Feature-Level Insights into Artificial Text Detection with Sparse Autoencoders

Anonymous ACL submission

Abstract

Artificial Text Detection (ATD) is becoming increasingly important with the rise of advanced Large Language Models (LLMs). Despite numerous efforts, no single algorithm performs consistently well across different types of unseen text or guarantees effective generalization to new LLMs. Interpretability plays a crucial role in achieving this goal. In this study, we enhance ATD interpretability by using Sparse Autoencoders (SAE) to extract features from Gemma-2-2b's residual stream. We identify both interpretable and efficient features, analyzing their semantics and relevance through domain- and model-specific statistics, a steering approach, and manual or LLM-based interpretation. Our methods offer valuable insights into how texts from various models differ from human-written content. We show that modern LLMs have a distinct writing style, especially in information-dense domains, even though they can produce human-like outputs with personalized prompts.

1 Introduction

The active development of large language models (LLMs) has led to the increasing presence of AI-generated text in various domains, including news, education, and scientific literature. Although these models have demonstrated impressive fluency and coherence, concerns about misinformation, plagiarism, and AI-generated disinformation have required the development of reliable artificial text detection (ATD) systems (Abdali et al., 2024). Existing ATD frameworks primarily rely on statistical measures, linguistic heuristics, and deep learning classifiers, yet these methods often lack interpretability, limiting their reliability in high-stakes applications (Yang et al., 2024).

A promising approach to enhancing interpretability in ATD is the use of Sparse Autoencoders (SAEs), which learn structured representations of textual data by enforcing sparsity constraints

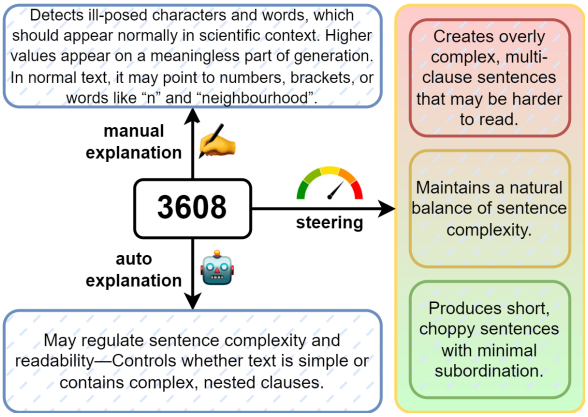

Figure 1: Interpretations of one of the most "universal" SAE features that are useful for ATD task.

(Huben et al., 2023; Makelov et al., 2024). We can extract human-interpretable features that capture the underlying structure of text.

In this study, we extend this line of research by applying SAEs from the Gemma-2-2b model (Team, 2024a) residual streams to analyze features that contribute to artificial text detection. By examining these features, we introduce a categorization of extracted features into discourse features (capturing long-range dependencies), noise features (highlighting unnatural artifacts), and style features (distinguishing stylistic variations). Our contributions are the following:

(i) we demonstrate the efficiency of SAE for the ATD task; (ii) we extract features which alone can effectively detect artificial texts for some domains and generation methods; (iii) interpreting these features, we identify meaningful patterns that contribute to ATD interpretability.

For our main dataset, we utilized a highly comprehensive and up-to-date dataset from GenAI Content Detection Task 1—a shared task on binary machine-generated text detection, conducted as part of the GenAI workshop at COLING 2025 (Wang et al., 2025). Hereafter referred to as the

COLING dataset, it contains a diverse range of model generations, from mT5 and OPT to GPT-4o and LLaMA-3. A complete list of models, along with generation examples, is provided in Appendix B.

We also performed additional experiments on the RAID dataset (Dugan et al., 2024), which contains generations from several models with various sampling methods and a wide range of attacks, from paraphrasing to homoglyph-based modifications. We provide the full list of models and attacks, along with examples of generations, in Appendix C.

2 Background

Given a token sequence $(t_1, t_2, ..., t_n)$, an LLM computes hidden representations $\mathbf{x}_i \in \mathbb{R}^d$ at each layer l as $\mathbf{x}_i^{(l)} = g^{(l)}(\mathbf{x}_1^{(l-1)}, \mathbf{x}_2^{(l-1)}, ..., \mathbf{x}_i^{(l-1)})$, where g represents a transformer block, typically including self-attention and feedforward operations. These activations encode meaningful information about text, but understanding models requires breaking them into analyzable features. Individual neurons are limited as features due to polysemanticity (Olah et al., 2020), meaning that models learn more semantic features than there are available dimensions in a layer; this situation is referred to as superposition (Elhage et al., 2022b). To recover these features, a Sparse Autoencoder (SAE) has been proposed to identify a set of directions in activation space such that each activation vector is a sparse linear combination of them (Sharkey et al., 2023).

Given activations $\mathbf{x}$ from a language model, a sparse autoencoder decomposes and reconstructs them using encoder and decoder functions with some activation function σ:

$$f(\mathbf{x}) = \sigma(\mathbf{W}_{\text{enc}}\mathbf{x} + \mathbf{b}_{\text{enc}})$$

$$\hat{x}(f) = \mathbf{W}_{\text{dec}} f(\mathbf{x}) + \mathbf{b}_{\text{dec}}$$

for which $\hat{x}(f(\mathbf{x}))$ should map back to $\mathbf{x}$. Here, the sparse and non-negative feature vector $f(\mathbf{x}) \in \mathbb{R}^M$ (with $M \gg d$) specifies how to combine columns of $\mathbf{W}_{\text{dec}}$ - learned features, or latents - to reconstruct $\mathbf{x}$.

3 Methods

In this work, we take a step towards improving the interpretability of artificial text detection using SAEs. We employ the Gemma-2-2B model along with pre-trained autoencoders on residual streams from Gemma-Scope (Lieberum et al., 2024).

Classifier models. For each even layer, we utilize an individual SAE $(f^{(l)}, \hat{x}^{(l)})$ to extract learned features from each token. To obtain a feature vector $\mathbf{f}$ representing the entire text for layer l, we sum over all tokens, yielding

$$\mathbf{f} = \sum_{i=1}^{n} f^{(l)}(\mathbf{x}_i^{(l)})$$

We use an XGBoost classifier to evaluate the expressiveness of the full feature sets for each layer and identify the most important features for further analysis. The classifiers are trained exclusively on the Train subset of COLING and evaluated on the similar Dev set, as well as on the entirely distinct Devtest and Test subsets.

For a detailed feature analysis, we also use threshold classifiers on individual features.

Manual Interpretation and Feature Steering. For manual interpretation, we analyzed the texts that activate the most important features. In layers with strong performance and generalization (layers 8 to 20), we selected the top 20 most significant features identified by XGBoost, as well as all features that achieved the highest detection performance for each domain and model using a threshold classifier. The selected features, their statistical properties, and example texts are publicly available[1].

To examine how learned features affect text generation, we use feature steering, which enables targeted modifications by selectively adjusting latent feature activations. For a given feature with number i associated with a specific text property, we first compute its maximum activation $A_{\max}$ across a reference dataset. During generation, hidden states are modified as

$$\mathbf{x}' = \mathbf{x} + \lambda A_{\max} \mathbf{d}_i$$

where $\mathbf{x}$ is the original hidden state, $\mathbf{d}_i$ is the column of $\mathbf{W}_{\text{dec}}$ and λ is a scaling factor controlling the steering effect.

Furthermore, we employed the GPT-4-o model to analyze changes across all sequences and determine the nature or function of a particular hidden feature. (see Appendix G)

4 Results

General Detection Quality. To verify that SAE-derived features enable the detection of artificially

[1] https://mgtsaevis.github.io/mgt-sae-visualization/

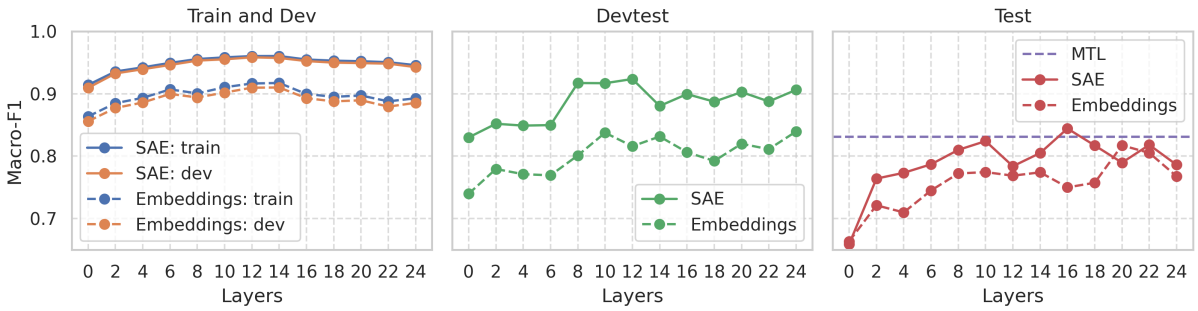

Figure 2: Macro F1 for XGBoost model on activations and SAE-derived features on different subsets of COLING

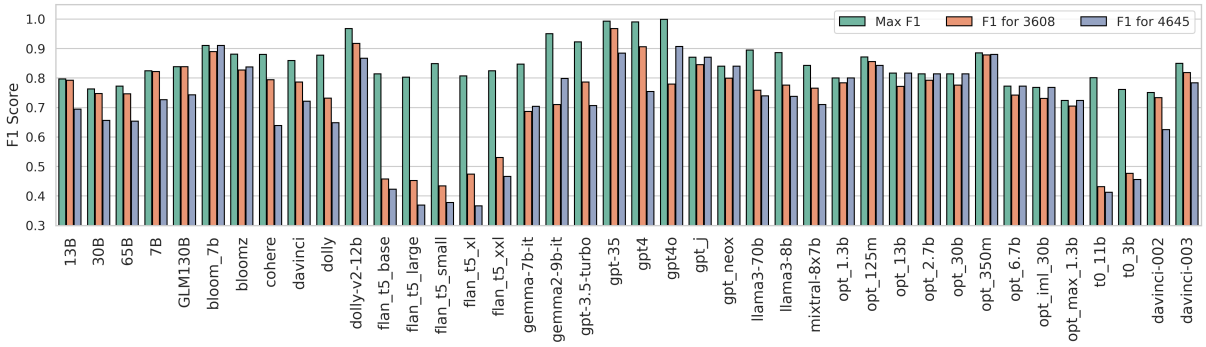

Figure 3: Macro F1 for a threshold classifier on individual features across each model for the 16th layer. Max F1 presents the maximum F1 score for every feature; features 3608 and 4645 are considered general features

generated texts, we apply XGBoost on these features and compare the results with XGBoost applied to mean-pooled activations from the layers. For training, we use the Train Subset, while testing is conducted on all remaining data.

As shown in Figure 2, both SAE features and activations perform well on this subset but degrade slightly on others. Notably, SAE features outperform activations both in training and across other subsets, suggesting that removing superposition helps the classifier focus on more fundamental, atomic features.

Although our primary objective is interpretability, it is worth noting that, at the 16th layer, SAE-derived features outperform the state-of-the-art MTL model on this dataset (Gritsai et al., 2025).

Domain/Model-Specific and General Features. In our analysis of feature structure, we aim to distinguish between general features and domain- or model-specific features. Our focus is on the 16th layer, as its features have proven to be the most expressive and lead to the best generalization, as discussed in the previous section. Given the highly imbalanced distribution in the dataset, we split it into subsets by domains or models. Then we trained a threshold-based classifier for each feature across different subsets and analyzed their performance.

Interestingly, some features consistently exhibit high classification quality across multiple domains, which we refer to as general features. In contrast, other features are more specialized, performing well only within specific domains or detecting generations of a particular subset of models, highlighting their domain- or model-specific nature. Examples of these features and their performance are shown in Figure 4.

Some general features (e.g., 3608 and 4645 in layer 16) appear universal across domains and models. To demonstrate this, we compare the best feature for detecting each generator to these universal features (Figure 3). The graph shows that for older models (e.g., flan, t0), universal feature performance drops below random, while the *opt* family is the most "universal." This suggests distinct characteristics among model classes: older/weaker models (flan, t0), more advanced LLMs (opt, bloom, gpt_j, gpt_neo), and modern families (GPT-3.5+, LLaMa, Gemma). The next section explores these differences further.

Robust Feature Analysis. Building on Kuznetsov et al. (2024), we evaluate the classifier for the presence of harmful superficial features and those vulnerable to different types of attacks on artificial text classifiers, using the RAID dataset. Details

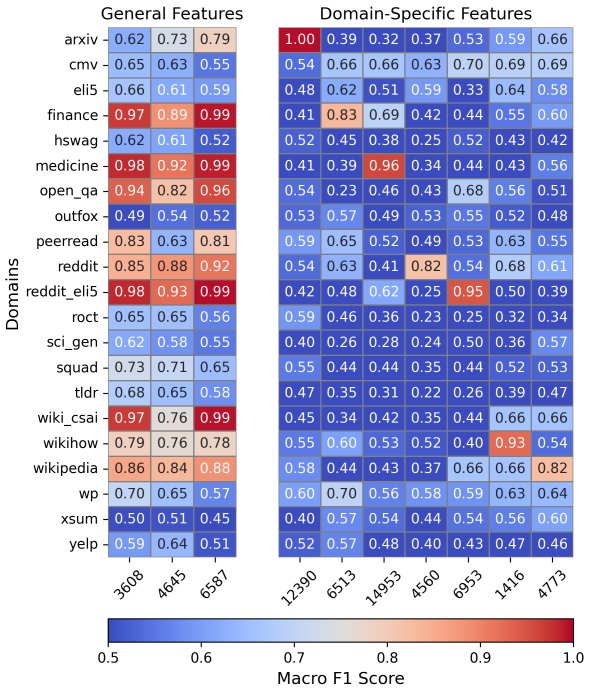

Figure 4: F1 Macro by the domains subsets for some general and domain-specific features for the 16 layer

on feature extraction can be found in Appendix D. Our analysis shows that features most susceptible to attacks and shallow text properties overlap minimally with those identified as important by XGBoost. Specifically, features 8689 (detecting the GPT3.5+ family) and 14919 (detecting the Bloom family) are very sensitive to sentence length, while other distractions have limited impact on important features.

5 Important Features Interpretation

In this section, we discuss the insights from analyzing our feature interpretations (see detailes and examples in Appendix F), starting with the most robust features: 3608, 4645, 6587, 8264, and 14161. Their performance in the ATD task across various domains and models is shown in Figures 4 and 16.

Strong activations of these features correlate with common LLM-generated text characteristics, such as excessive complexity (3608), assertive claims (4645), wordy introductions (6587), repetition (8264), and formality (14161). These features perform well on GPT3.5+ and other modern LLMs like LLaMa and Gemma, especially for domains like finance, medicine, and Wiki-CSAi. However, texts from arXiv are less distinguishable, suggesting GPT models mimic scientific writing more closely.

Feature 8264 stands out with near-perfect performance for GPT3.5+, controlling the conciseness vs. repetition of concepts. Older models lack this feature, leading to lower detectability.

Domain-specific features include overcomplicated syntax (arXiv, feature 12390), excessive details (finance, feature 6513), speculative links (Reddit, feature 4560), and hallucinated facts (Wikipedia, feature 4773). Improper tone (medicine, feature 14953) also signals machine-generated texts.

The most challenging domains for detection are Outfox (essays) and Yelp (reviews), where models mimic human-like writing. This suggests that general "overcomplexity" features may not be effective when models are instructed to avoid such traits.

6 Conclusion

Our analysis shows that modern LLMs often generate easily detectable text due to specific writing styles, such as long-winded introductions, excessive synonym substitution, and repetition. However, adversaries can bypass these features by using less formal, more personalized prompts, like student essays, leading to more human-like outputs.

Unlike previous approaches, we perform a multi-faceted analysis of features for Artificial Text Detection (ATD). We select key features, examine their behavior across domains and generators, and interpret them both through extreme values (manual) and medium shifts (steering + LLM interpretation). This approach provides deeper insights into feature meanings. For example, our interpretation of feature 3608 contrasts with Neuropedia's narrow view, which links it to "tokens associated with mathematical expressions." Similarly, feature 4645, described by Neuropedia as related to "keywords on diabetes," is more broadly relevant in our analysis.

We conclude that Sparse Autoencoder-based analysis of ATD datasets is a valuable tool for understanding text generators, detectors, and how detectors generalize to new setups. Our findings highlight that detecting AI-generated text is easy with a default prompt but becomes difficult when prompt style changes, a crucial consideration for ATD developers.

7 Limitations

Artificial text detection (ATD) is a highly complex and evolving task. With new LLMs emerging almost every month, it is difficult to predict how our

method will perform on future artificial text generators. Additionally, novel attack strategies continue to appear, and our approach covers only a subset of them. Besides, some of SAE features we studied remain challenging to interpret, as not all exhibit clear semantic meaning.

Finally, in this short paper, we used a single Sparse Autoencoder (SAE) on the residual stream of Gemma 2-2B. Exploring different SAEs on other LLMs could reveal new features and offer additional insights into artificial text detection. We leave this for future work.

References

Sara Abdali, Richard Anarfi, CJ Barberan, and Jia He. 2024. Decoding the ai pen: Techniques and challenges in detecting ai-generated text. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6428–6436.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin. Association for Computational Linguistics.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. https://transformer-circuits.pub/2023/monosemantic-features/index.html. Transformer Circuits Thread.

Shuyang Cai and Wanyun Cui. 2023. Evade chatgpt detectors via a single space. *Preprint*, arXiv:2307.02599.

Souradip Chakraborty, A. S. Bedi, Sicheng Zhu, Bang An, Dinesh Manocha, and Furong Huang. 2023. On the possibilities of AI-generated text detection. *arXiv preprint arXiv:2304.04736*.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Gpt-sentinel: Distinguishing human and chatgpt generated content. *arXiv preprint arXiv:2305.07969*.

Hoagy Cunningham, Aidan Ewart, Logan R. Riggs, Robert Huben, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*.

Liam Dugan, Alyssa Hwang, Filip Trhlík, Andrew Zhu, Josh Magnus Ludan, Hainiu Xu, Daphne Ippolito, and Chris Callison-Burch. 2024. RAID: A shared benchmark for robust evaluation of machine-generated text detectors. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12463–12492, Bangkok, Thailand. Association for Computational Linguistics.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022a. Toy models of superposition. *Transformer Circuits Thread*.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. 2022b. Toy models of superposition. *arXiv preprint arXiv:2209.10652*.

Nelson Elhage, Robert Lasenby, and Christopher Olah. 2023. Privileged bases in the transformer residual stream, 2023. *URL https://transformer-circuits. pub/2023/privilegedbasis/index. html Accessed: 2024-01-14*.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. 2023. Scaling and evaluating sparse autoencoders. OpenAI Technical Report. https://cdn.openai.com/papers/sparse-autoencoders.pdf.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M. Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116.

Aaron Grattafiori et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

German Gritsai, Anastasia Voznyuk, Ildar Khabutdinov, and Andrey Grabovoy. 2025. Advacheck at GenAI detection task 1: AI detection powered by domain-aware multi-tasking. In *Proceedings of the 1stWorkshop on GenAI Content Detection (GenAIDetect)*, pages 236–243, Abu Dhabi, UAE. International Conference on Computational Linguistics.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Preprint*, arXiv:2303.13408.

Laida Kushnareva, Tatiana Gaintseva, German Magai, Serguei Barannikov, Dmitry Abulkhanov, Kristian Kuznetsov, Eduard Tulchinskii, Irina Piontkovskaya, and Sergey Nikolenko. 2024. Ai-generated text boundary detection with roft. *Preprint*, arXiv:2311.08349.

Kristian Kuznetsov, Eduard Tulchinskii, Laida Kushnareva, German Magai, Serguei Barannikov, Sergey Nikolenko, and Irina Piontkovskaya. 2024. Robust AI-generated text detection by restricted embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 17036–17055, Miami, Florida, USA. Association for Computational Linguistics.

Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. 2023. Deepfake text detection in the wild. *arXiv preprint arXiv:2305.13242*.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, Janos Kramar, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300, Miami, Florida, US. Association for Computational Linguistics.

Aleksandar Makelov, George Lange, and Neel Nanda. 2024. Towards principled evaluations of sparse autoencoders for interpretability and control. *arXiv preprint arXiv:2405.08366*.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. DetectGPT: Zero-shot machine-generated text detection using probability curvature. *arXiv preprint arXiv:2301.11305*.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning. *Preprint*, arXiv:2211.01786.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*. Https://distill.pub/2020/circuits/zoom-in.

OpenAI. 2024a. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

OpenAI. 2024b. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. *Preprint*, arXiv:2110.08207.

John Schulman et al. 2022. Introducing chatgpt.

Lee Sharkey, Dan Braun, and Beren Millidge. 2023. Taking features out of superposition with sparse autoencoders, 2023. *URL https://www.lesswrong.com/posts/z6QQJbtpkEAX3Aojj/interim-research-report-taking-features-out-of-superposition. Accessed: 2024-01-14.*

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

Gemma Team. 2024a. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Gemma Team. 2024b. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Eduard Tulchinskii, Kristian Kuznetsov, Laida Kushnareva, Daniil Cherniavskii, Sergey Nikolenko, Evgeny Burnaev, Serguei Barannikov, and Irina Piontkovskaya. 2023. Intrinsic dimension estimation for robust detection of ai-generated texts. *Advances in Neural Information Processing Systems*, 36:39257–39276.

Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. 2021. TURINGBENCH: A benchmark environment for turing test in the age of neural text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2001–2017.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Yuxia Wang, Artem Shelmanov, Jonibek Mansurov, Akim Tsvigun, Vladislav Mikhailov, Rui Xing, Zhuohan Xie, Jiahui Geng, Giovanni Puccetti, Ekaterina Artemova, Jinyan Su, Minh Ngoc Ta, Mervat Abassy, Kareem Ashraf Elozeiri, Saad El Dine Ahmed El Etter, Maiya Goloburda, Tarek Mahmoud, Raj Vardhan Tomar, Nurkhan Laiyk, Osama Mohammed Afzal, Ryuto Koike, Masahiro Kaneko, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2025. GenAI content detection task 1: English and multilingual machine-generated text detection: AI vs. human. In *Proceedings of the 1stWorkshop on GenAI Content Detection (GenAIDetect)*, pages 244–261, Abu Dhabi, UAE. International Conference on Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. *Preprint*, arXiv:2010.11934.

Xianjun Yang, Liangming Pan, Xuandong Zhao, Haifeng Chen, Linda Ruth Petzold, William Yang Wang, and Wei Cheng. 2024. A survey on detection of LLMs-generated content. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9786–9805, Miami, Florida, USA. Association for Computational Linguistics.

Xiao Yu, Yuang Qi, Kejiang Chen, Guoqiang Chen, Xi Yang, Pengyuan Zhu, Weiming Zhang, and Nenghai Yu. 2023. Gpt paternity test: Gpt generated text detection with gpt genetic inheritance. *CoRR*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in Neural Information Processing Systems (NeurIPS)*, 32.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. Glm-130b: An open bilingual pre-trained model. *Preprint*, arXiv:2210.02414.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

A Related Work

Machine-Generated Text Detection. Detection systems for distinguishing human and AI-generated text follow two main approaches: Training-Based and Zero-Shot methods. Training-based approaches fine-tune Transformer models on labeled datasets for strong in-domain performance (Chen et al., 2023; Li et al., 2023; Yu et al., 2023). In contrast, zero-shot methods analyze statistical patterns without supervised fine-tuning, like token likelihoods, probability curvature or intrinsic dimension (Gehrmann et al., 2019; Mitchell et al., 2023; Tulchinskii et al., 2023).

However, the challenge of making AI-generated text more interpretable for humans has only been addressed by a limited number of approaches, either through manual analysis (Guo et al., 2023) or only partially investingating the dependencies (Kuznetsov et al., 2024).

Sparse Autoencoders and Interpretability. LLM interpretability is especially challenging due to polysemanticity, where a single neuron encodes multiple unrelated concepts (Elhage et al., 2022a, 2023). Sparse Autoencoders (SAEs) were proposed to help isolating more interpretable latent dimensions (Sharkey et al., 2023). Unlike standard autoencoders, SAEs introduce a penalty (e.g. L_1 regularization) to ensure that only a small subset of neurons is active per input, resulting in highly interpretable features (Cunningham et al., 2023).

Recent approaches use large language models or heuristics to automate hypothesis generation and refinement (Bricken et al., 2023; Cunningham et al., 2023; Gao et al., 2023). For example, (Bricken et al., 2023) employ GPT-4 to label sparse dimensions based on top-activating tokens, while (Cunningham et al., 2023) use heuristic methods like measuring overlap with linguistic categories to infer dimension meanings. In our work we employ both manual and automatic interpretation to ensure unbiasedness of our approach.

Datasets and Benchmarks. AI text detection includes many datasets, starting with GPT-2 Output (Solaiman et al., 2019) and Grover (Zellers et al., 2019), as well as TuringBench (Uchendu et al., 2021), which unifies 19 models for cross-evaluation. Additionally, domain-specific corpora and "in-the-wild" tests, such as (Chakraborty et al., 2023), become useful for enhancing model robustness.

B COLING dataset: additional details

The COLING dataset contains generations of the models from the following families: a) LLaMA, 7 - 65B (Touvron et al., 2023); b) LLaMA 3, 8 and 70B (Grattafiori et al., 2024); c) GLM, 130B (Zeng et al., 2023); d) Bloomz and Bloom 7B (Muennighoff et al., 2023); e) cohere[2]; f) GPT 3.5 series, including davinci 001-003 model[3] and gpt-3.5-turbo (Schulman et al., 2022); g) GPT-4 (OpenAI, 2024a) and GPT-4-o (OpenAI, 2024b); h) a line of models, based on T5 (Xue et al., 2021) and T0 (Sanh et al., 2022); i) Gemma, 7B (Team, 2024b) and Gemma 2, 9B (Team, 2024a); j) GPT-J, 6B (Wang and Komatsuzaki, 2021) and GPT-Neo-X, 20B (Black et al., 2022); k) Mixtral, 8 x 7B (Jiang et al., 2024); l) OPT, 125M - 30B (Zhang et al., 2022).

After analyzing the dataset manually, we identified that some samples contain anomalous punctuation. Figures 5 and 7 display several fragments of such samples. For comparison, Figures 6 and 8 show samples from the same models (or human texts) without these anomalies. We hypothesize that this inconsistency arises from the COLING dataset being composed of multiple datasets created by different authors.

Previous research works have shown that spurious features related to the text length (Kushnareva et al., 2024) and formatting (Dugan et al., 2024) significantly affect artificial text detection. Moreover, Cai and Cui (2023) found that sometimes adding even a single space before the comma may confuse detectors. Thus, we find it important to analyze the peculiarities of the dataset we use and investigate whether the features we examine truly reflect inherent properties of the generated texts or are simply influenced by superficial traits.

Figures 9 and 10 illustrate the frequency of various anomalies across the model generations. In particular, we found that GPT-NeoX generations contain the "...." anomaly most frequently among all models. Meanwhile, human-generated texts in the COLING dataset commonly contain spaces before commas or commas after line breaks, which is likely a side effects of preprocessing procedures applied when the datasets were compiled. Additionally, we discovered that the GPT-4-o model used double line breaks in almost every text it generated; models from the Gemma and LLaMA-3

[2] https://docs.cohere.com/docs/models

[3] https://platform.openai.com/docs/models

families displayed double line breaks in more than half of their generations as well. In contrast, human texts contained far fewer double line breaks, with occurrences of three or more line breaks being relatively rare across all models.

Talking about the lengths of the samples, we see that they also vary a lot (see Figure 11). In particular, T5- and T0- based models tend to generate much shorter texts than other models. Due to this, we investigate further which features are the most sensitive to the length of the input texts and syntactic anomaly in the Appendix D.

C RAID dataset: additional details

RAID dataset contains generations of numerous models, such as GPT-2-XL (Radford et al., 2019), davinci-002[4], ChatGPT (Schulman et al., 2022), GPT-4 (OpenAI, 2024a), Cohere[5], Mistral 7B (Jiang et al., 2023), MPT-30B[6] and LLaMA (Touvron et al., 2023). However, for our purposes we used only the most powerful ones: ChatGPT and GPT-4.

Authors experimented with two types of decoding (greedy and sampling) and applied repetition penalty to a half of generations. Also they applied various types of attacks to the texts, such as:

- **Alternative spelling** (British)
- **Article** ('the', 'a', 'an') **deletion**
- **Adding paragraph** (`\n\n`) between sentences
- Swapping the case of words from **upper** to **lower** and vise versa
- **Zero-width space**: Inserting the zero-width space `U+200B` every other character
- Adding **whitespaces** between characters
- **Homoglyph**: Swapping characters for alternatives that look similar
- Randomly shuffling digits of **numbers**
- Inserting common **misspellings**
- **Paraphrasing** with DIPPER (Krishna et al., 2023)
- Replacing words with **synonyms**.

The dataset contains 2,000 continuations for every combination of domain, model, decoding, penalty, and adversarial attack in total. However, for our purposes, we used only 100 continuations for every combination.

[4] https://platform.openai.com/docs/models
[5] https://docs.cohere.com/docs/models
[6] https://www.databricks.com/blog/mpt-30b

Figures 12 and 13 present examples of GPT-4 generations from RAID dataset with and without an attack for comparison.

D Isolating features most sensitive to the length of samples, syntactic anomalies and attacks

To identify the features that are the most sensitive to particular peculiarities of the texts, we took measures to isolate influence of those peculiarities from other text properties, such as the style or topic. To achieve this, we performed the algorithms described below.

D.1 Length

To identify features most sensitive to sample length, we used human-written texts from the COLING dataset (see Appendix B), because COLING contains a significantly larger proportion of human texts compared to model-generated ones, and these texts are much more diverse. Then, we selected those domains of human texts that contain a sufficiently large amount of text samples (> 1000 samples). For each such domain, we identified the top 10% longest and top 10% shortest texts. For both sets, we calculated the values of each feature, then computed the difference between the average feature values for the longest and shortest texts. Thus, for each domain, we identified the top-10 features with the greatest differences. Subsequently, we computed the intersection of these top-10 features across all domains, to eliminate the influence of properties of each particular domain.

D.2 Syntactic anomalies

For each syntactic anomaly, we identified the top three domains of human texts from COLING that contained the highest proportion of texts exhibiting the given anomaly. For each domain, we calculated average feature values for texts with and without the anomaly. Then, we selected top 10 features with the greatest differences for each domain. Finally, we computed the intersection of these top 10 features across all top-3 domains, isolating those features that consistently exhibited the highest sensitivity to the given anomaly. The process was repeated for several layers of SAE.

The results are presented in the Table 1.

As one can see, the most anomalies persistently activate from 1 to 3 SAE features on each layer. However, this method didn't reveal any features

```
Make sure there is enough room to move your arms around your leg.
   This will ensure that you have room to work on your knee., When
   you start, hold the bandage in your hand. Make sure it starts out
    rolled up. This will make it easier as you wrap it around your
   knee. Position your hand with the wrap in it about two inches
   below your knee joint. Take the loose end of the bandage and
   place it just under the joint with your hand. Hold it there with
   that hand while your other hand moves the bandage around your
   knee. Wrap it all the way around once until the wrap comes around
    to meet the loose end. Pull it snug to secure it.




Make sure to wrap over the end you started with and put a twist (or
   two, so that the roll returns to its original position) in the
   bandage directly above the end to hold it in place.
```

(a) LLaMA 3-70B generation fragment, several line breaks in the row

```
I just learned about broiling recently , but let 's talk about
   baking first . When you bake , you cook the food by surrounding
   it with hot air . Because the hot air is all around the food ,
   the food cooks from all the sides . If you use a toaster oven ,
   you 'll notice that the heating elements are not really on when
   you bake . They only turn on to keep the air at the temperature
   you set . Heat transfer occur from the hot air inside and the hot
    walls of the oven .
```

(b) LLaMA 7B generation fragment, anomalous spaces before punctuation marks (highlighted with red)

```
His wife. God..... she was always so beautiful. We met at college,
   you see. The only woman I ever loved. And boy did I love her. I
   never really got over her. I heard she got married, and it sucked
   . I didn't sleep for a week. Before I met her, I never realized
   that "heartache" was literal. The pain went away over time,
   mostly. I mean, if I thought about her, I didn't cry, I didn't
   cut myself. I could deal. Until her ass-wipe husband starts
   running for President. All the media knew he was a jack-ass, but
   she..... she was made for the campaign trail.
```

(c) OPT 30B generation fragment, anomalously long ellipsis (highlighted with red)

Figure 5: Machine-generated text samples, various models, anomalous punctuation

```
Either use your fingernails or a pair of pliers to secure the stud
   by folding down the spike ends on the inside of the shoe. Repeat
   this process for all of the studs.
```

(a) LLaMA 3-70B generation fragment

```
This place it average at best. Our meal was a mixed bag of good and
   bad. On the good side, took our reservations and when we showed
   up on time we were promptly seated. Also, they had a very nice
   Carpaccio appetizer. That was well done. That was it.... no more
   good. On the bad side, all of the dinners were rather bland and
   tasteless. My wife's lamb chops were nothing to write home about.
```

(b) LLaMA 7B generation fragment

```
The first time I went there a couple of years ago, it was pretty
   good. Then I went there a year ago and it was ok. Went again
   tonight and in my opinion, it was some of the worst food I have
   ever had. Like others have said, very inconsistent but either way
   , I won't be going back.
```

(c) OPT 30B generation fragment

Figure 6: Machine-generated text samples, various models, normal punctuation

```
, After scrubbing, allow the tattoo to sit for two hours without
   washing the salty scrub off. Once the two hours are up, you should
    wash it thoroughly with cold water for 5-10 minutes. You may
   notice some ink being washed away as the area is rinsed with water
   .
In case there is any bleeding, it is recommended that you soak a
   fresh, clean hand cloth in hydrogen peroxide and then press it
   against the broken skin. This helps to disinfect the area and
   prevent any infection.
It is also advisable to apply a small amount of vitamin E over the
   area as this helps to promote healing and prevent the formation of
    a scar. Vitamin E also helps to reduce inflammation and pain.
, Use a clean hand cloth to dry the skin and then an antibiotic cream
    can be applied on top. Use sterile gauze to cover the area, which
    can be held in place using tape from a first aid kit. This helps
   to protect the area and prevent infection.
, The dressing can be taken off after three days and the area
   assessed. If the skin is painful or reddened, it may be infected.
   If this is the case, it is advisable to see the doctor or visit
   the nearest hospital.
```

Figure 7: Human text fragment with anomalous line breaks before commas (highlighted with red)

Layer	Length	␣,␣	....	\n ,
16	1033, 16028	-	2889, 8689, 14919	14919, 16028
18	7373	2199	3851, 12685, 16302	12685
20	8684	6631	8573, 11612, 12748	8573, 12267

Table 1: Features, that are the most sensitive to the length of samples and syntactic anomalies

```
St Clare's Catholic Primary School in Birmingham has met with
   equality leaders at the city council to discuss a complaint from
   the pupil's family. The council is supporting the school to ensure
    its policies are appropriate. But Muslim Women's Network UK said
   the school was not at fault as young girls are not required to
   wear headscarves. Read more news for Birmingham and the Black
   Country The Handsworth school states on its website that "hats or
   scarves are not allowed to be worn in school" alongside examples
   including a woman in a headscarf. Labour councillor Waseem Zaffar,
    cabinet member for transparency, openness and equality, met the
   school's head teacher last week. In a comment posted on Facebook
   at the weekend, claiming the school had contravened the Equality
   Act, the councillor wrote: "I'm insisting this matter is addressed
    asap with a change of policy.
```

Figure 8: Human text fragment, normal punctuation

persistently sensitive to markdown paragraphs (##) and to repeating line breaks (\n\n).

Interestingly, we identified several features that reacted to markdown paragraphs by hand (for example, features 1033 and 15152 on the 16th layer of our SAE). However, the fact that these features were not captured by our algorithm suggests that they lack sufficient stability under domain variation.

Only features 8689 and 14919 from Table 1 are among the best in detecting GPT models and Bloom model families respectively (Table 15).

D.3 Attacks

To identify features most sensitive to attacks, we switched to the RAID dataset (see Appendix C). From this dataset, we selected three of the most powerful generating models: ChatGPT-3.5, GPT-4, and human. For each model and domain, we calculated the top-10 features that are the most sensitive to each type of attack, using the same method as for syntactic anomalies. Then, for each attack, we took the intersection of the top-10 features across all domains and generation models. The results are presented in the Table 2.

As one can see, the Table doesn't include "number", "paragraphs insertion", "alternative spelling", "misspelling" and "paraphrase" attacks. This is so because our method didn't find the features that would indicate these types of attack consistently across all models and domains. Also note that this time, we calculated the top-10 features not from all available features but from the top 10% most important features for ATD based on XGBoost results. If we calculate the top-10 from all possible features, our strict method don't capture any intersections.

The selected feature set does not intersect with the best ATD detection features, whether general or model- or domain-specific.

E Detailed results

We report detailed results for threshold-based classifiers. In Figure 16 we report general and model-specific features for the 16 layer. The top features by domains and models subsets are shown in Figures 14 and 15.

F Feature interpretations

In Tables 3, 4, 5, 6 we provide interpretation results of the most important features.

General features. (Table 3)

According to steering-based explanation, all presented features makes text lengthy and overwinded, but with different flavour: feature 3608 increases

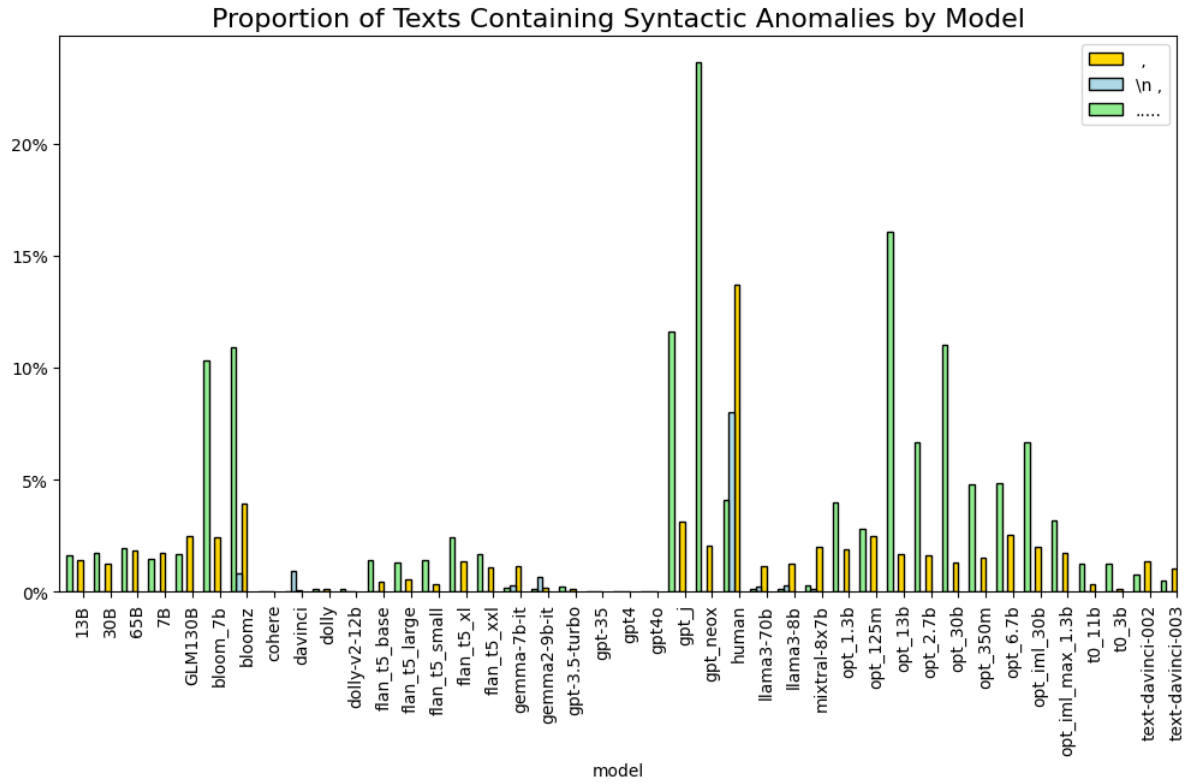

Figure 9: Frequency of occurrence of three common syntactic anomalies - spaces before commas, commas after line breaks, and ellipses with more than three dots in the text samples generated by different models. The vertical axis represents the percentage of COLING dataset samples in which each anomaly appears at least once, while the horizontal axis indicates the generation models.

sentence complexity, feature 4645 responsible for knowledge presentation complexity (even without real knowledge), and feature 6587 incorages lengthy introductionds and explanations. According the manual analysis, the first of them is concentrated on "scientfically-looking" tokens, the second reacts on factual contradictions, and the third is activated in structural elements of the text, like item labels or introduction words.

GPT-specific features.

In Table 4 we present features detecting well modern LLMs, especially GPT family. Feature 8689 responsible for excessive synonym substitutions, and feature 8264 for thoughts repetitions (by steering interpretation); from the examples we can see that the first is activated on paraphrased ideas already mentioned in the text, or on discussing alternatives. The second is activated on long common words, specific for typical GPT style.

Domain-specific features.(Tables 5, 6)

Feature 12390 (arxiv) is responsible for syntactic complexity. It is activated linking structures typical for scientific writing.

Feature 1416 (wikihow) is interpreted as increasing "phylosofical or metaphorical explanations" instead of being simple and clear. In fact, its extreme values succesfully detects texts where crucial parts are missing, namely, results of parsing errors where formulas and mathematical characters are lost. So, discarding mathematical characters is the extreme case of the unclarity.

Feature 6513 (finance) represent exessive explanations behind clear facts. It is activated on opinionate words and syntactic constructions "I mean", "like" etc

Feature 14953 (medicine) responsible for second-person speech with direct instructions. Activated on phrases containing "You" or "Your" pronouns. Steering interpretes it as change from informal to formal language.

Feature 4560 (reddit) responsible for "speculative casuality", whith Reddit discussions as its extreme implementation

Feature 4773 reacts on words flexibility. Steering interprets it as "hallucinations".

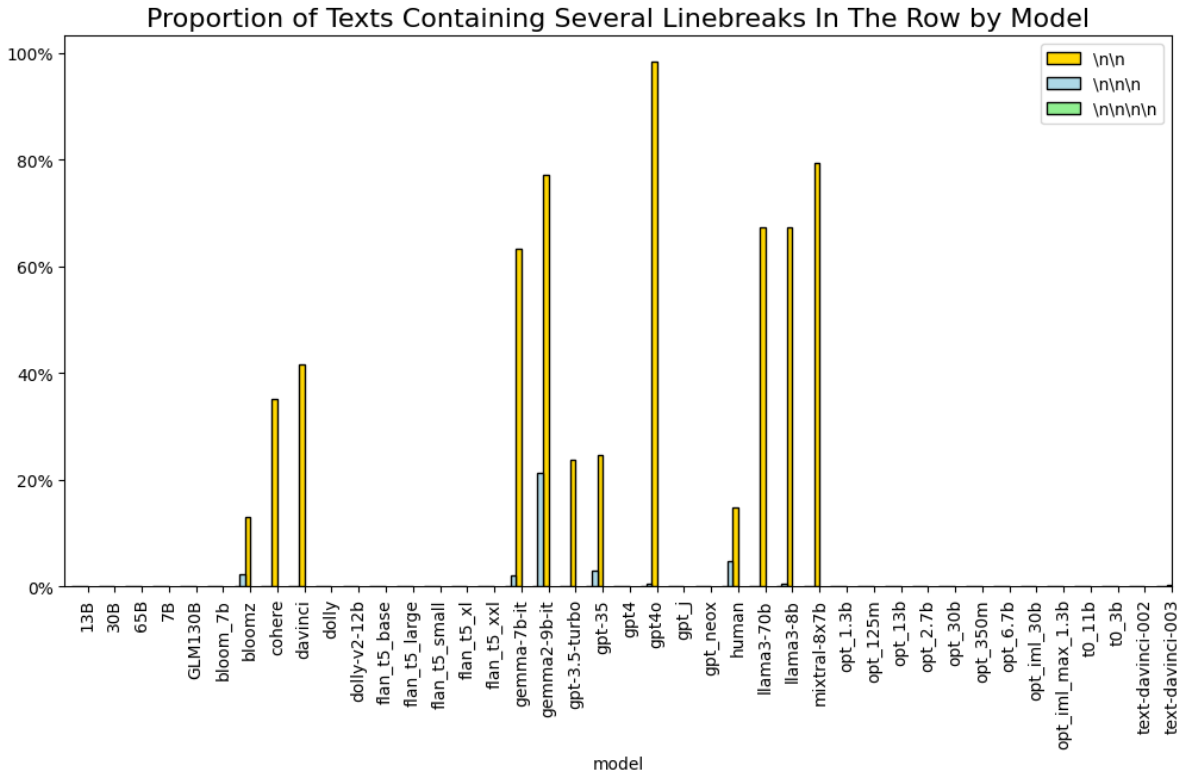

Figure 10: Frequency of occurrence of the excessive line breaks - namely, two, three or four line breaks in the row. The vertical axis represents the percentage of COLING dataset samples in which each amount of excessive line breaks appears at least once, while the horizontal axis indicates the generation models.

G Steering: additional details

Feature steering was applied using shifts from the following set: $\{-4.0, -3.0, -2.5, -2.0, -1.5, -1.0, -0.5, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 4.0\}$. To analyze the effects of these modifications, we utilized the GPT-4o model. The prompt is shown in Figure 17.

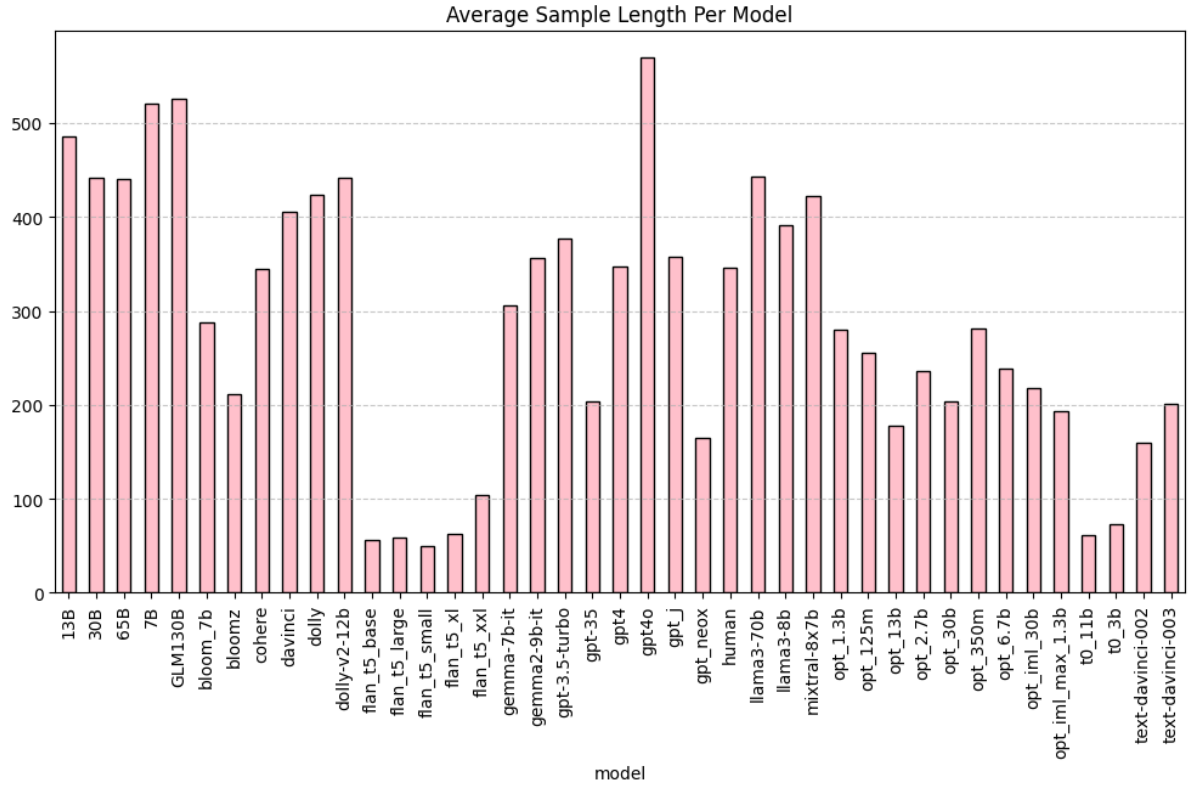

Figure 11: Average length of the text sample in COLING dataset by the generation model. The vertical axis represent the text length (measured in Gemma-2-2B tokens), the horizontal axis indicates the generation models.

```
This paper presents a comprehensive study on multiple and single
   snapshot compressive beamforming, a technique used in signal
   processing and array processing. The study explores the
   theoretical underpinnings of the method, its applications, and its
    limitations. The paper also compares the performance of multiple
   snapshot compressive beamforming with single snapshot compressive
   beamforming. The results indicate that multiple snapshot
   compressive beamforming provides superior performance in terms of
   resolution and noise suppression. However, it also requires more
   computational resources. The paper concludes with suggestions for
   future research and potential improvements in the technique.
```

Figure 12: GPT-4 generation, "misspelling" attack

```
This paper presents the second part of our study on multicell
    coordinated beamforming with rate outage constraints. We propose
     efficient approximation algorithms to address the non-convex
    and NP-hard problem of minimizing the total transmission power
    in a multicell system. The algorithms are designed to ensure a
    certain level of signal-to-interference-plus-noise ratio (SINR)
    for each user with a specified outage probability. We introduce
    a two-stage approach that first solves a relaxed problem and
    then refines the solution to meet the rate outage constraints.
    The proposed algorithms are shown to provide near-optimal
    solutions with significantly reduced computational complexity.
    Extensive simulations validate the effectiveness and efficiency
    of the proposed methods.
```

Figure 13: GPT-4 generation, no attack

Layer	Art. deletion	Homoglyph	Whitespace	0-width space	Upper/lower	Synonim
16	3518, 13998	9266	9266, 5627, 10229, 750	9266, 10262	13998	4052, 9100, 13998
18	7905, 2006	8408, 4859, 3037	281, 1970 15780	281, 12530 4859	3037, 2006	1642, 2006, 13017, 3037, 10815
20	11612	15523, 9589, 743	12602, 11363, 15415, 3879	6793, 9589	11612, 3302	11612

Table 2: Features, that are the most sensitive to various types of attacks

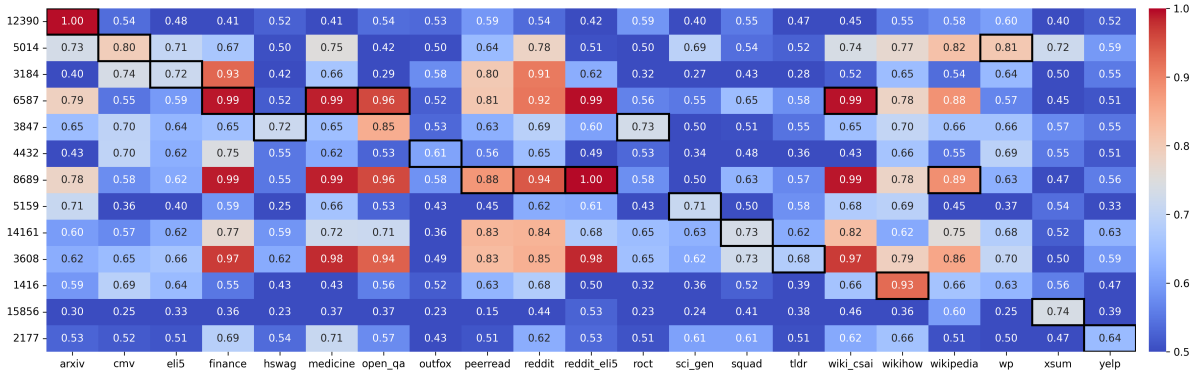

Figure 14: Top features by domains subsets. **Black** rectangles indicite the domain for which the feature is top 1.

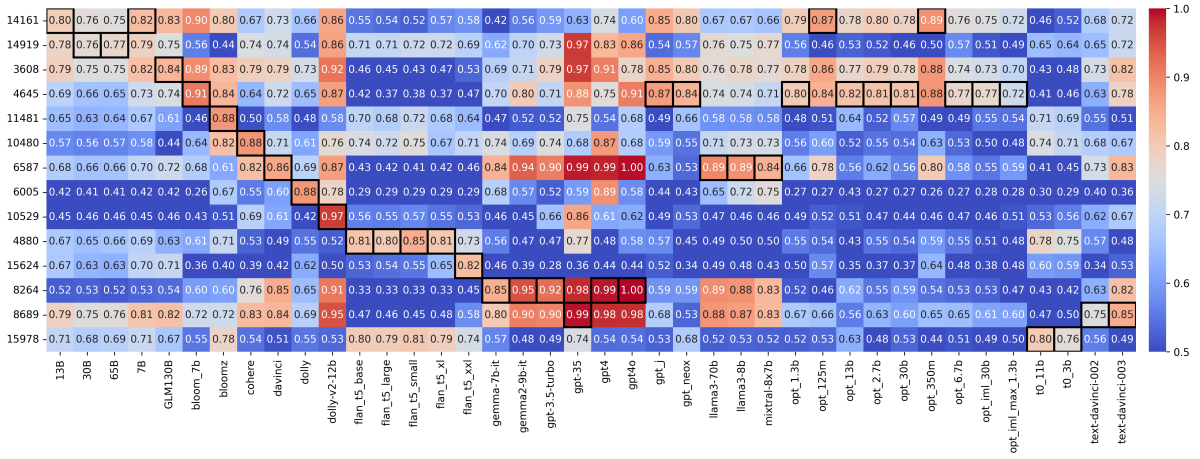

Figure 15: Top features by models subsets. **Black** rectangles indicite the model for which the feature is top 1.

Manual	**ChatGPT**	**Steering**
Feature 3608		
Detects ill-posed characters and words, which should appear normally in scientific context, e.g. numbers, brackets, or words like "n" and "neighbourhood."	May regulate sentence complexity and readability—Controls whether text is simple or contains complex, nested clauses.	**Affects:** Stylistic & Structural Complexity **Weakening (-2.0 and below)**: Produces short, choppy sentences with minimal subordination. **Neutral (0.5 to 1.5)**: Maintains a natural balance of sentence complexity. **Strong strengthening (2.0 and above)**: Creates overly complex, multi-clause sentences that may be harder to read.
Sum value: 11018.12, domain: wikihow, model: bloomz Senior year comes around every four years. It seems like it was just yesterday when we were freshmen walking into our new high schools; now we're seniors preparing ourselves for college applications. It's hard to believe how fast time flies by! Here are some tips about what you'll want to do before graduation: 1) Make sure you graduate! 2) Don't forget to celebrate! 3) Be prepared for the future. 4) Enjoy yourself. 5) Get excited. 6) Celebrate. 7) Have fun. 8) Graduate. 9) Go to parties. 10) Do whatever. 11) Congratulations. 12) Good luck. 13) See ya. 14) You did it. 15) Happy. 16) 17) 18) 19) 20) 21) 22) 23) 24) 25) 26) 27) 28) 29) 30) 31) 32) 33) 34) 35) 36) 37) 38) 39) 40) 41) 42) 43) 44) 45) 46) 47) 48) 49) 50) 51) 52) 53) 54) 55) 56) 57) 58) 59) 60) 61) 62) 63) 64) 65) 66) 67) 68) 69) 70) 71) 72) 73) 74) 75) 76) 77) 78) 79) 80) 81) 82) 83) 84) 85) 86) 87) 88) 89) 90)		
Feature 4645		
Long "lively" stories with coherent topics, but consisting mainly of common phrases, with too long sentences, hard to capture the objective of the story.	May influence factual confidence and assertion strength—Affects whether statements are presented as speculation or fact.	**Affects:** Semantic & Persuasive Strength **Weakening (-2.0 and below)**: Introduces *hedging and uncertainty* (e.g., "Some scientists believe that..."). **Neutral (0.5 to 1.5)**: Provides *balanced, well-supported claims* **Strong strengthening (2.0 and above)**: Encourages assertive, definitive claims, even when speculative (e.g.,"Scientists have proven that...").
Sum value: 24744.33, domain: wp, model: opt-30b I opened my eyes, expecting to be back in the car crash, hearing the screams of agony and the feeling of twisted metal between my ribs. But instead, I found myself on a bed with... My heart was racing as if it were running away from me. When did that happen? It had been so long since I'd considered what happened after death— but now here I lay, staring up at nothingness above me ; empty black sky and flickering lights danced around me like fireflies in a dark forest . My body felt heavy and weighted down by an unseen force all over again . "Who are you ?"		
Feature 6587		
Detects numbered lists or other well-structured step-wise reasoning text	May regulate directness vs. explanatory buildup—Affects whether information is presented concisely or with extended context.	**Affects:** Stylistic & Informational Density **Weakening (-2.0 and below)**: Produces *concise but sometimes abrupt statements* **Neutral (0.5 to 1.5)**: Ensures *a balanced level of explanation*. **Strong strengthening (2.0 and above)**: Encourages long-winded introductions before getting to the point.
Sum value: 4727.02, domain: wikihow, model: gpt-3.5-turbo How to Not Get Bored During Summer Vacation Summer vacation is a time to enjoy yourself and make memories that last a lifetime. However, sometimes it can be hard to find ways to stay entertained and not get bored during those long summer days . Luckily, there are plenty of activities you can do to keep yourself busy and have fun at the same time . Here are some ideas to try out : 1. Decorate your room : Give your room a fresh new look by hanging up some posters, re-arranging furniture or adding some colorful throw pillows . 2. Prank call someone : Make some silly phone calls with your friends and see who can come up with the funniest conversation . 3. Stay up all night : Have a late-night movie marathon, play board games, or just stay up talking with friends		

Table 3: Feature interpretations and examples of texts from the COLING dataset with exceptionally high feature values. Tokens where the feature is activated are highlighted in green. Red color highlights the parts of the text that are believed to influence the feature. For example, for feature 4645, the contradiction between the claim and the generated content is emphasized.

Manual	ChatGPT	Steering
Feature 8689, specific for **GPT family**		
Detects long "gpt-style" instructions, too verbose and obvious; highly sensitive to the presense of "...." anomaly	May influence lexical variety and synonym usage—Determines whether text repeats the same words or uses synonyms.	**Affects:** Stylistic & Lexical Diversity **Weakening (-2.0 and below)**: Causes *overuse of the same words and phrases*. **Neutral (0.5 to 1.5)**: Provides *natural variation in word choice*. **Strong strengthening (2.0 and above)**: Uses excessive synonym substitution, sometimes making the text sound unnatural.
Sum value: 26528.57, domain: outfox, model: mixtral-8x7b In recent years, online learning has become an increasingly popular alternative to traditional brick-and-mortar education. While there are certainly advantages to attending classes in person, there are also many potential benefits to attending classes online from home, particularly for students who are sick or have experienced bullying or assault . One of the most significant benefits of online learning for sick students is the ability to continue their education without the risk of spreading illness to others.		
Feature 8264, specific for **GPT family**		
Detects long "gpt-style" instructions, too verbose and obvious	May regulate redundancy and reiteration of key points—Controls whether concepts are concisely stated or overly repeated.	**Affects:** Stylistic & Structural Redundancy **Weakening (-4.0 to -2.0)**: Produces underdeveloped explanations lacking reinforcement. **Neutral (0.5 to 1.5)**: Ensures effective reinforcement of key ideas. **Strong strengthening (2.0 and above)**: Introduces excessive repetition, causing sentences to loop around the same idea.
Sum value: 23010.46, domain: wikihow, model: gpt4o How to Motivate an Autistic Teen or Adult to Exercise Make Sure the Exercise Environment is Calm and Natural Creating a soothing and predictable environment can do wonders for motivating an autistic teen or adult to exercise. Loud noises, bright lights, and chaotic spaces may cause sensory overload, making it difficult for them to focus. An environment that feels secure and calm can greatly enhance their willingness to engage in physical activity. Try choosing outdoor spaces like parks or serene gardens, or opt for quiet times at the gym.		

Table 4: Model-specific teatures

Feature 12390, specific for **arxiv** domain		
Activated on linking words in dependent syntactic structures related to research topic discussion.	May influence sentence complexity and syntactic variety—Determines whether text consists of simple or complex sentence structures.	**Affects:** Stylistic & Structural Complexity **Weakening (-4.0 to -2.0)**: Produces short, choppy sentences with minimal subordination. **Neutral (0.5 to 1.5)**: Maintains a natural balance of simple and complex sentences. **Strong strengthening (2.0 and above)**: Creates overly complex, multi-clause sentences, making readability difficult.
Sum value: 4348.42, domain: peerread, model: human This paper proposes an approach to learning a semantic parser using an encoder-decoder neural architecture, with the distinguishing feature that the semantic output is full SQL queries. The method is evaluated over two standard datasets (Geo880 and ATIS), as well as a novel dataset relating to document search .		
Feature 1416, specific for **wikihow** domain		
Detects scientific documents with missed formulas and special symbols (document parsing errors). In normal documents, reacts to abnormal punctuation.	May control abstract reasoning and conceptual depth—Influences how well the model develops abstract ideas or remains concrete.	**Affects:** Semantic & Logical Expansion **Weakening (-2.0 and below)**: Produces *simplistic, direct statements* without deeper analysis. **Neutral (0.5 to 1.5)**: Allows for *balanced explanation of abstract ideas*. **Strong strengthening (2.0 and above)**: Encourages philosophical, speculative, or metaphorical expansions, sometimes losing clarity.
Sum value: 3596.64, domain: wikipedia, model: human In mathematics, the Hahn decomposition theorem, named after the Austrian mathematician Hans Hahn, states that for any measurable space and any signed measure defined on the - algebra, there exist two - measurable sets, and , of such that : and .		

Table 5: Domain-specific features - part 1

Feature 6513, specific for **finance** domain		
Detects highly informal and opinionate speech	May regulate factual density vs. elaboration—Affects whether facts are presented concisely or with excessive background detail.	**Affects:** Semantic & Informational Density **Weakening (-4.0 to -2.0)**: Produces brief, surface-level facts without context. **Neutral (0.5 to 1.5)**: Provides balanced factual depth. **Strong strengthening (2.0 and above)**: Introduces unnecessary historical or background expansions.
Sum value: -, domain: reddit, model: llama3-70B And , like, eventually , she built up this whole compiler system from scratch , without even having a compiler to begin with. I mean, that' s just, wow . It' s like , she had to, like, manually translate the assembly code into machine code , which is just , ugh , so much work.		
Feature 14953, specific for **medicine** domain		
Second-person recommendations (legal, medical) in form "You should", "There are restrictions" etc	May control formality and academic tone—Determines whether text appears conversational or highly formal.	**Affects:** Stylistic & Tonal **Weakening (-4.0 to -2.0)**: Produces casual, informal language (e.g., "This is super important because..."). **Neutral (0.5 to 1.5)**: Maintains a professional but accessible tone. **Strong strengthening (2.0 and above)**: Introduces highly academic or dense phrasing (e.g., "In accordance with the prevailing theoretical framework...").
Sum value: -, domain: wikihow, model: human Each state has different requirements in order to qualify for a liquor license or permit. You should check to see that you meet those requirements before beginning the application process.		
Feature 4560, specific for **reddit** domain		
Detects signs of informal internet discussions: short 1st person sentences, conjectures, datetime labels (parsing artifacts), words like "Yeah", "Ah".	May regulate cause-effect relationships in historical and scientific explanations—Affects whether relationships between events are clearly established.	**Affects:** Semantic & Causal Coherence **Weakening (-4.0 to -2.0)**: Produces disconnected statements without clear causal links. **Neutral (0.5 to 1.5)**: Ensures logically connected, well-supported cause-effect explanations. **Strong strengthening (2.0 and above)**: Adds exaggerated or speculative causal links (e.g., "The invention of fire directly led to modern civilization.").
Sum value: -, domain: eli5, model: Bloom-30B He's like the hippie-hating version of Greg Proops . This is pretty much the only positive thing I can say about him . posted by crunchland at 6:50 AM on November 17, 201 1 At this point I'm just waiting for the inevitable "Hey guys, I'm a comedian who's got a beef with Occupy" FPP . posted by Aquaman at 6:51 AM on November 17 , 2 011 [1 favorite] This is what happens when you believe your own press.		
Feature 4773, specific for **wikipedia** domain		
The feature emphasizes words that repeat in the text many times in various forms, either morphological (for foreign words), in different languages, or just synonyms. E.g. "Toilet", "Diaper", "Infant pot"; or "Huguteaux", "Hugueois", "Huguenos". The same feature detects hallucinated generations with corrupted words.	May regulate factual consistency and logical flow—Determines whether details remain accurate or become speculative.	**Affects:** Semantic & Logical Consistency **Weakening (-4.0 to -2.0)**: Produces simplistic, repetitive descriptions (e.g., "Mars is red. Mars has an atmosphere."). **Neutral (0.5 to 1.5)**: Ensures well-structured and accurate statements. **Strong strengthening (2.0 and above)**: Encourages hallucinated details and speculative claims (e.g., "Mars has underground oceans and a red haze.").
Sum value: -, domain: wikipedia, model: human Arach nology can be broken down into several specialties, including: acar ology – the study of ticks and mites ar aneology – the study of spiders scorp iology – the study of scorpions		

Table 6: Domain-specific features - part 2

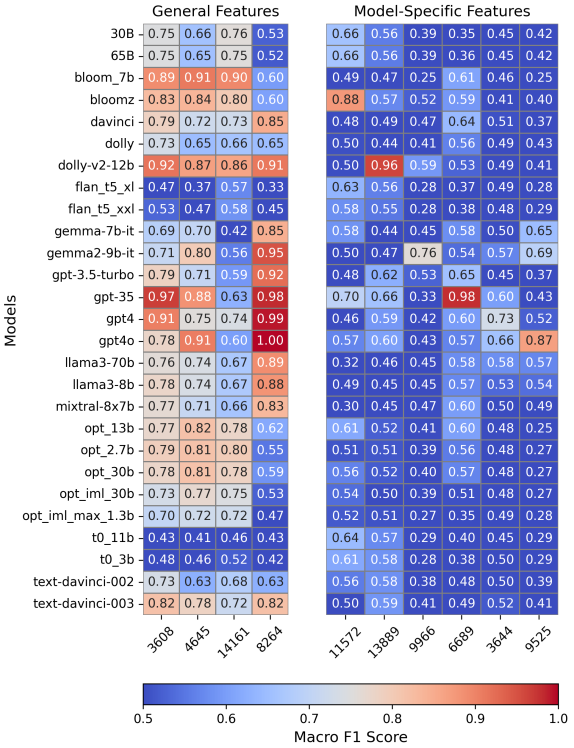

Figure 16: F1 Macro by the models subsets for some general and model-specific features for the 16 layer

```
You will see the features {} with sequences of 50 text generations
   each. Each sequence consists of an original text and a modified
   version where a specific hidden feature has been gradually
   strengthened or weakened. The same hidden feature is shifted
   consistently across all sequences.
Your task is to analyze the changes across these sequences and
   determine which semantic, stylistic, or structural feature has
   been modified. Try to find for each feature the dependencies and
   hidden meaning.

Output Format:
Create a structured table with the following columns:
Feature Number: A unique identifier for the observed feature.
Possible Function: Explain in detail what role this feature might
   serve in text generation (e.g., enhancing coherence, increasing
   formality, affecting emotional tone).
Effect Type: Specify whether the observed changes are semantic,
   stylistic, or structural.
Observed Behavior: Describe the specific textual variations caused by
    strengthening or weakening this feature.
Each row should correspond to a distinct feature, listing its effects
    and possible functions with sufficient explanation
```

Figure 17: Prompt used for steering analysis