Wisent Guard: A General Framework for Reliable Representation Identification and Representation Steering

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Representation engineering is a powerful method for identifying and modifying high-level concepts within the internal layers of large language models. Despite its potential, real-life deployments of activation steering remain difficult. We present Wisent-Guard, a flexible, open-source framework for monitoring and steering internal activations of large language models. Practical applications of the framework show 95 percent hallucination reduction, 25 percent improvement in coding ability and deep personalization capabilities.

1 Introduction

Large language models, with billions of parameters and Internet-scale training dataset, have displayed significant capabilities across a wide range of tasks, such as writing, coding or reasoning. However, their internal mechanisms of generating the next token cannot be precisely explained, with interactions between layers and parameters increasing in complexity as the size of these models increases.

Experiments with representation engineering (also known as steering or activation steering) have shown activation modification to be a powerful method of identifying and influencing high-level concepts (representations) within the layers of an LLM. Despite strong empirical performance on selected truthfulness, safety or personalization tasks, representation engineering methods lack a universal formulation and a unifying framework for understanding the underlying phenomenon, comparing methods and applying them to new problems.

We propose Wisent-Guard, a modular framework for analyzing the internal mechanisms within a large language model and influencing them to improve performance and individual alignment.

2 Representation Engineering Problem

We formulate the **Representation Engineering Problem** as the following:

For a given model M and a Representation

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

3 Representation Reading Functionalities

3.1 Classifier

3.2 Detection Handling Method

4 Representation Control Functionalities

References

[1] Stéphane Aroca-Ouellette, Cory Paik, Alessandro Roncone, and Katharina Kann. Prost: Physical reasoning about objects through space and time. 2021.

[2] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[3] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, 2013. Association for Computational Linguistics.

[4] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. *arXiv preprint arXiv:1911.11641*, 2019.

[5] Federico Cassano, John Gouwar, Daniel Nguyen, Sydney Nguyen, Luna Phipps-Costin, Donald Pinckney, Ming-Ho Yee, Yangtian Zi, Carolyn Jane Anderson, Molly Q. Feldman, Arjun Guha, Michael Greenberg, and Abhinav Jangda. Multipl-e: A scalable and extensible approach to benchmarking neural code generation. *arXiv preprint arXiv:2208.08227*, 2022.

[6] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. 2021.

[7] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

[8] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[9] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, John Schulman, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, and Jerry Tworek. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[10] Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. Xnli: Evaluating cross-lingual sentence representations. 2018.

[11] Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. Mutual: A dataset for multi-turn dialogue reasoning. 2020.

[12] Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*, 2021.

[13] Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. In *Proceedings of Sinn und Bedeutung 23*, volume 2, pages 107–124, Bellaterra (Cerdanyola del Vallès), 2019. Universitat Autònoma de Barcelona.

[14] William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. *Proceedings of the Third International Workshop on Paraphrasing (IWP 2005)*, 2005.

[15] Mingzhe Du, Anh Tuan Luu, Bin Ji, Liu Qian, and See-Kiong Ng. Mercury: A code efficiency benchmark for code llms. *arXiv preprint arXiv:2402.07844*, 2024.

[16] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019.

[17] Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with apps. *arXiv preprint arXiv:2105.09938*, 2021.

[18] Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, and Luke Zettlemoyer. Mapping language to code in programmatic context. *arXiv preprint arXiv:1808.09588*, 2018.

[19] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

[20] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *arXiv preprint arXiv:2009.13081*, 2020.

[21] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. 2019.

[22] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, 2017. Association for Computational Linguistics.

[23] Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. 2018.

[24] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*, 2017.

[25] Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Scott Wen-tau Yih, Daniel Fried, Sida Wang, and Tao Yu. Ds-1000: A natural and reliable benchmark for data science code generation. *arXiv preprint arXiv:2211.11501*, 2022.

[26] Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. *arXiv preprint arXiv:1105.4590*, 2011.

[27] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

[28] Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona T. Diab, Veselin Stoyanov, and Xian Li. Few-shot learning with multilingual language models. *arXiv preprint arXiv:2112.10668*, 2021.

[29] Hanmeng Liu, Jian Liu, Leyang Cui, Zhiyang Teng, Nan Duan, Ming Zhou, and Yue Zhang. Logiqa 2.0—an improved dataset for logical reasoning in natural language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[30] Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*, 2020.

[31] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. In *NeurIPS 2023 Datasets and Benchmarks Track*, 2023.

[32] Junnan Liu, Hongwei Liu, Linchen Xiao, Ziyi Wang, Kuikun Liu, Songyang Gao, Wenwei Zhang, Songyang Zhang, and Kai Chen. Are your llms capable of stable reasoning? *arXiv preprint arXiv:2412.13147*, 2024.

[33] Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin Clement, Dawn Drain, Daxin Jiang, Duyu Tang, Ge Li, Lidong Zhou, Linjun Shou, Long Zhou, Michele Tufano, Ming Gong, Ming Zhou, Nan Duan, Neel Sundaresan, Shao Kun Deng, Shengyu Fu, and Shujie Liu. Codexglue: A machine learning benchmark dataset for code understanding and generation. *arXiv preprint arXiv:2102.04664*, 2021.

[34] Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers. *arXiv preprint arXiv:2106.15772*, 2021.

[35] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

[36] Niklas Muennighoff, Qian Liu, Armel Zebaze, Qinkai Zheng, Binyuan Hui, Terry Yue Zhuo, Swayam Singh, Xiangru Tang, Leandro von Werra, and Shayne Longpre. Octopack: Instruction tuning code large language models. *arXiv preprint arXiv:2308.07124*, 2023.

[37] Anselmo Peñas, Eduard Hovy, Pamela Forner, Álvaro Rodrigo, Richard Sutcliffe, and Roser Morante. Qa4mre 2011–2013: Overview of question answering for machine reading evaluation. *CLEF 2013: Information Access Evaluation. Multilinguality, Multimodality, and Visualization*, 2013.

[38] Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, and et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

[39] Mohammad Taher Pilehvar and Jose Camacho-Collados. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. 2019.

[40] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[41] Siva Reddy, Danqi Chen, and Christopher D. Manning. Coqa: A conversational question answering challenge. *arXiv preprint arXiv:1808.07042*, 2019.

[42] Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford, CA, 2011.

[43] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.

[44] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*, 2019.

[45] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. 2013.

[46] Alexey Tikhonov, Mikhail Ryabinin, Yuri Kuratov, and Thomas Wolf. It's all in the heads: Using attention heads as a baseline for cross-lingual transfer in commonsense reasoning. *arXiv preprint arXiv:2106.12066*, 2021.

[47] David Vilares and Carlos Gómez-Rodríguez. Head-qa: A healthcare dataset for complex reasoning. 2019.

[48] Shiqi Wang, Zheng Li, Haifeng Qian, Chenghao Yang, Zijian Wang, Mingyue Shang, Varun Kumar, Samson Tan, Baishakhi Ray, Parminder Bhatia, Ramesh Nallapati, Murali Krishna Ramanathan, Dan Roth, and Bing Xiang. Recode: Robustness evaluation of code generation models. *arXiv preprint arXiv:2212.10264*, 2022.

[49] Yiming Wang, Pei Zhang, Jialong Tang, Haoran Wei, Baosong Yang, Rui Wang, Chenshu Sun, Feitong Sun, Jiran Zhang, Junxuan Wu, Qiqian Cang, Yichang Zhang, Fei Huang, Junyang Lin, Fei Huang, and Jingren Zhou. Polymath: Evaluating mathematical reasoning in multilingual contexts. *arXiv preprint arXiv:2504.18428*, 2025.

[50] Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. *arXiv preprint arXiv:1707.06209*, 2017.

[51] Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. Paws-x: A cross-lingual adversarial dataset for paraphrase identification. 2019.

[52] Pengcheng Yin, Bowen Deng, Edgar Chen, Bogdan Vasilescu, and Graham Neubig. Learning to mine aligned code and natural language pairs from stack overflow. In *Proceedings of the 15th IEEE/ACM International Conference on Mining Software Repositories (MSR)*, pages 476–486. IEEE, 2018.

[53] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium, 2018. Association for Computational Linguistics.

[54] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

[55] Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*, 2018.

[56] Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. 2019.

A Wisent Guard Primitives

A.1 Model

A.2 Contrastive Pair

A.3 Activations

A.4 Activation Collection Method

A.5 Additional Utilities

B Representation Reading Functionalities

B.1 Classifier

B.2 Detection Handling Method

C Representation Control Functionalities

D Ablation

A All supported benchmarks

This section enumerates all benchmarks used in our study, the task traits, the evaluation protocol, and the contrastive pair generation method applied to produce minimally perturbed negative targets. We first merged the *coding* and *mathematics* benchmark lists you provided and then appended them to the original master list.

Contrastive pair generation methods (definitions)

Reading Comprehension Abstention Swap [RC-Abstain] For extractive/open-domain RC: positive is the gold span; negative is an abstention (e.g., "Not provided in the text."). If gold is *No answer*, the negative is a confident but wrong claim.

Conversational Reading Comprehension Abstention [ConvRC-Abstain] As RC-Abstain, but with dialogue context (CoQA). Negatives are generic abstentions; yes/no items are flipped when applicable.

Language Modeling Corrupted Continuation [LM-CorruptCont] Language modeling: positive is the true continuation; negative is a corrupted continuation (local shuffles/randomization) to break coherence.

Text Generation Corruption [TG-Corrupt] Text generation: positive is true continuation, negative is shuffling postive if it is string and adding letter if shuffle results in identity, +1 if positive is number.

Two-Choice Flip [2C-Flip] Two-option tasks (PIQA, COPA, WinoGrande, CB): negative is simply the other option.

Multichoice First Distractor [MC-FirstDistr] Multi-choice tasks: If one list is provided then negative is the first incorrect option that comes after correct option, if the correct option is last, use the first option. If separate list with incorrect answers is provided, take first from the list. (deterministic).

Multichoice Random Distractor [MC-RandDistr] Multi-choice tasks: negative is a randomly chosen incorrect option from the same set.

Exact Match Partial Mask [EM-PartialMask] Exact-match free-form answers (HLE-EM): negative is the gold text with partial token masking (approximately 1/3 words, or partial masking for single-word answers).

Keyword-Preserving Token Deletion [KP-Del] Coding tasks: negative program created by deleting non-keyword tokens while preserving syntax-critical keywords; aims to remain plausible but fail unit tests.

Numeric Offset (+1) Perturbation [Num+1] Negative is the correct numeric answer offset by a small integer (typically +1); for non-integer answers, apply the minimal unit offset.

Summary Content-Polarity Flip [Summ-PolFlip] Code to text summarization: make a negative description by flipping key action words with simple opposites or adding "not" (e.g., "return" to "does not return", "add" to "remove"), while keeping the rest of the sentence the same.

Library Specific Flip [Lib-Spec-Filip] Coding tasks: negative program created by flipping functions, parameters (e.g. for numpy flip axis 0 to 1, for pandas flip mean() to sum()).

Logic inversion [Log-Inv] Coding tasks: negative program created by fliping bools, operators in code (e.g. return True to return False, <= to >=).

Offset (+-1) [+-1] Coding tasks: negative program created by adding/subtracting 1 from range or numeric value.

Replace empty [Empty] Coding tasks: negative program created by replacing string to empty string, list to empty list.

Generic incorrect continuation [Gen-Inc-Cont] Answer generation tasks: negative is created by generic incorrect answer.

Early return [Return] Coding tasks: negative program created by early return.

Evaluation types (definitions)

Log-likelihood option scoring [LL] The model scores each provided option/target by conditional log-probability given the prompt. Metrics typically compute accuracy over the highest-likelihood choice (MC tasks) or compare likelihoods of gold vs. negative targets.

Text generation string matching [TG] The model generates free-form text (or a number), which is then judged by task-specific metrics (e.g., exact match on numerical value for GSM8K/MATH; span/string matching for RC tasks; structured checks for DROP). Used also for CoT/generative GPQA variants and HLE-Exact-Match.

Perplexity (language modeling) [PPL] The model's next-token distribution is evaluated over a reference text to compute Perplexity (lower is better). Used for language-modeling corpora like WikiText.

Code execution against unit tests [CE] The model generates code, which is executed in a sandbox against unit tests provided by a dataset (e.g., pass@1). Applies to HumanEval/MBPP/APPS, MultiPL-E, DS-1000, LiveCodeBench, etc.

Table 1: Benchmarks (short names), evaluation abbreviations, contrastive method (short), and traits. Versions merged where applicable.

Benchmark	**Eval**	**Method [CM]**	**Traits**
DROP [16]	[TG]	**TG-Corrupt**	reading comprehension
ReCoRD [55]	[TG]	**MC-FirstDistr**	reading comprehension
SQuAD2 [40]	[TG]	**RC-Abstain**	reading comprehension
WebQuestions [3]	[TG]	**TG-Corrupt**	factual QA
TriviaQA [22]	[TG]	**TG-Corrupt**	factual QA
CoQA [41]	[TG]	**TG-Corrupt**	conversational RC
BoolQ [7]	[LL]	**2C-Flip**	boolean RC
Race [24]	[LL]	**MC-FirstDistr**	reading comprehension
QA4MRE [37]	[LL]	**MC-FirstDistr**	reading comprehension
QASPER [12]	[TG]	**2C-Flip**	scientific QA
MultiRC [23]	[LL]	**2C-Flip**	multi-sentence reasoning
XStoryCloze [28]	[LL]	**MC-FirstDistr**	commonsense
LogiQA [30]	[LL]	**MC-FirstDistr**	logical reasoning
LogiQA2 [29]	[LL]	**MC-FirstDistr**	logical reasoning
WSC [26]	[LL]	**2C-Flip**	reading comprehension
RTE	[LL]	**2C-Flip**	reading comprehension

Benchmark	Eval	Method [CM]	Traits
XWinograd [46]	[LL]	**MC-FirstDistr**	commonsense
WinoGrande [43]	[LL]	**2C-Flip**	commonsense
PIQA [4]	[LL]	**2C-Flip**	commonsense
COPA [42]	[LL]	**2C-Flip**	causal reasoning
HellaSwag [54]	[LL]	**MC-FirstDistr**	commonsense
SWAG [53]	[LL]	**MC-FirstDistr**	commonsense
OpenBookQA [35]	[LL]	**MC-FirstDistr**	science MCQ
ARC Easy [8]	[LL]	**MC-FirstDistr**	science reasoning
ARC Challenge [8]	[LL]	**MC-FirstDistr**	science reasoning
MC-TACO [56]	[LL]	**2C-Flip**	temporal reasoning
Social IQA [44]	[LL]	**MC-FirstDistr**	social reasoning
PROST [1]	[LL]	**MC-FirstDistr**	physical reasoning
Mutual [11]	[LL]	**MC-FirstDistr**	dialogue reasoning
HLE [38]	[TG]/[LL]	**EM-PartialMask; MC-FirstDistr**	expert exams
HLE Exact Match [38]	[TG]	**EM-PartialMask**	expert exams
HLE Multiple Choice [38]	[LL]	**MC-FirstDistr**	expert exams
TruthfulQA MC1 [27]	[LL]	**MC-FirstDistr**	truthfulness
TruthfulQA MC2 [27]	[LL]	**MC-FirstDistr**	truthfulness
TruthfulQA Gen [27]	[TG]	**MC-FirstDistr**	truthfulness
PubMedQA [21]	[LL]		biomedical QA
SciQ [50]	[LL]	**MC-FirstDistr**	science MCQ
HeadQA [47]	[LL]	**MC-FirstDistr**	healthcare QA
MedQA [20]	[LL]	**MC-FirstDistr**	medical QA
GSM8K [9]	[TG]	**Num+1**	mathematics
ASDiv [34]	[TG]	**Num+1**	mathematics
Arithmetic 1dc	[TG]	**Num+1**	mathematics
Arithmetic 2da	[TG]	**Num+1**	mathematics
MATH–500	[TG]	**Num+1**	mathematics (contest)
AIME	[TG]	**Num+1**	mathematics (contest)
AIME2024	[TG]	**Num+1**	mathematics (contest)
AIME2025	[TG]	**Num+1**	mathematics (contest)
HMMT	[TG]	**Num+1**	mathematics (contest)
HMMT Feb 2025	[TG]	**Num+1**	mathematics (contest)
PolyMath [49]	[TG]	**Num+1**	multilingual mathematics
Polymath EN Medium [49]	[TG]	**Num+1**	mathematics (olympiad)
Polymath ZH Medium [49]	[TG]	**Num+1**	mathematics (olympiad)
Polymath EN High [49]	[TG]	**Num+1**	mathematics (olympiad)
Polymath ZH High [49]	[TG]	**Num+1**	mathematics (olympiad)
LiveMathBench [32]	[TG]	**Num+1**	mathematics
LiveMathBench CNMO EN [32]	[TG]	**Num+1**	mathematics
LiveMathBench CNMO ZH [32]	[TG]	**Num+1**	mathematics
MBPP [2]	[CE]	**+-1; Empty; Return**	coding (Python)
MBPP+ [31]	[CE]	**+-1; Empty; Return**	coding (Python)
HumanEval [6]	[CE]	**Log-Inv; +-1**	coding (Python)
HumanEval+ [31]	[CE]	**Log-Inv; +-1**	coding (Python)
HumanEvalPack [36]	[CE]	**Log-Inv; +-1**	coding (multi-language)
InstructHumanEval	[CE]	**Log-Inv; +-1**	coding (Python)
CoNaLa [52]	[CE]	**KP-Del**	coding (Python)
CONCODE [18]	[CE]	**KP-Del**	coding (Java)

Benchmark	Eval	Method [CM]	Traits
Mercury [15]	[CE]	**Log-Inv; +-1**	coding (multi-language)
APPS [17]	[CE]	**KP-Del**	coding (Python)
DS–1000 [25]	[CE]	**Lib-Spec-Flip**	coding (Python)
ReCode [48]	[CE]	**Log-Inv; +-1**	coding (Python)
LiveCodeBench [19]	[CE]	**KP-Del**	coding (Python)
Multiple CPP [5]	[CE]		coding (C++)
Multiple Go [5]	[CE]		coding (Go)
Multiple Java [5]	[CE]		coding (Java)
Multiple JS [5]	[CE]		coding (JavaScript)
Multiple PY [5]	[CE]		coding (Python)
Multiple RS [5]	[CE]		coding (Rust)
CodeXGLUE Code to Text Python [33]	[TG]	**Summ-PolFlip**	coding (code-to-text)
CodeXGLUE Code to Text Go [33]	[TG]	**Summ-PolFlip**	coding (code-to-text)
CodeXGLUE Code to Text Java [33]	[TG]	**Summ-PolFlip**	coding (code-to-text)
CodeXGLUE Code to Text JavaScript [33]	[TG]	**Summ-PolFlip**	coding (code-to-text)
CodeXGLUE Code to Text PHP [33]	[TG]	**Summ-PolFlip**	coding (code-to-text)
CodeXGLUE Code to Text Ruby [33]	[TG]	**Summ-PolFlip**	coding (code-to-text)
CB [13]	[LL]	**MC-FirstDistr**	NLI
MRPC [14]	[LL]	**2C-Flip**	paraphrase detection
QNLI	[LL]	**2C-Flip**	NLI
QQP	[LL]	**2C-Flip**	paraphrase detection
SST2 [45]	[LL]	**2C-Flip**	sentiment analysis
WNLI	[LL]	**2C-Flip**	NLI
WiC [39]	[LL]	**2C-Flip**	word-in-context
PAWS-X [51]	[LL]	**2C-Flip**	paraphrase detection
XNLI [10]	[LL]	**MC-FirstDistr**	NLI

Category legend

- RC/ODQA
- Multi-choice Reasoning
- Exams & Knowledge Tests
- Mathematics
- Coding
- Other

Abbreviation legend

[LL]	Log-likelihood option scoring
[TG]	Text generation (string match)
[PPL]	Perplexity (LM)
[CE]	Code execution vs. unit tests

Method [CM] codes

RC-Abstain	RC abstention swap
ConvRC-Abstain	Conversational RC abstention
LM-CorruptCont	LM corrupted continuation
2C-Flip	Two-choice flip
MC-FirstDistr	First distractor (MC)
MC-RandDistr	Random distractor (MC)
MC-LetterSwap	Letter swap (MC)
Bool-Flip	Boolean flip
EM-PartialMask	Exact-match partial mask
KP-Del	Keyword-preserving deletion
Summ-WordDrop	Summary word drop
Num+1	Numeric offset (+1)

B GSM8K Pipeline Visualization

1. Load task using lm-eval-harness

question: Janet’s ducks lay 16 eggs per day. She eats 3 for breakfast and bakes 4 into muffins. She sells the remainder for $2 per egg. How much does she make daily?

answer: “Janet has 16 - 3 - 4 = 9 eggs left to sell. She makes 9 × $2 = $18 per day. 18”

↓

2. Split Data

Partition the dataset into training and testing subsets based on the ratio provided by the user.

↓

3. Extract contrastive pairs, Num+1 method

Positive prompt: “Janet’s ducks lay 16 eggs per day. She eats 3 for breakfast and bakes 4 into muffins. She sells the remainder for $2 per egg. How much does she make daily? 18”

Negative prompt: “Janet’s ducks lay 16 eggs per day. She eats 3 for breakfast and bakes 4 into muffins. She sells the remainder for $2 per egg. How much does she make daily? 19.0”

↓

4. Collect activations

Get activations for positive and negative prompts.

↓

5. Prepare data for training classifier

The dataset contains features derived from activations by averaging them across the sequence, and binary labels (0 = truthful, 1 = untruthful).

↓

6. Train classifier

Fit a logistic regression model using the data defined above.

↓

Ground truth and classifier evaluation

Ground truth evaluation usses lm-eval-harness to measure the model’s actual performance by comparing generated responses against correct answers. Classifier evaluation tests how accurately the trained classifier can distinguish between truthful and untruthful responses based solely on internal activations, validating the effectiveness of our truthfulness detection system.

Figure: GSM8K evaluation pipeline showing data flow from task loading through dual evaluation.