.18	74.91	63.78	61.38	77.84	72.98	65.00
UnifiedQA CoT [28]	223M	74.11	71.00	76.04	78.91	66.42	66.53	81.81	77.06	68.82
MM-COT [60]	223M	84.91	87.52	77.17	85.82	87.88	82.90	86.83	84.65	85.37
MM-COT$_{Large}$ [60]	738M	91.68	95.91	82.00	90.82	95.26	88.80	92.89	92.44	90.31
LLaMA-Adapter$_T$ [59]	1.2M	78.31	79.00	73.79	80.55	78.30	70.35	83.14	79.77	75.68
LLaMA-Adapter [59]	1.8M	85.19	84.37	88.30	84.36	83.72	80.32	86.90	85.83	84.05
Few-shot GPT-3										
GPT-3 [3]	0M	74.04	75.04	66.59	78.00	74.24	65.74	79.58	76.36	69.87
GPT-3 CoT [28]	0M	75.17	75.44	70.87	78.09	74.68	67.43	79.93	78.23	69.68
Published results (Above) ▲										
Few-shot ChatGPT										
ChatGPT CoT	0M	78.31	78.82	70.98	83.18	77.37	67.92	86.13	80.72	74.03
Chameleon (ChatGPT)	0M	79.93	81.62	70.64	84.00	79.77	70.80	86.62	81.86	76.53
Few-shot GPT-4										
GPT-4 CoT	0M	83.99	85.48	72.44	90.27	82.65	71.49	92.89	86.66	79.04
Chameleon (GPT-4)	0M	**86.54**	**89.83**	**74.13**	**89.82**	**88.27**	**77.64**	**92.13**	**88.03**	**83.72**

Experiments on TabMWP

Model	#Tuned Params	ALL	FREE	MC	INT	DEC	EXTR	BOOL	OTH	G1-6	G7-8
Heuristic baselines											
Heuristic guess	-	15.29	6.71	39.81	8.37	0.26	30.80	51.22	26.67	17.55	12.27
Human performance	-	90.22	84.61	93.32	84.95	83.29	97.18	88.69	96.20	94.27	81.28
Fine-tuned models											
UnifiedQA$_{Base}$ [18]	223M	43.52	34.02	70.68	40.74	7.90	84.09	55.67	73.33	53.31	30.46
UnifiedQA$_{Large}$ [18]	738M	57.35	48.67	82.18	55.97	20.26	94.63	68.89	79.05	65.92	45.92
TAPEX$_{Base}$ [25]	139M	48.27	39.59	73.09	46.85	11.33	84.19	61.33	69.52	56.70	37.02
TAPEX$_{Large}$ [25]	406M	58.52	51.00	80.02	59.92	16.31	95.34	64.00	73.33	67.11	47.07
Zero-shot GPT-3											
GPT-3 [3]	0M	56.96	53.57	66.67	55.55	45.84	78.22	55.44	54.29	63.37	48.41
GPT-3 CoT [53]	0M	57.61	54.36	66.92	55.82	48.67	78.82	55.67	51.43	63.62	49.59
Few-shot GPT-3											
GPT-3 [3]	0M	57.13	54.69	64.11	58.36	40.40	75.95	52.41	53.02	63.10	49.16
GPT-3 CoT [53]	0M	62.92	60.76	69.09	60.04	63.58	76.49	61.19	67.30	68.62	55.31
GPT-3 CoT-PromptPG [29]	0M	68.23	66.17	74.11	64.12	74.16	76.19	72.81	65.71	71.20	64.27
Codex PoT* [5]	0M	73.2	-	-	-	-	-	-	-	-	-
Codex PoT-SC* [5]	0M	81.8	-	-	-	-	-	-	-	-	-
Published results (Above) ▲											
Few-shot ChatGPT											
ChatGPT CoT	0M	82.03	78.43	92.32	75.38	90.30	92.30	92.89	87.62	83.06	80.66
ChatGPT PoT	0M	89.49	90.24	87.35	89.31	93.82	92.10	85.89	55.24	90.60	88.00
Chameleon (ChatGPT)	0M	93.28	93.13	93.72	92.71	94.76	91.29	98.11	78.85	93.37	93.17
Few-shot GPT-4											
GPT-4 CoT	0M	90.81	88.48	97.49	86.16	97.51	96.86	99.11	89.52	92.40	88.70
GPT-4 PoT	0M	96.93	97.40	95.58	98.48	93.22	96.25	98.00	68.57	96.97	96.87
Chameleon (GPT-4)	0M	**98.78**	**98.95**	**98.29**	**99.34**	**97.42**	**98.58**	**98.56**	**93.33**	**98.95**	**98.54**

Tool Use Planning

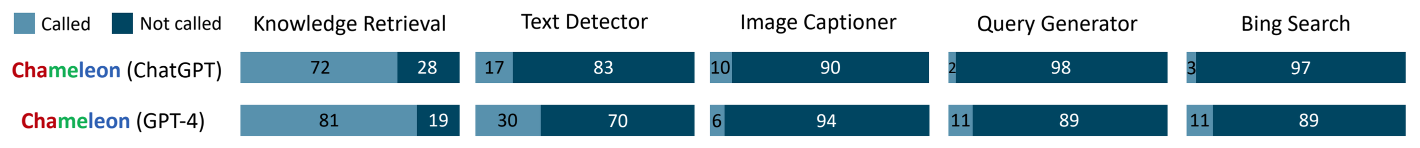


Figure 4: Tools called in the generated programs from Chameleon on ScienceQA.

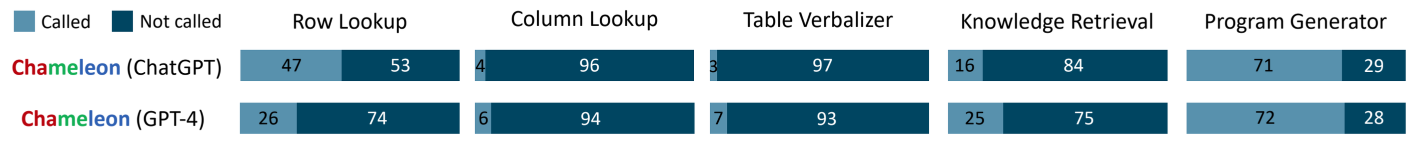
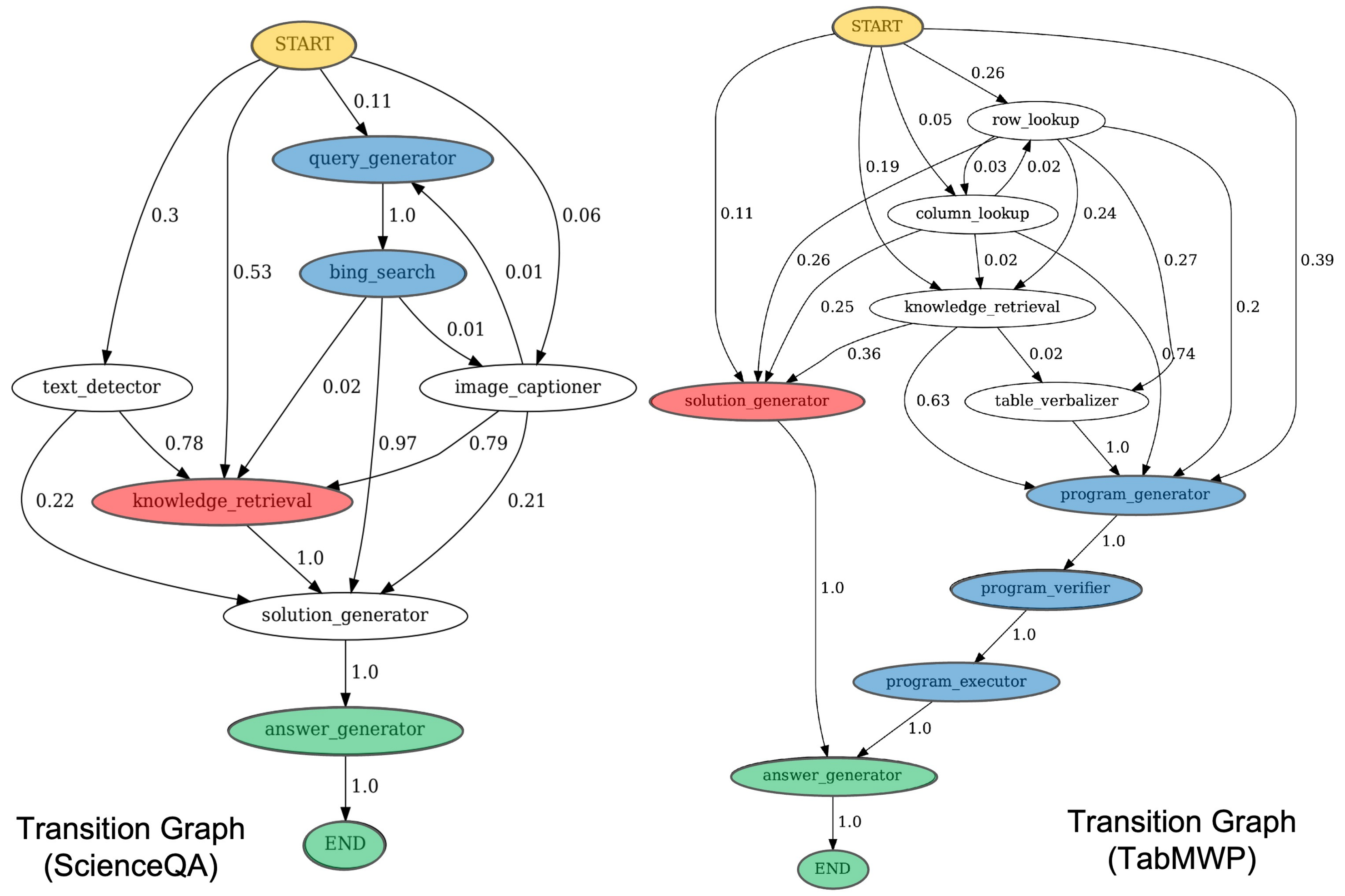


Figure 5: Tools called in the generated programs from Chameleon on TabMWP.

Transition Graph for Generated Programs

- Transition graphs of modules for generated programs by Chameleon (GPT-4)
- Transition probabilities are computed from tool transitions observed on the test sets
- GPT-4 planner **makes good decisions on how to sequence tools**



Failure Case

- Limitations of the current modules or suboptimal programs generated by the planner
- The module inventory may lack tools capable of addressing specific abilities

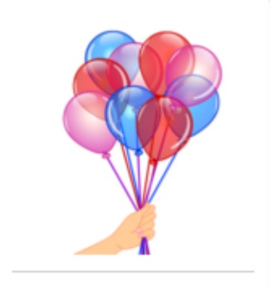
▷ *Query*

Question: Which property matches this object?

Context: Select the better answer.

Options: (A) stretchy (B) yellow

Metadata: 'pid': '10338', 'has_image': True, 'grade': 2, 'subject': 'natural science', 'topic': 'physics', 'category': 'Materials', 'skill': 'Identify properties of an object'



helium balloons

▷ *Response*

Generated Program: `["Image_Captioner","Solution_Generator","Answer_Generator"]`

Image Captioner:
A colorful kite is being flown in the sky.

Solution Generator:
Look at the image of the object, which is a colorful kite. A stretchy object can be stretched without breaking. Kites are typically made of materials that are not stretchy, such as nylon or polyester. A yellow object has a yellow color. The kite in the image is colorful and includes the color yellow. Therefore, the answer is B.

Answer Generator (Wrong prediction): (B) yellow

Correct Answer (Ground truth): (A) stretchy