Toward Trustworthy Language Models: Interpretation Methods and Clinical Decision Support Applications

Aliyah Hsu

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2025-57
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2025/EECS-2025-57.html

May 14, 2025

Toward Trustworthy Language Models: Interpretation Methods and Clinical Decision Support Applications

By

Aliyah Hsu

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Bin Yu, Chair
Professor John DeNero
Assistant Professor Irene Chen

Spring 2025

Toward Trustworthy Language Models: Interpretation Methods and Clinical Decision Support Applications

Abstract

Toward Trustworthy Language Models: Interpretation Methods and Clinical Decision Support Applications

by

Aliyah Hsu

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Science

University of California, Berkeley

Professor Bin Yu, Chair

As deep learning models are increasingly deployed in high-stakes domains like healthcare, understanding their decision-making processes has become essential. While numerous interpretation methods have been proposed in response, many remain unreliable (i.e., being sensitive to input perturbations, or misaligned with real-world reasoning) and struggle to scale effectively. This dissertation advances interpretability in deep learning through a structured investigation across three fronts: post-hoc explanations for black-box models, mechanistic insights into deep learning model internals, and interpretable real-world clinical applications guided by domain expertise. A central emphasis is placed on ensuring the *trustworthiness* of the developed methods through internal stability analyses and external validation in collaboration with domain experts on real-world tasks. First, we develop two black-box interpretation methods: one distills symbolic rules from concept bottleneck models, and the other uses prompt-based techniques to generate natural language explanations from text modules, both offering interpretable outputs without internal model access. Next, by extending the utility of contextual decomposition (a prior work proposed for local interpretations), we introduce a scalable, mathematically grounded method for mechanistic interpretability in transformers, efficiently identifying task-relevant computational subgraphs at fine granularity. Finally, we explore interpretability in real-world clinical decision support. In collaboration with clinicians, we develop a framework for analyzing fine-tuned transformer feature spaces to inform model suitability for tasks, and design a rule-based LLM system that autonomously applies clinical decision rules from unstructured notes to support emergency care, guided by expert feedback throughout development. These contributions collectively demonstrate how *trustworthy interpretability* can bridge the gap between model performance and trustworthy deployment in practice.

Contents

Contents **i**

1 Overview **1**
1.1 Part I: Interpreting black-box neural networks 2
1.2 Part II: Mechanistic interpretability for transformers 3
1.3 Part III: Real-world clinical decision support enhanced by interpretability . . 4

I Interpreting black-box neural networks **6**

2 Symbolic explanations and adaptive test-time interventions (FIGS-BD) **7**
2.1 Motivating the need for symbolic explanations and adaptive test-time interventions 7
2.2 Background on concept models and distillation 8
2.3 Methods for FIGS Binary Distillation and ATTI 9
2.4 FIGS-BD distills effectively and identifies relevant concepts for accurate predictions 10

3 Natural language explanations for black-box text modules (SASC) **15**
3.1 Motivating the need for natural language explanations 15
3.2 Background on explaining neural networks in natural language 16
3.3 SASC methodology 17
3.4 SASC recovers ground truth explanations for synthetic modules 19
3.5 SASC explains BERT transformer factors 23
3.6 SASC generates fMRI-voxel explanations 25
3.7 Discussion 27

II Mechanistic interpretability for transformers **29**

4 Circuit discovery using contextual decomposition (CD-T) **30**
4.1 Motivating the need for understanding transformers mechanistically through circuit discovery 30

4.2 Background on causal interpretation and circuit discovery 32
4.3 CD-T methodology and mechanistic interpretability 33
4.4 CD-T excels at identifying circuits responsible for underlying mechanisms efficiently 38
4.5 Discussion 43

III Real-world clinical decision support enhanced by interpretability 44

5 Feature space analysis with clinician input for reliable model selection (SUFO) 45
5.1 Motivating the need for interpreting fine-tuned feature space for reliable model selection in healthcare 45
5.2 Background on language models for clinical tasks and feature analysis 47
5.3 Experiment setup 48
5.4 How much does in-domain pre-training help? 50
5.5 What happens during fine-tuning? 52
5.6 Interpretation of the fine-tuned feature space 54
5.7 Discussion 57

6 Grounding large language model agents in clinical decision rules (CDR-Agent) 58
6.1 Motivating the need for an intelligent CDR selection and execution system . 58
6.2 Background on clinical decision support systems (CDSS) using LLMs 60
6.3 Problem description 60
6.4 CDR-Agent methodology 61
6.5 Dataset construction 64
6.6 Experiment setup 65
6.7 CDR-Agent selects relevant CDRs efficiently and makes cautious yet effective decisions 67
6.8 Discussion 70

Bibliography 71

A Appendices 89
A.1 FIGS-BD experiment details extended 89
A.2 SASC methodology details extended 94
A.3 CD-T circuit discovery details extended 107
A.4 SUFO experiment details extended 114

Acknowledgments

My time as a PhD student at Berkeley has been shaped by the incredible support and insight of many brilliant people. I feel deeply fortunate to have learned from, collaborated with, and been encouraged by a wide circle of mentors, colleagues, and friends, roles that often overlapped. Their guidance and generosity have played a central role in my growth as a researcher and as a person.

To begin, I am profoundly grateful to my advisor, Bin Yu, whose mentorship and guidance has shaped every facet of my PhD experience. Working with Bin has been both a privilege and an adventure: one that stretched my intellectual horizons and constantly challenged me to aim higher. With her unmatched breadth of knowledge, boundless curiosity, and tireless dedication, Bin has been an amazing teacher and role model. She taught me the value of connecting ideas across disciplines (from emergency medicine to machine learning theory) and the importance of doing rigorous work that brings real, positive impact to the world. Beyond her academic brilliance, Bin has shown a rare generosity of time and spirit. I will never forget the weekends and late nights she spent helping us refine papers, or the frequent brainstorming sessions where she met with us multiple times a week just to nurture a promising idea. Her passion for research is infectious, motivating those around her, including me, to dig deeper and stay optimistic even in the face of challenges. Bin's support extended far beyond research, offering guidance during both professional crossroads and personal moments. I truly could not have asked for a more inspiring or supportive advisor.

Next, I am thankful to Dr. Anobel Odisho, the first physician I had the opportunity to collaborate with during my time at Berkeley. Over the past five years, Anobel and I worked together on two of my most significant projects, SUFO and CD-T, and he has been an exceptional collaborator throughout. As we exchanged medical expertise and machine learning knowledge, I was continually impressed by his deep clinical knowledge and meticulous attention to detail. These projects would not have been possible without his guidance and contributions. I am especially appreciative of his kindness and patience, which created a thoughtful and supportive environment for collaborative research.

I have also been lucky to work closely with Dr. Aaron Kornblith. Through our collaboration in CDR-Agent, I learned to navigate the differing priorities between emergency medicine and computer science research, and how to approach interdisciplinary problems with clarity and care. Aaron showed me, by example, how to engage respectfully and effectively with clinicians, an essential lesson for anyone hoping to build real-world medical AI systems. I've always admired Aaron's warmth and easy rapport, traits that make him a joy to work with, and I'm equally grateful to have collaborated with the wonderful colleagues I met through him, especially Newton Addo, Margaret Lin-Martore, Jasmanpreet Kaur, and Vasuda Dokiparthi.

I am grateful to the distinguished members of my thesis and qualifying examination committee, John DeNero and Irene Chen. Their insightful feedback and suggestions have meaningfully contributed to the development and refinement of this dissertation.

I would also like to extend my gratitude to the members of the Yu Group, whose support and collaboration have greatly enriched my PhD experience. Among the many talented researchers at Berkeley, I consider myself especially fortunate to have shared this journey with such thoughtful and dedicated colleagues. Chandan Singh has been both an outstanding role model and an inspiring collaborator. His deep expertise in interpretability and remarkable ability to generate impactful research ideas make him one of the leading researchers in the field. I worked with him closely in the SASC project. I am truly grateful for his mentorship and the guidance he offered as I explored this area. During my later years, I had the pleasure of working with Abhineet Agarwal on our project exploring tree-based distillation for neural networks, which ultimately became the FIGS-BD work. Abhineet brought visionary direction to the project, drawing from a broad understanding of the literature and his own innovative thinking. His contributions made him an exceptional collaborator. Yaxuan Huang has been both a trusted colleague and a good friend during our joint research on causal interpretation for LLMs. I have always admired her strong foundations in mathematics and statistics, as well as her sharp intuition when navigating the theoretical aspects of our CD-T work. Her warmth and optimism brought energy to even the most challenging moments.

I am deeply grateful to Yeshwanth Cherapanamjeri, who has been an exceptional collaborator and mentor throughout my PhD. Yeshwanth is not only intellectually brilliant but also remarkably kind and dependable. His contributions to both the SUFO and CD-T projects were invaluable; his strong mathematical background and analytical insight helped ensure the theoretical soundness and empirical depth of our work. Beyond research, his thoughtful advice during times of uncertainty provided me with much-needed clarity and reassurance. I would also like to thank Georgia Zhou, a remarkable collaborator and the most talented undergraduate I have had the privilege to mentor. Georgia consistently impressed me with her intelligence, diligence, curiosity, technical maturity, and initiative. Working with her on the CD-T project was a true highlight; her dedication played a key role in overcoming challenges and bringing the paper to completion. On a personal level, I deeply admire her perseverance and courage in navigating life's difficulties. I am thankful to the neuroscience collaborators (Richard Antonello, Shailee Jain, and Alexander Huth) for the opportunity to work on exciting projects involving brain voxel interpretation and causal testing of brain activity steerability. Their willingness to collaborate opened a new and enriching interdisciplinary chapter in my research journey.

I also had the pleasure of working with Matthew Shen, whose meticulous attention to detail and relentless effort were instrumental to the success of the FIGS-BD paper. Austin Zane has been a valuable collaborator on our medical AI projects, consistently helping the team distill key insights from complex ideas and delivering them with exceptional clarity and presentation skills. I've enjoyed exchanging ideas with Omer Ronen, whose enthusiasm for the intersection of biology and machine learning is always inspiring, and with Zeyu Yun, a go-to colleague for engaging discussions on the latest in interpretability research. I am especially grateful to Tiffany Tang, who has been my closest mentor since my first year. Exceptionally insightful and deeply thoughtful, Tiffany has been a steady source of guidance and support, always generous in sharing her experiences and empathetic in her advice. Finally, I thank

all the former and current Yu Group members who have made my PhD journey meaningful, especially Ana Kenney, Briton Park, Corrine Elliot, James Duncan, Robin Netzorg, and Zhongyuan Liang.

I would also like to thank the many collaborators outside of Berkeley who have contributed to the development of my research. Zhen Xiang has been a fantastic collaborator on the CDR-Agent project; I consistently learned from his deep expertise in LLM agent design. I was also fortunate to work with Tristan Naumann, an exceptionally talented researcher in medical AI whose sharp experimental instincts and insightful questions greatly enriched our collaboration on the SUFO project. It was a humbling and rewarding experience to learn from him. I truly enjoyed working with Bo Li, whose clear structural thinking and extensive experience in computer science and AI security helped shape the direction of the CDR-Agent work. I'm grateful as well for the summer I spent at Microsoft Research under the guidance of Jianfeng Gao, collaborating with Xiaodong Liu and Lucas Liu. Finally, I've greatly appreciated the opportunity to work with the Einstein Language Intelligence team at Salesforce over the past two summers, especially Yixin Mao, James Zhu, Bin Bi, and Sitaram Asur.

I am also grateful to the dedicated staff in the EECS Department, the Statistics Department, and BAIR, whose behind-the-scenes efforts have played an essential role in making day-to-day life smoother throughout my PhD, especially the incredible Shirley Salanio and Judy Smithson.

Finally and most importantly, I would like to thank my family, my mom, grandma, and grandpa, for their unconditional love, unwavering support, and guidance. They have been my anchor in turbulent times and the steady light that helped me find my way. Through every step of this journey, their support has meant more than words can express. I hope what I achieved is something that brings them pride, as it was built on the foundation they gave me.

Chapter 1

Overview

Recent advances in deep learning have yielded impressive results across domains such as language understanding [24], scientific discovery [75], and healthcare [48]. However, their opaque decision-making processes remain a major obstacle to deployment in high-stakes settings. In such domains, achieving high performance on benchmark tasks is not enough, models must also be transparent, stable, and aligned with domain knowledge and values [144, 15, 156]. At the core of these requirements lies interpretability, broadly defined as the ability to explain or meaningfully understand a model's behavior. A detailed definition of interpretability in the context of machine learning is offered by Murdoch et al. [117], who propose a principled framework for evaluating interpretation methods based on three desiderata: predictive, descriptive, and relevant (as judged by a human audience). Furthermore, interpretability is essential not only for building trust, but also for enabling expert oversight [84], diagnosing failure modes for improvement [202], and selecting reliable models in collaboration with domain experts for real-world use [68].

Given its importance, the interpretation of machine learning models has received growing attention over the past decade [103, 105, 104, 106, 147, 116, 161]. While numerous interpretation methods have been proposed in response, many remain unreliable – being sensitive to input perturbations or misaligned with real-world reasoning [82, 45, 5]– and struggle to scale effectively [4, 78]. Furthermore, most prior work has focused on local importance methods, techniques that attribute model predictions to individual input features (e.g., a word in a document or a pixel in an image), or examine how input features interact to produce a prediction. These methods are popular in part because input features are directly interpretable to humans, leaving the black-box operations *within* the models largely unexplained. Relatively few efforts have sought to examine the functions of a model's internal components, understand how they compose to perform a task, or build interpretable machine learning models grounded in expert knowledge to support reliable deployment. The interpretability for the inner-workings of the models is crucial, as the internal computations consist of a significant part of the strong predictive performance.

This dissertation aims to address these challenges by presenting a structured exploration of interpretability across three levels: from post-hoc explanations of black-box models (Part

I), to mechanistic understanding of model internals (Part II), and finally to real-world clinical decision support enhanced by interpretability grounded in clinician's input (Part III). The developed interpretation methods are tested under the guidance of the PCS framework for veridical data science [203] through internal (train/test splits) and external (real-world data) validations to ensure that they are trustworthy and stable. Part I introduces two complementary methods for interpreting *black-box* neural networks: symbolic explanations via tree-distilled concept bottleneck models (CBMs) (Chapter 2) and natural language explanations extracted from black-box text modules using prompting techniques (Chapter 3). These approaches aim to generate trustworthy explanations for predictions and functions of black-box text modules separately, even when the model internals remain inaccessible. Part II builds on this foundation by probing transformer models from the inside out, introducing a scalable circuit-discovery algorithm to uncover and analyze internal computational mechanisms (Chapter 4). Finally, Part III grounds interpretability in practice: it leverages feature space analyses and clinician feedback to improve model selection in clinical natural language processing (NLP) systems (Chapter 5) and proposes a rule-grounded large language model (LLM) agent design to align LLM outputs with clinical decision logic (Chapter 6). These contributions collectively underscore the multifaceted nature of interpretability and demonstrate its potential to bridge the gap between model performance and real-world reliability.

1.1 Part I: Interpreting black-box neural networks

Part I focuses on generating post-hoc interpretations for *black-box* neural networks, where a user only has access to the input/output behavior of the models but not the internals, a common setup for proprietary LLMs [131]. Two interpretation methods are covered in this part.

Chapter 2 introduces FIGS-BD, a method that distills a binary-augmented concept-to-target portion of the CBM into an interpretable tree-based model, while maintaining the competitive prediction performance of the original CBM. FIGS-BD can be used in downstream tasks to explain and decompose CBM predictions into interpretable binary-concept-interaction attributions and guide adaptive test-time intervention to correct model errors efficiently. This work was published in a 2025 ICLR workshop [157], and was joint with Matthew Shen, Abhineet Agarwal, and Bin Yu.

While FIGS-BD demonstrates how to extract symbolic explanations from black-box models via distillation, it relies on the intermediate concept predictions offered by CBMs. In practice, many models operate as true black boxes without access to such signals. To address this, Chapter 3 presents Summarize and Score (SASC), a method that generates natural language explanations for text modules (i.e., functions that map text to scalar values, including LLM submodules or fitted models of brain activity) while also providing a reliability score for each explanation. SASC operates under a strict black-box assumption, relying solely on input-output behavior to characterize module selectivity. This was published in a

2023 NeurIPS workshop [163], joint with Chandan Singh, Richard Antonello, Shailee Jain, Alexander G. Huth, Bin Yu, and Jianfeng Gao.

1.2 Part II: Mechanistic interpretability for transformers

Given the introduced methods for interpreting model behavior through symbolic and natural language explanations of black-box models in Part I, Part II advances toward a deeper understanding by examining how internal computations are organized within the models themselves. In many cases, especially with LLMs, it becomes increasingly important not just to describe what a model has learned, but to mechanistically understand how specific components contribute to its overall behavior, as the latter uniquely provides an avenue for guiding manual modifications [37, 190] and reverse-engineering solutions [38, 110]. Part II shifts focus to this deeper level of interpretability, which is formally called "mechanistic interpretability".

Automated mechanistic interpretation has attracted significant interest for its potential to scale explanations of neural network internals to large models. Among the various approaches, circuit discovery [128], which identifies computational subgraphs (*circuits*) in neural networks responsible for specific tasks, stands at the frontier of this effort and is considered one of the most promising directions, due to its potential to reveal the "biology" of neural networks [98]. Existing circuit discovery methods typically rely on activation patching [31] or its approximations [172] to identify task-relevant subgraphs. However, these techniques often suffer from slow runtimes, approximation errors, and specific requirements of metrics, such as non-zero gradients.

To address these limitations, Chapter 4 develops Contextual Decomposition for Transformers (CD-T), a method for constructing interpretable circuits in LLMs automatically. Contextual decomposition was originally proposed in earlier work [116, 161] as a local importance method for RNNs and CNNs. Our work extends the utility of contextual decomposition beyond local importance, demonstrating how this general mathematical framework can be applied to mechanistic interpretability – specifically, for circuit discovery in transformers. CD-T consists of a set of mathematical equations designed to isolate the contributions of model components. By recursively computing the contribution between components (e.g., attention heads or their outputs at specific sequence positions) within a model's computational graph, and then applying pruning, CD-T reduces the circuit discovery runtime from hours to *seconds* compared to prior methods. CD-T can produce circuits at arbitrary levels of granularity and is the *first* method capable of efficiently generating circuits as fine-grained as individual attention heads at specific sequence positions. It is broadly compatible with all transformer architectures and requires neither training nor manually crafted examples. This work was published at ICLR 2025 [67], in collaboration with Georgia Zhou, Yeshwanth Cherapanamjeri, Yaxuan Huang, Anobel Y. Odisho, Peter R. Carroll, and Bin Yu.

1.3 Part III: Real-world clinical decision support enhanced by interpretability

While Part II emphasizes a detailed, computational understanding of how transformer models operate internally, interpretability must ultimately serve real-world needs, especially in high-stakes domains like healthcare. In these settings, abstract model understanding alone is insufficient; interpretability must translate into actionable insights that support human decision-makers, build trust, and ensure reliability under uncertainty. Part III shifts the focus from internal mechanisms to external impact, exploring how interpretability methods, with clinician's input, can be applied to improve clinical decision support systems.

Chapter 5 presents SUFO, a systematic framework designed to enhance the interpretability of fine-tuned transformer feature spaces. SUFO combines a suite of analytic and visualization techniques, including Supervised probing, Unsupervised similarity analysis, Feature dynamics, and Outlier analysis, to answer key questions related to model trust and interpretability. These include determining a model's suitability for a task, understanding how its feature space evolves during fine-tuning, and diagnosing failure modes. We apply SUFO in a case study investigating how pre-training data affects fine-tuning performance for real-world pathology classification tasks. The results demonstrate that some fine-tuned feature spaces align more closely with domain expertise than others, as validated by an expert clinician, offering grounded guidance for model selection in clinical contexts. This work was published at ICLR 2024 [68], in collaboration with Yeshwanth Cherapanamjeri, Briton Park, Tristan Naumann, Anobel Y. Odisho, and Bin Yu.

Building on the foundation established in Chapter 5, which uses feature space analyses and clinician input to support model evaluation and selection, Chapter 6 addresses the next key question: how can we design intelligent systems that not only support but actively assist in clinical decision-making? More precisely, Chapter 6 explores how interpretability, grounded in domain expertise, can be embedded directly within an AI system to guide real-time decisions. Clinical decision-making is inherently complex and fast-paced, especially in emergency departments (EDs), where critical, high-stakes judgments must be made quickly. Clinical Decision Rules (CDRs) are standardized evidence-based tools that combine signs, symptoms, and clinical variables into decision trees to make consistent and accurate diagnoses. However, in practice, CDR usage is often hindered by the cognitive load placed on clinicians, limiting their ability to recall and apply appropriate rules in the moment.

To address this, Chapter 6 introduces CDR-Agent, a novel LLM-based system that autonomously identifies and applies the most relevant CDRs based on unstructured clinical notes, supporting decision-making in EDs. CDR-Agent reduces hallucination in outputs by decomposing the reasoning process into three steps: CDR selection, variable extraction, and CDR execution. The system is equipped with a curated toolbox of real-world CDRs, ensuring that predictions are not only accurate but also interpretable and clinically grounded. To rigorously evaluate CDR-Agent, we constructed two novel CDR datasets, annotated and verified through multiple rounds by domain experts (ED physicians), ensuring the system's

alignment with real-world clinical practice. This work is currently[1] under review at AMIA 2025, in collaboration with Zhen Xiang, Austin V. Zane, Aaron E. Kornblith, Margaret J. Lin-Martore, Jasmanpreet C. Kaur, Vasuda M. Dokiparthi, Bo Li, and Bin Yu.

[1] As of May 2025.

Part I

Interpreting black-box neural networks

Chapter 2

Symbolic explanations and adaptive test-time interventions (FIGS-BD)

2.1 Motivating the need for symbolic explanations and adaptive test-time interventions

As machine learning models grow in complexity, understanding how predictions are made remains a critical challenge, especially when models are deployed in sensitive, human-centered domains. Concept Bottleneck Models (CBMs)[84] offer a promising approach by introducing interpretable intermediate representations (the "concepts"). CBMs can functionally be decomposed into two models: an input-to-concept model and a concept-to-target (CTT) model. Prior CBM work typically uses a linear CTT model for interpretability [84, 197, 102]. This limits the expressivity of the overall CBM, hurting downstream performance which instead requires CTT models that can capture more complex relationships between concepts. Furthermore, CBMs, especially with practitioner intervention (i.e., check correctness and edit prediction if necessary), have the potential to improve the trustworthiness and usability of models for cases like medical diagnosis [127, 204]. However, current concept intervention work does not account for difficulties of interventions in high pressure environments with practitioners lacking full domain experience: a surprisingly common scenario where machine learning could be most effectively utilized.

In this chapter, we introduce FIGS-BD [1], a method that distills the concept-to-target model of CBMs into an interpretable symbolic form without sacrificing predictive performance. By leveraging tree-based symbolic explanations, FIGS-BD enables a deeper understanding of concept interactions and supports adaptive test-time interventions (ATTI) by adaptively proposing and ranking concepts that are of highest priority for a practitioner to intervene on. Across 4 datasets with linear to complex concept interactions in nature, we demonstrate that our adaptive test-time intervention identifies key concepts that sig-

[1] Code and scripts for running FIGS-BD and experiments are available at `https://github.com/mattyshen/adaptiveTTI`.

nificantly improve performance for realistic human-in-the-loop settings that only allow for limited concept interventions. The proposed distillation and adaptive test-time intervention process is visualized in Figure 2.1.

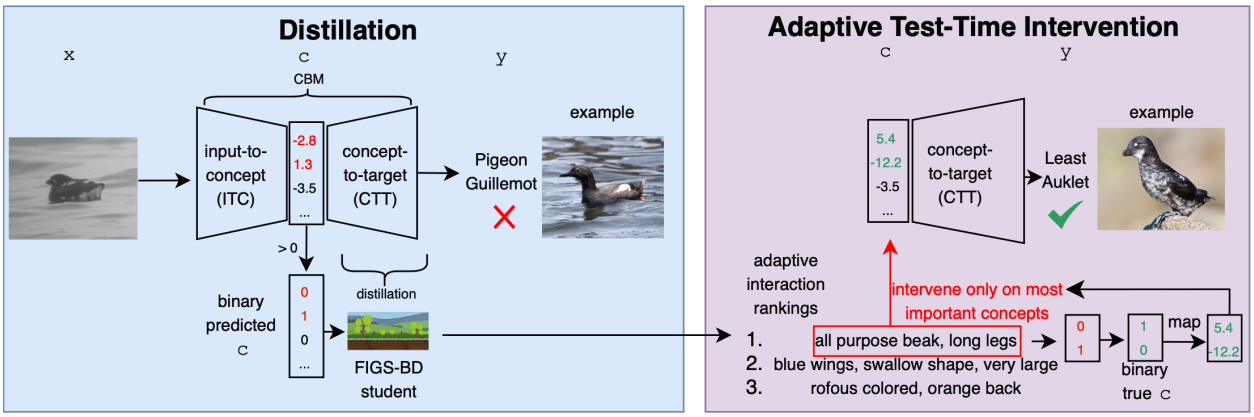


Figure 2.1: The CBM incorrectly identifies "long legs" in the image, perhaps due to the spurious correlations between water and long legged birds like seagulls. FIGS adaptive test-time intervention (ATTI) recommends a small number (2) of concepts based on a binarization of predicted concepts (including "long legs") to intervene on, which results in the correct prediction.

2.2 Background on concept models and distillation

Concept models

To improve model interpretability, models can be bottlenecked on human-level "concepts", popularized by Koh et al. [84]. The usage of concepts to understand models has expanded to analyzing models post-hoc [204], using other models (i.e. LLMs) or adapting models to iteratively generate and refine concepts for tasks [127, 153, 28, 95, 102]. Some concept models further learn soft rules [189] or (decision tree) structures [121], using the predicted concepts to improve interpretability and practitioner usage. Xu et al. [200] propose energy based CBMs to address limitations of CBMs in capturing nonlinear interactions, and similarly recognize the lack of a principled approach to test-time intervention.

Knowledge and model distillation

In knowledge and model distillation, introduced by Hinton, Vinyals, and Dean [63], a compact student model is trained on the predictions of a larger, more complex teacher model

to improve inference speed, computation, or even interpretability, while maintaining competitive predictive performance [71]. Having an interpretable model that mimics a complex model through distillation can increase the trustworthiness of complex models, streamlining their use into real-life environments.

2.3 Methods for FIGS Binary Distillation and ATTI

We utilize the Fast Interpretable Greedy Sum-Trees (FIGS) algorithm [176] to distill the CTT CBMs. We modify the original FIGS algorithm by restricting the maximum depth of the trees learned to maintain interpretability and introduce a multi-output variant to distill the soft-labels (i.e. target logits or probabilities) of CTT CBMs. The FIGS composition of a flexible, yet upper bounded, number of trees and "rules" is inherently interpretable, and practitioners can thus understand predictions made by the (CTT) CBM (and the FIGS student model) as a sum of interactions between concepts. In traditional CBMs, predicted concepts are often logits, which are highly uninterpretable and bring about unnecessary uncertainty to practitioners. An "on" or "off" binary representations of concepts alleviates this uncertainty and lack of interpretability. Thus, we binarize CBM predicted concepts with data-driven (minimize distance between true concepts) or interpretable (> 0) thresholds, and distill the CTT CBM using these binary concepts, (teacher) predicted target logits, and FIGS, which we call FIGS-BD.

Why FIGS? Predicting targets from binary concepts constitutes learning a Boolean function $f : \{0, 1\}^d \to \mathbb{R}$. All Boolean functions can be expressed as Fourier series [168]. Learning this Fourier series exactly requires exponential samples and time; FIGS-BD instead greedily approximates f by constructing a sum of shallow trees.

Algorithm 1 FIGS-BD ATTI algorithm

FIGSBD_ATTI(f_{FIGS}: FIGS-BD model, x: $\mathbb{R}^{n_{\text{concepts}}}$)
$all\_trees = trees(f_{\text{FIGS}})$
$tree\_predictions = []$
$tree\_paths = []$
for *tree* in *all_trees* **do**
    $tree\_prediction$.append($tree$.predict(x))
    $tree\_paths$.append($\text{path}_{tree}(x)$)
end for
$predictions\_and\_paths = \text{zip}(tree\_predictions, tree\_paths)$
$rankings = sort(predictions\_and\_paths, \text{lambda } x_{pred} : \text{var}(|x_{pred}|) \text{ or } \max(|x_{pred}|)))$
return *rankings*

FIGS-BD ATTI algorithm FIGS-BD ranks interactions of concepts that are embedded in the structure of its collection of trees. The ranking procedure is described in pseudo code in Algorithm 1. Thus, every set or interaction of concepts to be intervened on in Figure 2.2 and Figure 2.3 are of size maximum depth of grown tree. Note that this is not always equal to the maximum depth hyperparameter of the model, as the FIGS model does not have to grow to full depth. Additionally, concepts are re-used in some learned interactions, so intervention is not as effective after many interventions have occurred (and are thus the most impactful for the earlier sets of interactions intervened on). For each observation, these interactions of varying size are ranked based on a heuristic function (variance of absolute value of multi-output prediction and maximum of absolute value for 1-dimensional output prediction). For random ATTI, we randomly choose concepts without replacement and group/parse them of corresponding size to every FIGS ATTI to make them comparable to FIGS ATTI. For linear ATTI, we rank the n_{concepts} concepts based on variance of absolute value of product of concept prediction and concept coefficient, and group/parse them of corresponding size to every FIGS ATTI to make them comparable to FIGS ATTI. Note that when talking about variance, we refer to the variance of predictions across the multi-output target dimension.

For all of our experiments, the best FIGS-BD student models grow to depths of 3, which means that the maximum size of every interaction or cluster of concepts intervened on is 3.

2.4 FIGS-BD distills effectively and identifies relevant concepts for accurate predictions

Experimental details

Our experiments contain two tasks: computer vision (CV) and natural language processing (NLP). For CV, we train CBMs [84] on the Caltech-UCSD Birds-200-2011 (CUB) dataset [192] and the TravelingBirds [84] dataset, which is a variant of CUB where the image backgrounds associated with each bird class are changed from train to test time. CUB and TravelingBirds both pose as challenging prediction tasks with a high number of classes, while TravelingBirds also showcases a distribution shift from train to test time. For NLP, we train LLM-based Text Bottleneck Models (TBMs) [2] [102] on the AGNews topic classification dataset [214] and the CEBaB restaurant reviews [3] dataset (regression task). These two datasets are deemed to have complicated concept interactions in nature that could not be captured in previous TBM work with a linear CTT model [102]. More details of experiments are in Appendix A.1.

[2]Following the definition in Section 2.1, a TBM is also a CBM. However we refer to the models used in the NLP tasks as TBMs in the following sections of this chapter to differentiate from the CBMs used in the CV tasks.

Table 2.1: Best CBM/TBM test prediction performance with FIGS-BD and XGBoost student models across the 4 datasets. "Teacher Pred" and "Student Pred" denote teacher and student test prediction performance, respectively.

Dataset	**Teacher**	**Student**	**Teacher Pred**	**Student Pred**
CUB (Acc %)	CBM Linear	FIGS-BD	79.8	75.9
	-	XGBoost	-	75.9
TravelingBirds (Acc %)	CBM Linear	FIGS-BD	51.8	47.9
	-	XGBoost	-	47.7
AGNews (Acc %)	TBM Transformer	FIGS-BD	89.6	88.8
	-	XGBoost	-	88.0
CEBaB (R-squared)	TBM Transformer	FIGS-BD	0.868	0.871
	-	XGBoost	-	0.877

Distillation and prediction performance

Table 2.1 displays the best test performing CBM/TBM models (with CTT model specified), as well as FIGS and XGBoost [29] student models' test prediction performance on the CUB, TravelingBirds, AGNews, and CEBaB datasets. A complete table, with other comparative baselines (decision tree and random forest), is in Appendix A.1. Depending on the dataset, the relationship between concept and target can either be very simple or very complex. CUB and TravelingBirds have a fairly linear CTT relationship. For AGNews and CEBaB, complex Transformer models capture the CTT relationship the best, necessitating distillation to improve interpretability and prediction understanding. As evident in the small difference between teacher and student model prediction performance, FIGS-BD is distilling effectively, even in out-of-sample data. FIGS-BD achieves over 92.5 % of the performance of its teacher CBM on the test sets of all surveyed datasets, while generalizing better than the original CBM in some cases (CEBaB). FIGS-BD performs closely with XGBoost in the CUB and CEBaB datasets, and even outperforms XGBoost in the AGNews and TravelingBirds datasets despite being smaller (significantly less rules) and more interpretable (XGBoost fits a separate model for each class, while FIGS-BD fits a single multi-output model).

Adaptive test-time intervention results

In high-stakes environments (e.g., emergency rooms), practitioners cannot intervene across all concepts but rather can only do so for a limited number of concepts. In such scenarios, identifying an important ranking of concepts is crucial for accurate prediction. In this subsection, we consider the task: *adaptive test-time intervention (ATTI)* in which a human is allowed to intervene on a small number of concepts for a given test-example. We show how FIGS-BD can be used to adaptively rank the most important concepts for a human to

validate before prediction.

We propose constructing a sample-specific ranking of concepts based on the highest variance of absolute predictions (across the target dimension) path, from where the concepts are identified, that the sample falls down. Algorithm 1 describes this process in pseudo code. Similarly, for linear CTT portions, we propose ranking concepts based on the highest variation of absolute values of the product of fitted coefficients and predicted concept values. We believe that higher variance (across the target dimension) represents "volatile" contributions that are the most important to intervene on. More details can be found in Section 2.3 and Appendix A.1.

Quantitative prediction improvement on CUB and TravelingBirds We conduct an experiment where a practitioner is allowed to intervene on the top$-k$ interactions of concepts for a test sample recommended by various TTI methods. We consider top concepts recommended by FIGS-BD, a linear CTT, as well as random selection. We plot the results in Figure 2.2. FIGS identifies concepts that are much more relevant for making a correct prediction, indicating its utility in identifying relevant concepts for humans to validate.

Additionally, we conduct an ablation study comparing the original linear CTT model (CBM Linear) with linear ATTIs and the FIGS-BD CTT model (CBM FIGS) with FIGS ATTIs. We plot the results in Figure 2.3. The FIGS-BD CTT model quickly surpasses the linear with a practitioner's interventions, reaching drastically higher test accuracy %s with a moderate to large number of interventions. Specifically, in as few as 3 and 1 interaction interventions for CUB and TravelingBirds, respectively, the FIGS-BD CTTs outperforms linear CTTs. This highlights the impact of editing with binary values (rather than with predicted training data quantiles) and the effectiveness of FIGS ATTI. On TravelingBirds, FIGS-BD requires far less interventions (7) to reach the same accuracy as the maximum intervened on (30) linear model, potentially further disentangling the detrimental spurious correlation that was propagated into the original linear CTT model.

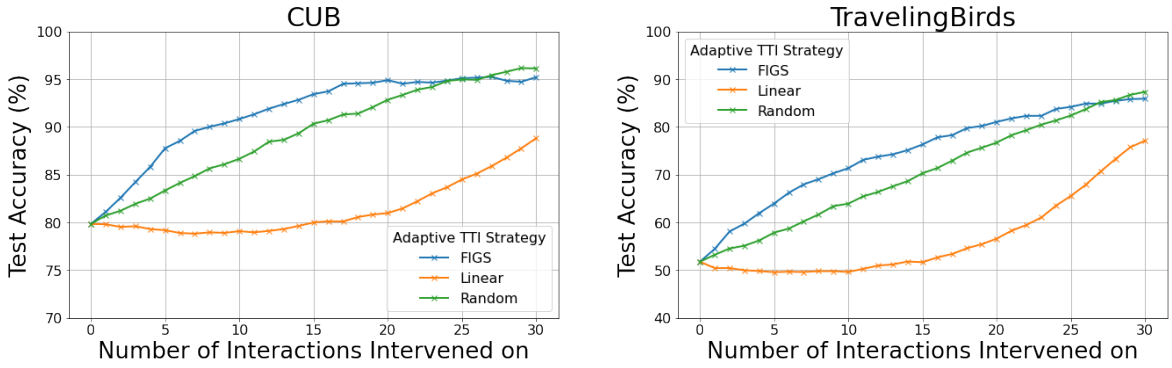


Figure 2.2: Performance of CBM linear with adaptive test-time interventions for concepts suggested by different CTT models. FIGS ATTI greatly out-performs Linear ATTI.

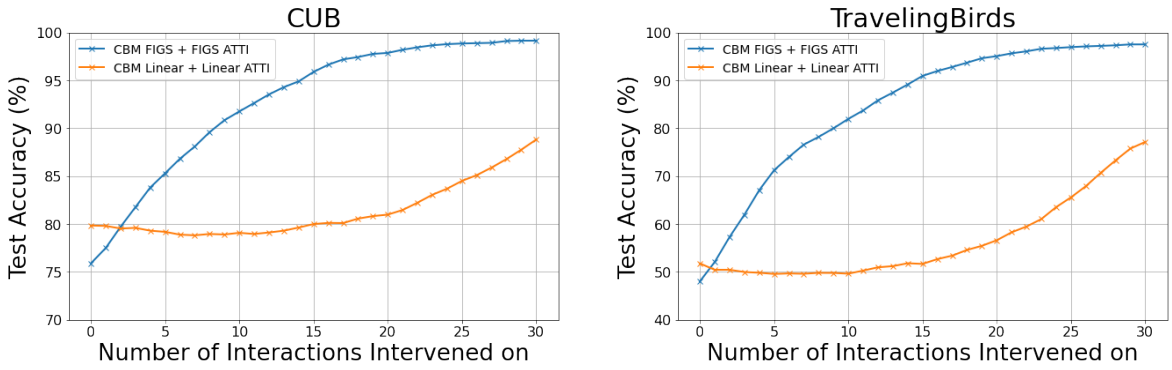


Figure 2.3: Effectiveness of adaptive test-time interventions for different concept-to-target models. Note the x-axis enumerates the number of *interactions* intervened on.

Effectiveness for correcting model prediction on AGNews and CEBaB Unlike CUB and TravelingBirds, AGNews and CEBaB lack human-labeled concepts. To address this, we manually annotate a small set of misclassified samples from these datasets with concept labels [3]. We then evaluate different intervention methods by measuring how many iterations of interventions it takes to "flip" an incorrect prediction to a correct one, intervening sequentially via ATTI interactions rankings.

Figure 2.4 shows results on AGNews and CEBaB (combined) using the linear CTT model with various ATTI strategies. FIGS ATTI consistently achieves successful flips with fewer iterations compared to other ATTIs. Moreover, it results in fewer uncorrectable samples (i.e., samples for which all recommended interventions fail to correct the prediction), and its uncorrectable samples are a strict subset of those from other methods, indicating that it successfully recovers some cases others cannot. In contrast, linear ATTI even underperforms random ATTI, suggesting that linear models struggle to recommend reliable concepts for intervention.

As a case study, we highlight an example from CEBaB where FIGS ATTI was the *only* method able to correct the model's incorrect prediction, requiring just *one* intervention. The linear CTT model initially misclassified the review `"My dining experience was one of the best. The food and service was outstanding. Everyone was very friendly just could have turned down the volume of the music a little."` with a rating of 5 instead of the ground-truth rating of 4. The model overemphasized the concept `"Customer Expectations"`, which was not present in the review. FIGS ATTI correctly identified `"Customer Expectations"`, `"Overall Satisfaction"`, and `"Service Quality"` as the most critical concepts for intervention, enabling a human to downweight the erroneous concept and produce the correct rating. In contrast, the other methods failed to surface this issue in their recommended interactions. In short, FIGS-BD ATTI's superior performance

[3] Annotations were performed by three PhD students specializing in statistics or computer science.

results from its ability to identify concept interactions crucial in determining between two competing classes effectively for human intervention.

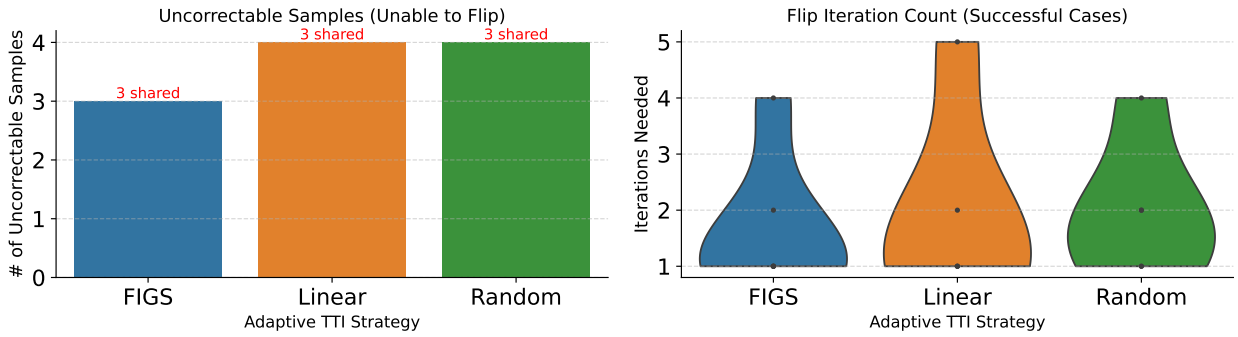


Figure 2.4: ATTI results on AGNews and CEBaB. **Left:** number of uncorrectable samples of each intervention method. **Right:** count of iterations of intervention needed of each method to flip a wrong prediction into a correct one.

Chapter 3

Natural language explanations for black-box text modules (SASC)

3.1 Motivating the need for natural language explanations

Although Chapter 2 focused on deriving symbolic explanations from black-box models by distilling them into interpretable tree-based structures, such symbolic methods require intermediate concept predictions. However, many models operate as true black boxes without accessible intermediate representations. To address this challenge, this chapter introduces Summarize and Score (SASC) [1], a framework that generates natural language explanations purely from a text module's input-output behavior. We define a *text module* f as any function that maps text to a scalar continuous value, e.g. a neuron in a pre-trained LLM[2]. By shifting from symbolic distillation to language-based summarization, SASC expands interpretability to a broader range of models, offering new pathways for understanding opaque systems.

SASC uses two steps to ground explanations in the responses of f (Figure 3.1). In the first step, SASC derives explanation candidates by sorting f's responses to ngrams and summarizing the top ngrams using a pre-trained LLM. In the second step, SASC evaluates each candidate explanation by generating synthetic text based on the explanation (again with a pre-trained LLM) and testing the response of f to the text; these responses to synthetic text are used to assign an *explanation score* to each explanation, which rates the reliability of the explanation. Decomposing explanation into these separate steps helps mitigate issues with LLM hallucination when generating and evaluating explanations.

While the expressive nature of natural-language explanations hinders formally proving

[1] Scikit-learn-compatible API available at `github.com/csinva/imodelsX` and code for experiments along with all generated explanations is available at `github.com/microsoft/automated-explanations`.

[2] Note that a neuron in an LLM typically returns a sequence-length vector rather than a scalar, so a transformation (e.g. averaging) is required before interpretation.

that SASC successfully identifies accurate explanations, we show through thorough empirical evaluations that SASC provides a significant improvement over prior work [76, 139, 118]. In our main evaluation, we test SASC on synthetic modules and find that it can recover ground truth explanations under different experimental conditions in over 88% of cases (Section 3.4). In our secondary evaluation, we use SASC to explain modules found within a pre-trained BERT model after applying dictionary learning (Section 3.5), and find that SASC explanations are of comparable quality to manual, human-given explanations. Furthermore, we verify that BERT modules which are useful for downstream text-classification tasks yield explanations related to the task more often than not.

The recovered explanations yield interesting insights. Modules found within BERT respond to a variety of different phenomena, from individual words to broad, semantic concepts. Additionally, we apply SASC to modules that are trained to predict the response of individual brain regions to language stimuli, as measured by fMRI. We find that explanations for fMRI modules pertain more to social concepts (e.g. relationships and family) than BERT modules, suggesting possible different emphases between modules in BERT and in the brain. These explanations also provide fine-grained hypotheses about the selectivity of different brain regions to semantic concepts, which could be investigated in future fMRI experiments.

3.2 Background on explaining neural networks in natural language

Explaining modules in natural language A few related works study generating natural language explanations. MILAN [60] uses patch-level information of visual features to generate descriptions of neuron behavior in vision models. iPrompt [164] and APE [219] use automated prompt engineering while D5 [218, 217], GSClip [221], and instruction induction [66] use LLMs to describe patterns in a dataset (as opposed to describing a module, as we study here). In concurrent work, Bills et al. 2023 propose an algorithm similar to SASC that explains individual neurons in an LLM by predicting token-level neuron activations. Two very related works use top-activating ngrams/sentences to construct explanations: (1) Kádár, Chrupała, and Alishahi 2017 build an explanation by *manually* inspecting the top ngrams eliciting the largest module responses from a corpus using an omission-based approach. (2) Na et al. 2019 similarly extract the top sentences from a corpus, but summarizes them using a parse tree. Alternatively, Poerner, Roth, and Schütze 2018 use input gradients to generate maximally activating text inputs.

Explaining neural-network predictions Most prior works have focused on the problem of explaining a *single prediction* with natural language, rather than an entire module, e.g. for text classification [25, 143, 120], or computer vision [59, 209]. Besides natural language explanations, some works explain individual prediction via feature importances (e.g.

LIME [147]/SHAP [106]), feature-interaction importances [115, 161, 184], or extractive rationales [206, 155]. They are not directly comparable to SASC, as they work at the prediction-level and do not produce a natural-language explanation.

Explaining neural-network representations We build on a long line of recent work that explains neural-network *representations*, e.g. via probing [32, 99], via visualization [208, 79], by categorizing neurons into categories [13, 12, 14, 34, 49], localizing knowledge in an LLM [110, 33], categorizing directions in representation space [154, 215], or distilling information into a transparent model [175, 162].

3.3 SASC methodology

SASC aims to interpret a text module f, which maps text to a scalar continuous value. For example f could be the output probability for a single token in an LLM, or the output of a single neuron extracted from a vector of LLM activations. SASC returns a driving explanation (defined below) along with an *explanation score* (Eq. 3.1), which rates how reliable the explanation is.

Definition 1 *SASC seeks a* ***driving explanation****, defined as a short, natural-language explanation that describes what elicits the strongest response from a given module f.*

In the process of explanation, SASC uses a pre-trained *helper LLM* to perform summarization and to generate synthetic text to check the stability of the explanation. This stability check is rooted in the PCS framework that has been shown to provide scientific explanations that are verified with high probability by wet-lab experiments [203, 11, 194]. To mitigate potential hallucination introduced by the helper LLM, SASC decomposes the explanation process into 2 steps (Figure 3.1) that greatly simplify the task performed by the helper LLM:

Step 1: Summarization The first step generates candidate driving explanations by summarizing ngrams. All unique ngrams are extracted from a pre-specified corpus of text and fed through the module f. To identify ngrams that drive the module, the ngrams that elicit the largest positive response from f are then fed through the helper LLM for summarization. To avoid over-reliance on the very top ngrams and to gain some stability relative to the top choices, we select a random subset of the top ngrams in the summarization step. This step is similar to prior works which summarize ngrams using manual inspection/parse trees [76, 118], but the use of the helper LLM enables flexible, automated summarization.

The computational bottleneck of SASC is computing f's response to the corpus ngrams. This computation requires two choices: the corpus underlying the extracted ngrams, and the length of ngrams to extract. Using a larger corpus/higher order ngrams can make SASC more accurate, but the computational cost grows linearly with the unique number of ngrams

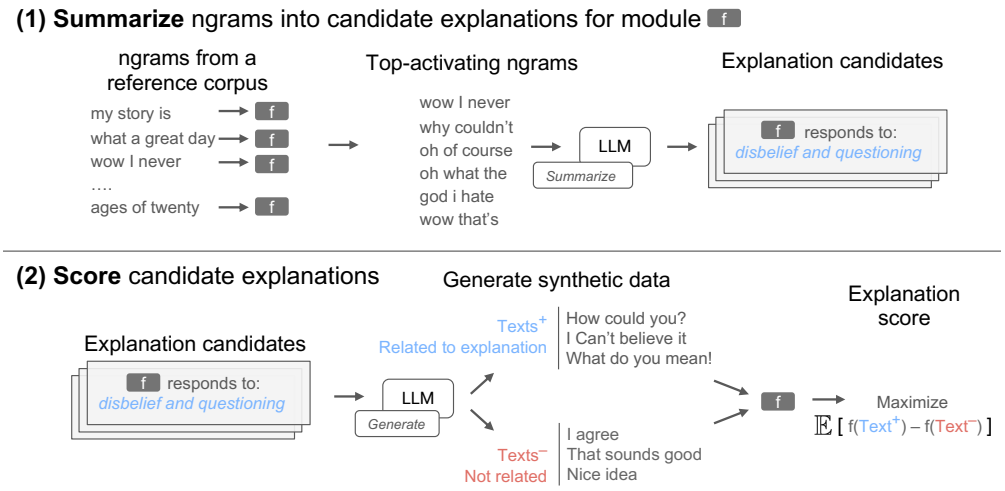


Figure 3.1: SASC pipeline for obtaining a natural language explanation given a module f. **(i)** SASC first generates candidate explanations (using a pre-trained LLM) based on the ngrams that elicit the most positive response from f. **(ii)** SASC then evaluates each candidate explanation by generating synthetic data based on the explanation and testing the response of f to the data.

in the corpus. The corpus should consist of a set of plausible inputs to f that is large enough to include relevant ngrams, as the corpus limits what generated explanations are possible (e.g. it is difficult to recover mathematical explanations from a corpus that contains no math). To speed up computation, ngrams can be subsampled from the corpus; see results when varying the corpus size in Figure A.2.

Step 2: Synthetic scoring The second step aims to evaluate each candidate driving explanation and select the most reliable one. SASC generates synthetic data based on each candidate explanation, again using the helper LLM (prompt in Appendix A.2). Intuitively, if the driving explanation accurately describes f, then f should output large values for text related to the explanation ($Text^+$) compared to unrelated synthetic text ($Text^-$).[3] We then compute the explanation score as follows:

$$\text{Explanation score} = \mathbb{E}[f(Text^+) - f(Text^-)] \quad (3.1)$$
$$\text{with units } \sigma_f,$$

where a larger score corresponds to a more reliable explanation. We report the score in units of σ_f, the standard deviation of f's response to the corpus. An explanation score of $1\sigma_f$ means that synthetic text related to the explanation increased the mean module

[3]The unrelated synthetic text should be neutral text that omits the relevant explanation. Instead of synthetic texts, a large set of neutral texts may be used for $Text^-$, e.g. samples from a generic corpus.

response by one standard deviation compared to unrelated text. SASC returns the candidate explanation that maximizes this difference, along with the synthetic data score. The selection of the highest-scoring explanation is similar to the reranking step used in some prompting methods, e.g. AutoPrompt [160], but differs in that it maximizes f's response to synthetic data rather than the likelihood of a pre-specified dataset.

Limitations and hyperparameter settings While effective, the explanation pipeline described here has some clear limitations. First and foremost, SASC assumes that f can be concisely described in a natural language string. This excludes complex functions or modules that respond to a non-coherent set of inputs. Second, SASC only describes the inputs that elicit the largest responses from f, rather than its full behavior. Finally, SASC requires that the pre-trained LLM can faithfully perform its required tasks (summarization and generation). If an LLM is unable to perform these tasks sufficiently well, users may treat the output of SASC as candidate explanations to be vetted by a human, rather than final explanations to be used.

We use GPT-3 (`text-davinci-003`, Feb. 2023) [24] as the helper LLM (see LLM prompts in Appendix A.2). In the summarization step, we use word-level trigrams, choose 30 random ngrams from the top 50 and generate 5 candidate explanations. In the synthetic scoring step, we generate 20 synthetic strings (each is a sentence) for each candidate explanation, half of which are related to the explanation.

3.4 SASC recovers ground truth explanations for synthetic modules

This section describes our main evaluation of SASC: its ability to recover driving explanations for synthetic modules with a known ground truth explanation.

Experimental setup for synthetic modules We construct 54 synthetic modules based on the pre-trained Instructor embedding model [170] (`hkunlp/instructor-xl`). Each module is based on a dataset from a recent diverse collection [216, 217] that admits a simple, verifiable keyphrase description for each underlying dataset, e.g. *related to math* (full details in Table A.4). Each module is constructed to return high values for text related to the module's groundtruth keyphrase and low values otherwise. Specifically, the module computes the Instructor embedding for an input text and for the groundtruth keyphrase; it then returns the negative Euclidean distance between the embeddings. We find that the synthetic modules reliably produce large values for text related to the desired keyphrase (Figure A.4).

We test SASC's ability to recover accurate explanations for each of our 54 modules in 3 settings: (1) The *Default* setting extracts ngrams for summarization from the dataset corresponding to each module, which contains relevant ngrams for the ground truth explanation. (2) The *Restricted corpus* setting checks the impact of the underlying corpus on the

Table 3.1: Explanation recovery performance. For both metrics, higher is better. Each value is averaged over 54 modules and 3 random seeds; errors show standard error of the mean.

	SASC		Baseline (ngram summarization)	
	Accuracy	BERT Score	Accuracy	BERT Score
Default	0.883 ±0.03	0.712 ±0.02	0.753 ±0.02	0.622 ±0.05
Restricted corpus	0.667 ±0.04	0.639 ±0.02	0.540 ±0.02	0.554 ±0.05
Noisy module	0.679 ±0.04	0.669 ±0.02	0.456 ±0.02	0.565 ±0.06
Average	**0.743**	**0.673**	0.582	0.580

performance of SASC. To do so, we restrict the ngrams we use for generating explanation candidates to a corpus from a random dataset among the 54, potentially containing less relevant ngrams. (3) The *Noisy module* assess the impact of noise in the module on SASC. Specifically, it adds Gaussian noise with standard deviation $3\sigma_f$ to all module responses in the summarization step.

Baselines and evaluation metrics We compare SASC to three baselines: (1) ngram-summarization, which summarizes top ngrams with an LLM, but does not use explanation scores to select among candidate explanations (essentially SASC without the scoring step); (2) gradient-based explanations [139], which use the gradients of f with respect to the input to generate maximally activating inputs; (3) topic modeling [19], which learns a 100-component dictionary over ngrams via latent dirichlet allocation.

We evaluate the similarity of a recovered explanation and its corresponding groundtruth explanation in two ways: (1) Accuracy: verifying whether the ground truth is essentially equivalent to the recovered explanation via manual inspection and (2) BERT-score [213][4]. We find that these two metrics, when averaged over the datasets studied here, have a perfect rank correlation, i.e. every increase in average accuracy corresponds to an increase in average BERT score. We also add a third evaluation metric that measures how often the SASC explanation matches the correct groundtruth explanation from the set of 54 module explanations in the synthetic dataset. For topic modeling, accuracy is evaluated by taking the top-30 scoring ngrams for the module (as is done with SASC), finding the 5 topics with the highest scores for these ngrams, and manually checking whether there is a match between the groundtruth and any of the top-5 words in any of these topics.

SASC can recover ground truth descriptions Table 3.1 shows the performance of SASC at recovering ground truth explanations. In the *Default* setting, SASC successfully identifies 88% of the ground truth explanations. In the two noisy settings, SASC still manages

[4]BERT-score is calculated with the recommended base model `microsoft/deberta-xlarge-mnli` [55].

Table 3.2: Explanation recovery accuracy when varying hyperparameters for the *Default* setting; averaged over 54 modules and 3 random seeds. Higher order ngrams and more powerful LLMs improve performance slightly.

	SASC (Original)	SASC (Bigrams)	SASC (4-grams)	SASC (LLaMA-2 summarizer)	SASC (LLaMA-2 generator)	Baseline (Gradient based)	Baseline (Topic modeling)
Acc.	0.883±0.03	0.815±0.04	0.889±0.03	0.870±0.03	0.852±0.04	0.093±0.01	0.111±0.01
BERT Score	0.712±0.02	0.690±0.03	0.714±0.02	0.705±0.02	0.701±0.02	0.351±0.01	0.388±0.01

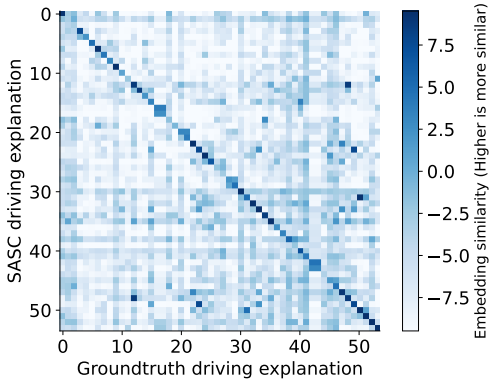


Figure 3.2: Embedding similarities for SASC explanations in the *Default*. Large values on the diagonal indicate that the explanation generated for a moduleis most similar to the groundtruth explanation for that module.

to recover explanations 67% and 68% of the time for the *Restricted ngrams* and *Noisy module* settings, respectively. In all cases, SASC outperforms the ngram-summarization baseline.

Table 3.2 shows the results for the *Default* setting when varying different modeling choices. Performance is similar across various choices, such as using bigrams or 4-grams rather than trigrams in the summarization step, or when using the LLaMA-2 13-billion parameter model [181] as the helper LLM rather than GPT-3. Table 3.2 also shows that the gradient-based baseline fails to accurately identify the underlying groundtruth text, consistent with previous work in prompting [164, 160] and that topic modeling performs poorly, largely because the topic model fails to construct topics relevant to each specific module, as the same input ngrams are shared across all modules.

Figure 3.2 shows results for the *Default* setting evaluated using embeddings from the bge-large model (`BAAI/bge-large-en`, Zhang et al. 2023). Embedding similarities are compared for each SASC explanation and each groundtruth explanation. A large similarity value in the diagonal entry compared to the rest of a row indicates that SASC correctly finds an explanation that is the most similar to the groundtruth explanation for a given dataset. The top-1 classification accuracy (i.e. how often the generated explanation is most similar to its corresponding groundtruth explanation) is 81.5% and the top-2 accuracy is 88.9%.

Table 3.3: Examples of recovered explanations for different modules in the *Default* setting.

	Groundtruth Explanation	SASC Explanation
Correct	atheistic	atheism and related topics, such as theism, religious beliefs, and atheists
	environmentalism	environmentalism and climate action
	crime	crime and criminal activity
	sports	sports
	definition	defining or explaining something
	facts	information or knowledge
Incorrect	derogatory	negative language and criticism
	ungrammatical	language
	entity	enterprise, estate, and creatures
	subjective	art and expression

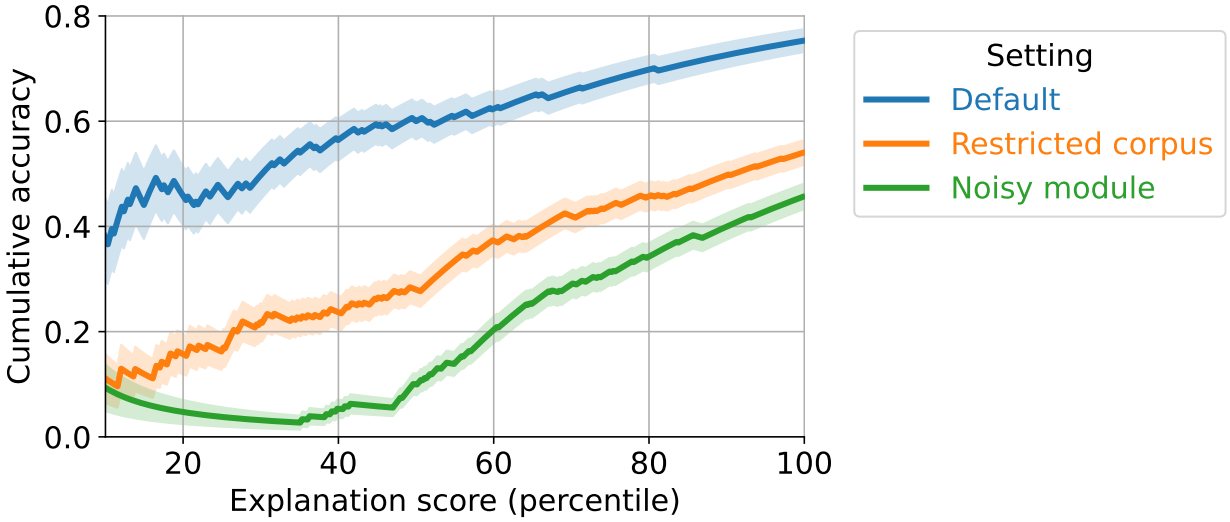


Figure 3.3: Cumulative accuracy at recovering the ground truth explanation, i.e. the accuracy for all explanations with an explanation score below a particular percentile, increases as a function of explanation score. Error bars show standard error of the mean.

Example SASC explanations Table 3.3 shows examples of correct and incorrect recovered explanations along with the ground truth explanation. For some modules, SASC finds perfect keyword matches, e.g. *sports*, or slight paraphrases, e.g. *definition* → *defining or explaining something*. For the incorrect examples, the generated explanation is often similar to the ground truth explanation, e.g. *derogatory* → *negative language and criticism*, but occasionally, SASC fails to correctly identify the underlying pattern, e.g. *ungrammatical* → *language*. Some failures may be due to the inability of ngrams to capture the underlying explanation, whereas others may be due to the constructed module imperfectly representing the ground truth explanation.

Correct explanations yield higher explanation scores Figure 3.3 shows the cumulative accuracy at recovering the ground truth explanation as a function of the explanation score. Across settings, cumulative accuracy increases as a function of explanation score, suggesting that higher explanation scores indicate more reliable explanations. This also helps validate that the helper LLM is able to successfully generate useful synthetic texts for evaluation. We additionally find that SASC's explanation performance increases with the capabilities of the helper LLM used for summarization/generation (Figure A.1) or the number of ngrams in the corpus (Figure A.2).

3.5 SASC explains BERT transformer factors

Next, we provide a secondary evaluation of SASC using explanations for modules within BERT [36] (`bert-base-uncased`). In the absence of ground truth explanations, we evaluated the explanations by (i) comparing them to human-given explanations and (ii) checking their relevance to downstream tasks.

BERT transformer factor modules One can interpret any module within BERT, e.g. a single neuron or attention head; here, we choose to interpret *transformer factors*, following a previous study that suggests that they are amenable to interpretation [205]. Transformer factors learn a transformation of activations across layers via dictionary learning (details in Appndix A.2; corpus used is the WikiText dataset [112]). Each transformer factor is a module that takes as input a text sequence and yields a scalar dictionary coefficient, after averaging over the input's sequence length. There are 1,500 factors, and their coefficients vary for each of BERT's 13 encoding layers.

Comparison to human-given explanations Table 3.4 compares SASC explanations to those given by humans in prior work (31 unique explanations from Table 1, Table 3 and Appendix in Yun et al. 2021). They are sometimes similar with different phrasings, e.g. *leaving or being left* versus *Word "left"*, and sometimes quite different, e.g. *publishing, media, or awards* versus *Institution with abbreviation.* For each transformer factor, we compare the explanation scores for SASC and the human-given explanations. The SASC explanation score is higher 61% of the time and SASC's mean explanation score is $1.6\sigma_f$ compared to $1.0\sigma_f$ for the human explanation. This evaluation suggests that the SASC explanations can be of similar quality to the human explanations, despite requiring no manual effort.

Mapping explained modules to text-classification tasks We now investigate whether the learned SASC explanations are useful for informing which downstream tasks a module is useful for. Given a classification dataset where the input X is a list of n strings and the output y is a list of n class labels, we first convert X to a matrix of transformer factor coefficients $X_{TF} \in \mathbb{R}^{n \times 19{,}500}$, where each row contains the concatenated factor coefficients across layers. We then fit a sparse logistic regression model to (X_{TF}, y), and analyze the

Table 3.4: Comparing sample SASC to human-labeled explanations for BERT transformer factors. Win percentage shows how often the SASC explanation yields a higher explanation score than the human explanation.

SASC Explanation	Human Explanation
names of parks	Word "park". Noun. a common first and last name.
leaving or being left	Word "left". Verb. leaving, exiting
specific dates or months	Consecutive years, used in football season naming.
idea of wrongdoing or illegal activity	something unfortunate happened.
introduction of something new	Doing something again, or making something new again.
versions or translations	repetitive structure detector.
publishing, media, or awards	Institution with abbreviation.
names of places, people, or things	Unit exchange with parentheses
SASC win percentage: **61%**	Human win percentage: 39%
SASC mean explanation score: $\mathbf{1.6}\sigma_f$	Human mean explanation score: $1.0\sigma_f$

explanations for the factors with the 25 largest coefficients across all classes. Ideally, these explanations would be relevant to the text-classification task; we evaluate what fraction of the 25 explanations are relevant for each task via manual inspection.

We study 3 widely used text-classification datasets: *emotion* [152] (classifying tweet emotion as sadness, joy, love, anger, fear or surprise), *ag-news* [214] (classifying news headlines as world, sports, business, or sci/tech), and *SST2* [167] (classifying movie review sentiment as positive or negative). Table 3.5 shows results evaluating the BERT transformer factor modules selected by a sparse linear model fit to these datasets. A large fraction of the explanations for selected modules are, in fact, relevant to their usage in downstream tasks, ranging from 0.35 for *Emotion* to 0.96 for *AG News*. The *AG News* task has a particularly large fraction of relevant explanations, with many explanations corresponding very directly to class labels, e.g. *professional sports teams* $\rightarrow$ *sports* or *financial investments* $\rightarrow$ *business*. See the full set of generated explanations in Appendix A.2.

Patterns in SASC explanations SASC provides 1,500 explanations for transformer factors in 13 layers of BERT. Figure 3.4 shows that the explanation score decreases with increasing layer depth, suggesting that SASC better explains factors at lower layers. The mean explanation score across all layers is $1.77\sigma_f$. To understand the breakdown of topics present in the explanations, we fit a topic model (with Latent Dirichlet Allocation [19]) to the remaining explanations. The topic model has 10 topics and preprocesses each explanation by converting it to a vector of word counts. We exclude all factors that do not attain an explanation score of at least $1\sigma_f$ from the topic model, as they are less likely to be correct. Figure 3.5 shows each topic along with the proportion of modules whose largest topic coefficient is

Table 3.5: BERT modules selected by a sparse linear model fit to text-classification tasks. First row shows the fraction of explanations for the selected modules which are relevant to the downstream task. Second row shows test accuracy for the fitted linear models. Bottom section shows sample explanations for modules selected by the linear model which are relevant to the downstream task. Values are averaged over 3 random linear model fits (errors show the standard error of the mean).

	Emotion	AG News	SST2
Fraction relevant	0.35±0.082	0.96±0.033	0.44±0.086
Test accuracy	0.75±0.001	0.81±0.001	0.84±0.001
Sample Relevant explanations	negative emotions such as hatred, disgust, disdain, rage, and horror	people, places, or things related to japan	a negative statement, usually in the form of not or nor
	injury or impairment	professional sports teams	hatred and violence
	humor	geography	harm, injury, or damage
	romance	financial investments	something being incorrect or wrong

for that topic. Topics span a wide range of categories, from syntactic concepts (e.g. *word, end, ..., noun*) to more semantic concepts (e.g. *sports, physical, activity, ...*).

3.6 SASC generates fMRI-voxel explanations

fMRI voxel modules A central challenge in neuroscience is understanding how and where semantic concepts are represented in the brain. To meet this challenge, one line of study predicts the response of different brain voxels (i.e. small regions in the brain) to natural language stimuli [69, 70]. We analyze data from LeBel et al. 2022 and Tang et al. 2023, which consists of fMRI responses for 3 human subjects as they listen to 20+ hours of narrative stories from podcasts. We fit modules to predict the fMRI response in each voxel from the text that the subject was hearing by extracting text embeddings with a pre-trained LLaMA model (`decapoda-research/llama-30b-hf`) [182]. After fitting the modules on the training split and evaluating them on the test split using bootstrapped ridge regression, we generate SASC explanations for 1,500 well-predicted voxel modules, distributed evenly among the three human subjects and diverse cortical areas (see details on the fMRI experimental setup in Appendix A.2).

Voxel explanations Table 3.6 shows examples of explanations for individual voxels, along with three top ngrams used to derive the explanation. Each explanation unifies fairly different ngrams under a common theme, e.g. *sliced cucumber, cut the apples, sauteed shiitake... → food preparation*. In some cases, the explanations recover language concepts similar to known

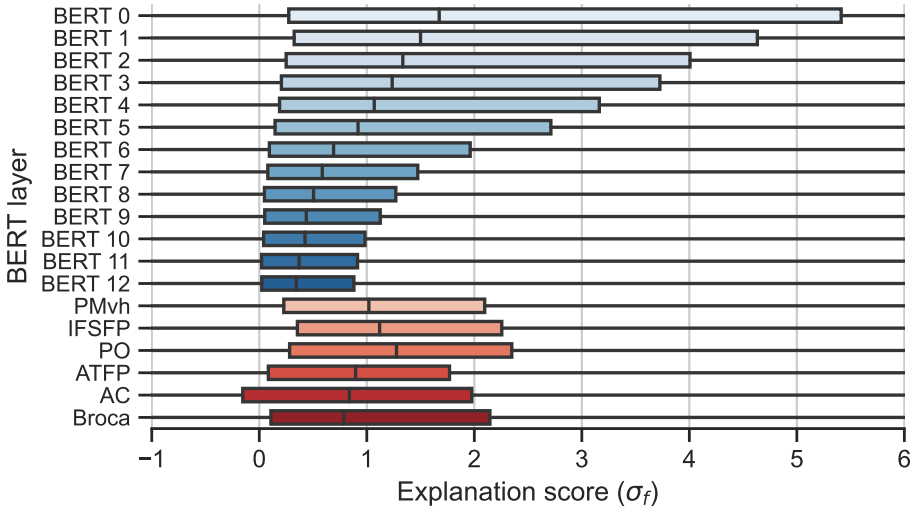


Figure 3.4: Explanation score for BERT (blue) and fMRI (red) modules. As the BERT layer increases, the explanation score tends to decrease, implying modules are harder to explain with SASC. Across regions, explanation scores for fMRI voxel modules are generally lower than scores for BERT modules in early layers and comparable to scores for the final layers. Boxes show the median and interquartile range. ROI abbreviations: premotor ventral hand area (PMvh), anterior temporal face patch (ATFP), auditory cortex (AC), parietal operculum (PO), inferior frontal sulcus face patch (IFSFP), Broca's area (Broca).

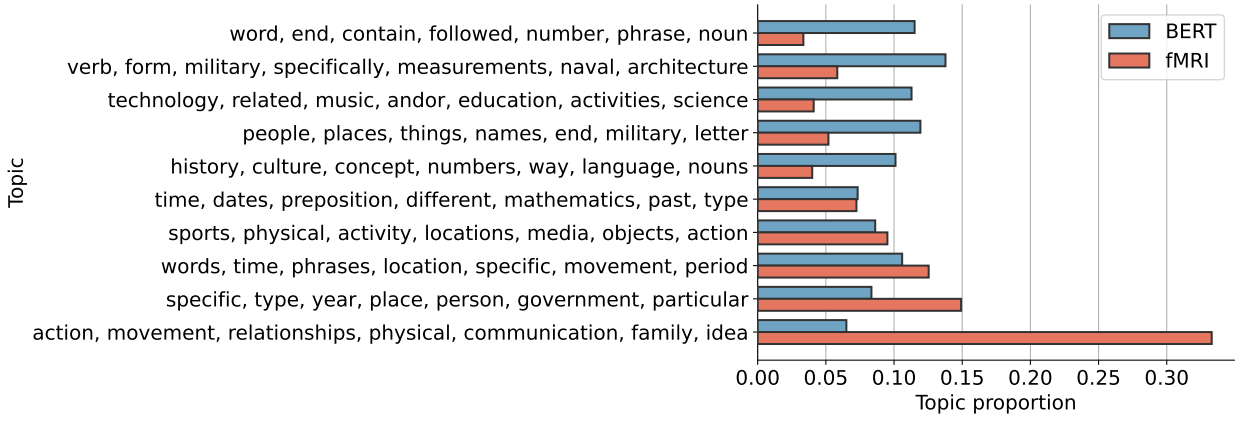


Figure 3.5: Topics found by LDA for explanations of BERT factors and fMRI voxels. Topic proportion is calculated by assigning each explanation to the topic with the largest coefficient. Topic proportions for BERT/fMRI explanations largely overlap, although the bottom topic consisting of physical/social words is much more prevalent in fMRI explanations.

Table 3.6: Examples of recovered explanations for individual fMRI voxel modules. All achieve an fMRI predicted correlation greater than 0.3 and an explanation score of at least 1σ. The third column shows 3 of the ngrams used in the SASC summarization step.

SASC Explanation	ROI	Example top ngrams
looking or staring in some way	IFSFP	eyed her suspiciously, wink at, locks eyes with
relationships and loss	ATFP	girlfriend now ex, lost my husband, was a miscarriage
physical injury or pain	Broca	infections and gangrene, pulled a muscle, burned the skin
counting or measuring time	PMvh	count down and, weeks became months, three more seconds
food preparation	ATFP	sliced cucumber, cut the apples, sauteed shiitake
laughter or amusement	ATFP, AC	started to laugh, funny guy, chuckled and

selectivity in sensory modalities, e.g. face selectivity in IFSFP [185] and selectivity for non-speech sounds such as laughter in primary auditory cortex [51]. The ngrams also provide more fine-grained hypotheses for selectivity (e.g. *physical injury or pain*) compared to the coarse semantic categories proposed in earlier language studies (e.g. *emotion*, see Huth et al. 2016; Binder et al. 2009; Mitchell et al. 2008).

Figure 3.5 shows the topics that fMRI explanations best fit into compared with BERT transformer factors. The proportions for many topics are similar, but the fMRI explanations yield a much greater proportion for the topic consisting of social words (e.g. *relationships*, *communication*, *family*) and perceptual words (e.g. *action*, *movement*, *physical*). This is consistent with prior knowledge, as the largest axis of variation for fMRI voxels is known to separate social concepts from physical concepts [69].

The selected 1,500 voxels often achieve explanation scores considerably greater than zero for their explanations (mean explanation score $1.27\sigma_f \pm 0.029$). Figure 3.4 (bottom) shows the mean explanation score for the six most common fMRI regions of interest (ROIs) among the voxels we study here. Across regions, the fMRI voxel modules generally attain explanation scores that are slightly lower than BERT modules in early layers and slightly higher than BERT modules in the final layers. We also find some evidence that the generated fMRI voxel explanations can explain not just the fitted module, but also brain responses to unseen data (see Appendix A.2). This suggests that the voxel explanations here can serve as hypotheses for followup experiments to affirm the fine-grained selectivity of specific brain voxels.

3.7 Discussion

SASC is introduced and shown to do well at finding driving explanations for a black-box text model's prediction in 3 contexts. It could potentially enable much better mechanistic interpretability for LLMs, allowing for automated analysis of submodules present in LLMs (e.g. attention heads). Along with an explanation score that helps inform when an explanation is reliable. Trustworthy explanations could help audit increasingly powerful LLMs

for undesired behavior or improve the distillation of smaller task-specific modules. SASC could also be a useful tool in many scientific pipelines. The fMRI analysis performed here generates many explanations which can be directly tested via followup fMRI experiments to understand the fine-grained selectivity of brain regions. SASC could also be used to explain text models in diverse domains, e.g. social science.

While effective, SASC has many limitations. SASC only explains a module's top responses, but it could be extended to explain the entirety of the module's responses (e.g. by selecting top ngrams differently using methods for feature attribution [147, 171] or data attribution [83]). Additionally, due to its reliance on ngrams, SASC fails to capture low-level text patterns or patterns requiring long context, e.g. patterns based on position in a sequence. Future explanations could consider adding information beyond ngrams, and also probe the interactions between different modules.

Part II

Mechanistic interpretability for transformers

Chapter 4

Circuit discovery using contextual decomposition (CD-T)

4.1 Motivating the need for understanding transformers mechanistically through circuit discovery

Transformers [188] have become the dominant architecture in modern machine learning, powering advances in language modeling, vision, and beyond [24, 54, 75]. Despite their widespread success, the internal mechanisms by which transformers compute remain largely opaque, posing challenges for trust, safety, and deeper scientific understanding. Traditional interpretability methods that treat the model as a black box often fail to capture the fine-grained interactions between internal components that drive model behavior [163, 157, 146, 105]. To address this gap, one of the main directions in mechanistic interpretability seeks to identify a computational subgraph (*circuit*) in neural networks responsible for solving a specific task [128]. Prior circuit discovery work in language models has found subgraphs of attention heads and multi-layer perceptrons (MLPs) that partially or fully explain model behaviors on certain tasks [193, 52, 119, 58]. However, most of them require tedious manual inspection, and are thus limited to smaller models [97]. Automated Circuit DisCovery (ACDC) [31] has been proposed as a scalable end-to-end technique to identify circuits in models of arbitrary sizes by automating the pruning of unimportant edges via activation patching. Although effective, ACDC suffers from slow runtimes. More recent work has focused on improving the efficiency, for example, through linear approximating activation patching with attribution patching [172, 53], or training sparse autoencoders (SAEs) to learn attention head output patterns [124]. While faster variants have been leveraged with some success, they still suffer from several drawbacks, such as approximation errors [172, 53], and the need of model training and carefully designed examples by human [124].

In this chapter, we introduce Contextual Decomposition for Transformers (CD-T) [1], a mathematical or analytical method to efficiently compute the contribution of model features to other model features within a transformer. This work builds on CD methods for CNNs and RNNs to examine local importance proposed in prior work [116, 161, 74]. We significantly extend CD by defining novel mathematical decomposition principles which can be applied to transformers of all types (e.g., decoder-based, encoder-based), because these principles are indispensable for CD to scale to modern state-of-the-art (SOTA) deep learning models as they are mostly based on transformers. Importantly, we are the first to showcase the utility of CD-T for mechanistic interpretability by proposing an automated circuit discovery algorithm utilizing CD-T. That is, CD-T is able to discover circuits with finer granularity by further splitting into sequence positions while at the same time reduce runtime from hours to *seconds*, when it is compared with prior work [172, 31]. CD-T is automatic or does not require any training or manually-crafted examples by humans. CD-T is fundamentally compatible with all common transformer architectures: in particular, CD-T supports circuit discovery in both BERT-like [36] and GPT-like [24] architectures.

Specifically, for evaluation, we use three standard circuit evaluation datasets: indirect object identification (IOI) [193], greater-than comparisons (Greater-than) [52], and docstring completion (Docstring) [58]. We compare recovered circuits by CD-T on these datasets with two SOTA baselines, ACDC [31] and EAP [172]. We demonstrate that CD-T outperforms the baselines by better identifying attention heads in the manual circuits with an average of 97% ROC AUC due to its outstanding ability in recovering negative and supporting heads efficiently. EAP obtains similar runtime as CD-T but performs slightly worse in ROC AUC for Greater-than and IOI. ACDC is the least efficient as it often takes hours to identify a circuit, yet with no better ROC AUC on all tasks compared to other methods. In addition, we demonstrate faithfulness of CD-T circuits is not due to random chance by showing our circuits are 80% more faithful than random circuits of up to 60% of the original model size. Finally, we show CD-T circuits are able to perfectly replicate original models' behavior (faithfulness = 1) using fewer nodes [2] than the baselines for all tasks. Our results underscore the great promise of CD-T for efficient automated mechanistic interpretability, paving the way for new insights into the inner-workings of large language models.

[1] Code and scripts for running CD-T and experiments are available at `https://github.com/adelaidehsu/CD_Circuit`.

[2] In our experiments, "nodes" refers to attention heads or attention heads' output at a specific sequence. See the general definition in Section 4.3.

4.2 Background on causal interpretation and circuit discovery

Causal Interpretation

Previous efforts to interpret neural networks have frequently drawn on ideas from causal inference [133]. Much of this work emphasizes counterfactual reasoning or enforcing causal constraints on computations to explain model outputs [133, 39, 44, 199, 77]. A related line of research [190, 169, 40] takes a different approach by treating internal components (or nodes, if the model is viewed as a computational graph) as mediators and conducting causal mediation analysis, also called *activation patching*. In these studies, the contribution of nodes is assessed by ablating subsets of them and measuring direct and indirect effects, with the direct effects serving as proxies for their overall contribution. In contrast to viewing nodes as mediators, Goldowsky-Dill et al. [46] and Wang et al. [193] propose treating input-to-output paths as more expressive mediators. They introduce the technique of *path patching* to explore how different edge subsets in a model's computational graph affect its behavior.

Circuit discovery methods

Ablation-based methods are essential for identifying key components within models. Through activation patching, prior research on circuit discovery in language models has uncovered subgraphs of attention heads and MLPs that explain model behavior [193, 52, 58]. However, most of these methods require multiple iterations of manual inspection and ad-hoc analysis which is specific to the model and task, limiting their application to small numbers of circuits in smaller models [97]. To address this, Conmy et al. [31] introduced Automated Circuit Discovery (ACDC), which scales to models of any size by automatically calculating edge importance between attention heads and MLPs based on model performance on a chosen metric via activation patching. While ACDC is effective, it suffers from slow runtimes and fails to recover certain attention head subsets within the manual circuits. Syed, Rager, and Conmy [172] introduced Edge Attribution Patching (EAP, or ACDC++ as later renamed), which improves ACDC's efficiency by using a linear approximation of activation patching. However, this reliance on linear approximations can lead to overestimation of edge importance and a weaker correlation with true causal effects. EAP also struggles when the gradient of the performance metric is zero. To resolve these issues, Hanna, Pezzelle, and Belinkov [53] proposed Edge Attribution Patching with Integrated Gradients (EAP-IG) for more faithful circuit recovery, although it increases runtime by a constant factor. Another approach trains sparse autoencoders (SAEs) on attention head outputs and uses the SAE-learned features to map attention head contributions to identified circuits. However, SAE-based methods either require carefully crafted human-generated examples [124] or require an impractical amount of compute to train an SAE [56, 108]. Another common limitation of prior methods is the limited level of granularity of the circuits they discover, because most work either doesn't support further attention head splitting at specific sequence positions [172, 53, 124], or it is

fundamentally feasible but requires a prohibitive amount of time to do so [31], which drastically hinders more fine-grained mechanistic interpretability analysis. Our method CD-T addresses the limitations of previous circuit discovery approaches by significantly reducing runtime, using mathematical decomposition equations with no approximations, allowing for arbitrary level of abstraction of circuits, and avoiding both model training and manually crafted examples (i.e. CD-T achieves all the above in an automated manner).

Contextual Decomposition in Neural Networks

Contextual Decomposition (CD), introduced by Murdoch, Liu, and Yu [116], attributes local importance to input features in LSTMs by analytically decomposing the output without any changes to the underlying model. For each word in a sentence, a forward pass computes activations for all cells and gates while partitioning each neuron's activation into parts influenced by a selected token or phrase (denoted as the *relevant*) and those that are not (denoted as the *irrelevant*), using a factorization of the update equations for hidden states and cell states. Jumelet, Zuidema, and Hupkes [74] propose Generalized Contextual Decomposition (GCD) for unidirectional LSTMs, applying it to phenomena like number agreement and pronoun resolution. Singh, Murdoch, and Yu [161] extend CD from LSTMs to RNNs and CNNs and introduce hierarchical interpretation with feature clustering for improved input feature attributions. A distantly related work, contextual explainability [16], focuses on feature attributions for each convolutional layer of CNNs by utilizing gradients as the feature importance measure. We further extend CD to transformers (CD-T) and we argue that our development of CD-T is a significant advance with two novel contributions. First, we carefully design mathematical decomposition principles tailored for transformers of all types (e.g., decoder-based, encoder-based), which makes it able to scale to larger models. Second, given prior work on CD only involves input feature attribution, to our knowledge, we are the first to demonstrate the utility of CD for mechanistic interpretability by proposing an automated circuit discovery algorithm that leverages CD-T.

4.3 CD-T methodology and mechanistic interpretability

In this section, we first recall the basic operations of transformers (Section 4.3). We then provide technical details on contextual decomposition and its application to transformers (section 4.3). Finally, we conclude with a circuit discovery algorithm based on CD-T (section 4.3).

Transformers

The main technical element unifying transformer-based architectures is the attention mechanism. This mechanism allows contextually useful information to be transmitted from one

position in the sequence to any other position. We start by describing the operation of a general attention mechanism before discussing the modifications needed to specialize it for generation, exemplified by models such as GPT [24], and for sequence-to-sequence transformation models such as BERT [36]. Formally, the attention mechanism computes a vector of contributions from a series of vectors $\{x_i\}_{i=1}^{l}$ to a target vector x at position p with $x_i, x \in \mathbb{R}^d$. An attention layer typically consists of a small number of heads, with each head parameterized by three functions, the key function f_k, value function f_v, and query function f_q. The output of the attention head to x is computed as follows where d_k denotes the output dimension of f_k:

$$\forall i \in [l] : k_i = f_k(x_i), q_x = f_q(x), v_i = f_v(x_i) \tag{4.1}$$

$$\forall i \in [l] : \alpha_i = \frac{\exp(q_x^\top k_i/\sqrt{d_k})}{\sum_{j\in[l]} \exp(q_x^\top k_j/\sqrt{d_k})} \tag{4.2}$$

$$y_x = \sum_{j=1}^{l} \alpha_i v_i. \tag{4.3}$$

The functions f_k, f_v, f_q are commonly simple learnable transformations such as linear transformations or one hidden-layer MLPs. An attention layer is composed of a series of attention heads applied to every position in the sequence such that the concatenation of their outputs equals the input dimension d. As the output of an attention layer is a series of vectors of the same length and dimensionality as its input, this allows for stacking of these layers to build increasingly complex representations of the input so far.

The two classes of transformer models we consider in this work, sequence-to-sequence models such as BERT and generative models such as GPT mainly differ in the set of positions that the attention mechanism operates on. Concretely, BERT maps a *fully specified* input sequence to an output sequence of the sample length while generative models produce increasingly larger sequences autoregressively with the $(t+1)^{th}$ token generated based on the previous t tokens. Since *all* the input tokens are specified for BERT-based models, the computation of the attention vector at position t utilizes representations at *every* position of the sequence. On the other hand, when generating the $(t+1)^{th}$, the positions at $t+1$ and beyond are not determined. As a consequence, only the representations for tokens before t are used to compute the attention vectors.

Contextual decomposition for transformers (CD-T)

We will now describe the general method of contextual decomposition (CD), as well as our extension to the transformer architecture and applications to mechanistic interpretability.

CD [116] was first proposed to compute contributions of input tokens to the output of LSTMs as a local interpretation method. CD divides each cell and hidden state into a sum of two parts: a β part (said to be *relevant*), which contains the part of this particular state that stems from the input tokens of interest, and a γ part (said to be *irrelevant*), which

contains information coming from tokens outside of the list of interest. β is often initialized with masked input embeddings (1's specify where token positions of interest are, and 0's the opposite), and γ the complement of β.

Given a decomposition of a vector $x \in \mathbb{R}^d$ into *relevant* and *irrelevant constituents* $\beta+\gamma = x$ (here we use the word *constituents* instead of "components" to avoid name collision with the "components of a neural network"), and a module $f : \mathbb{R}^d \to \mathbb{R}^k$, CD consists of a set of mathematical equations to determine the decomposition of the output of a module f when given x as input: $f(x) \in \mathbb{R}^k = \beta_o + \gamma_o$. The general principle is to find a symbolic expression for $f(x)$ in terms of β and γ, and group the terms in the expression according to whether they are solely a function of β or not: the sum of terms solely relying on β becomes β_o, and the sum of the remaining terms becomes γ_o. We refer interested readers to Murdoch, Liu, and Yu [116] and Singh, Murdoch, and Yu [161] for further examples of decomposition formulas of various modules. These decomposition rules can be composed through multiple modules (i.e, if $f(\beta_1, \gamma_1) = \beta_2, \gamma_2$, and $g(\beta_2, \gamma_2) = \beta_3, \gamma_3$, then $g(f(\beta_1, \gamma_1)) = \beta_3, \gamma_3$) in order to define decomposition rules for larger computational blocks, including entire neural networks.

Other authors have heuristically adjusted the decomposition rules for a module to reflect specific mechanisms by which (the relevance term in an earlier layer $i-1$) can affect (the relevance term in a later layer i) that are unaccounted for by this rule. For example, Singh, Murdoch, and Yu [161] adjust the decomposition rule for affine transformations $f(x) = Wx + b$, so that the bias term does not affect the relative magnitudes of β and γ:

$$\beta_i = W\beta_{i-1} + \frac{|W\beta_{i-1}|}{|W\beta_{i-1}| + |W\gamma_{i-1}|} \cdot b, \tag{4.4}$$

$$\gamma_i = W\gamma_{i-1} + \frac{|W\gamma_{i-1}|}{|W\beta_{i-1}| + |W\gamma_{i-1}|} \cdot b. \tag{4.5}$$

We succinctly represent the above equations as $\beta_i, \gamma_i = LinearDecomp(\beta_{i-1}, \gamma_{i-1})$, where W and b are understood to be tunable parameters of a neural network module.

Given the query, key, and value functions mentioned in Section 4.3 are linear transformations, the only module not accounted for in the context of transformers is the self-attention

module. We handle the decomposition of the attention equations as follows:

$$\beta_{query}, \gamma_{query} = LinearDecomp(\beta_{in}, \gamma_{in}) \tag{4.6}$$
$$\beta_{key}, \gamma_{key} = LinearDecomp(\beta_{in}, \gamma_{in}) \tag{4.7}$$
$$\beta_{value}, \gamma_{value} = LinearDecomp(\beta_{in}, \gamma_{in}) \tag{4.8}$$
$$\beta_{attention} = \beta_{key}^T \beta_{query} / \sqrt{d_k} \tag{4.9}$$
$$\gamma_{attention} = ((\beta_{key} + \gamma_{key})^T (\beta_{query} + \gamma_{query}) / \sqrt{d_k}) - \beta_{attention} \tag{4.10}$$
$$\beta_{probs} = Softmax(Mask(\beta_{attention})) \tag{4.11}$$
$$\gamma_{probs} = Softmax(Mask(\beta_{attention} + \gamma_{attention})) - \beta_{probs} \tag{4.12}$$
$$\beta_z = \beta_{probs} * \beta_{value} \tag{4.13}$$
$$\gamma_z = (\beta_{probs} + \gamma_{probs}) * (\beta_{value} + \gamma_{value}) - \beta_z \tag{4.14}$$
$$\beta_{output}, \gamma_{output} = LinearDecomp(\beta_z, \gamma_z) \tag{4.15}$$

where above $Mask$ represents a causal mask in appropriate architectures and $Softmax$ is the softmax along the same dimension as in a standard implementation of self-attention.

Although in the original setting of CD, "input" and "output" refer to an input sequence x and a model's output logits, the method generalizes to arbitrary *source* and *target* activations in the model. Furthermore, in the original context, the "relevant" portion of the decomposition was equal to 1 on a subsequence of the input and 0 elsewhere, but in a mechanistic interpretability context, the CD framework allows us to decompose activations into any choice of β, γ. Thinking of the network again as a computational graph, this gives us a method of calculating the effect of an arbitrary constituent of the activations of an arbitrary node on the activations of another node.

Choosing a decomposition of source nodes for mechanistic interpretability

In a manner analogous to the choice of ablation method elsewhere in mechanistic interpretability, the correct choice of starting decomposition can be subtle, and sometimes specific to the task. A general principle is that the relevant constituent of the activations at some node is the constituent that we hypothesize has a counterfactual effect on the task; often, this means the deviation from the mean activations over some distribution. (It follows that the irrelevant constituent of the activation at this node is the mean activation, which matches intuitions about mean-ablation.) We provide details on the evaluation tasks and name the distributions used for ablation in Appendix A.3.

Automating circuit discovery

Definitions of circuit discovery

Before delving into our method, we first formally define a *circuit*. To maintain consistency in notations with prior work [158, 37], we view a model as a computational graph M, where

nodes are activations of the model components in its forward pass (e.g. attention heads and MLPs) and edges are the interactions between those components, and a circuit C is a subgraph of M responsible for certain behavior of interest. Naturally, the first step of circuit discovery is to define a *task* of interest to investigate. This requires specifying a dataset and a task-specific metric to measure the performance of M and C, typically, by maximization of the task metric. Given an input x, similarly as to how the entire model defines a function $M(x)$ from inputs to logits, we also associate each circuit C with a function $C(x)$, defined by ablating away the effect of all components in $M \backslash C$ (i.e. the components not included in C). Circuit discovery may be performed on a single input example or, likelier, the result may be averaged over multiple input examples to reduce variance. Typically we find averaging across 10-100 samples, depending on the tasks being studied, can yield stable performance. Detailed information about which models are studied, input sample counts, and how the task metrics are defined is found in Appendix A.3. Finally, note that our method finds the circuit of interest but does not automatically generate an interpretation of it, however, known interpretability techniques [193] can be used to do so.

Method

Here we present our algorithm for discovering circuits with CD-T. Although it is possible to perform CD at arbitrary granularity and on arbitrary components of a transformer (e.g., MLPs, query/key/value vectors), in this paper we focus solely on finding circuits consisting of attention heads in a transformer to be comparable to prior work, and our unit of analysis is either *the output of an attention head* or *the output of an attention head at a specific sequence position*. For clarity, our method focuses on including or excluding specific nodes from a computational graph, rather than edges; CD-T at once implicitly models both direct and indirect effects of one node to another, and our algorithm does not make a distinction between the two.

In each iteration, we search over potential source nodes to find the ones with highest relevance to a set of target nodes (initialized to the task objective). Once we find the set of nodes which are most relevant to the task, we prune them by some simple heuristic (in our experiments, simply greedily removing the nodes with the least magnitude of impact was often sufficient, or sometimes pruning was not necessary), and designate these most important nodes the target nodes for the next iteration of the algorithm. We halt when the collection of nodes achieves adequate performance (e.g, comparable to the whole network), or has not improved from the previous iteration.

We need to specify how "highest relevance to a set of target nodes" is determined. As described above, we start with a decomposition β_s, γ_s of the output of a source attention head s, and we are interested in its relevance $R(s, T)$ to a set of target nodes T. Starting from s, we propagate the decomposition forward through the network using the CD-T equations for each module in the computational graph between s and T; again, if there is some series of functions $f_n, ...f_1$, so that $T = f_n(f_{n-1}(...f_1(s)))$, and $f_1(\beta_s + \gamma_s) = \beta_1 + \gamma_1, f_2(\beta_1 + \gamma_1) = \beta_2 + \gamma_2, ..., f_n(\beta_{n-1} + \gamma_{n-1}) = \beta_T + \gamma_T$, we can straightforwardly apply the modules in order

to obtain β_T, γ_T. In the case that T is the output of the network, β_T will have matching dimensions, and it is usually appropriate to let $R(s, T)$ be the task-specific objective as evaluated on β_T. (One intuitive instance of this is the case where T is a single "score" whose value we care about, so that β_T is the contribution of s to that score.) However, when T is an arbitrary set of nodes in the network's internals, we must define a proxy metric for the relevance of s to T; in this case, we define the relevance of a source node to a set of target nodes to be the sum of the relevances to the target nodes $t \in T$, and in this paper for $R(s, t)$ we choose a quick-to-compute measure of the size of β_t:

$$R(s, T) \coloneqq \sum_{t \in T} R(s, t) = \sum_{t \in T} \frac{\|\beta_t\|_{l_1}}{\|\gamma_t\|_{l_1}}. \tag{4.16}$$

Careful readers may have noticed that this particular metric measures the magnitude of one node's contribution to another, but not the "direction", so that nodes which maximize this metric may actually contribute negatively to the task under investigation. We found in our experiments that this does not preclude us from finding functioning circuits. On the contrary, this allows us to straightforwardly find nodes which have significant negative contribution to a task metric, such as the "Negative Name Mover Heads" in the IOI task. In principle, it is possible that different choices of target relevance metric can be developed to adjust the behavior of the circuit discovery algorithm to avoid (or accentuate) this behavior; we do not explore this in the paper but identify it as a possible avenue for further investigation. Our complete algorithm is described in Algorithm 2, presented in the specific case where we have chosen to decompose our source nodes s so that β_s is the activation's deviation from the mean over some distribution. More details on the heuristics used and a complexity analysis of the algorithm is found in Appendix A.3.

4.4 CD-T excels at identifying circuits responsible for underlying mechanisms efficiently

Experimental setup

We compare the performance of circuits recovered by CD-T with those obtained by two SOTA baselines: ACDC [31] [3] and EAP [172] [4]. SAE-based methods [124, 108, 56] and classic neural network pruning methods are not considered here because they either require model training or focus more on compressing neural networks for faster inference to reduce storage requirements. Furthermore, pruning for interpretability is also shown to be outperformed by ACDC [31]. We evaluate the circuit performance on three tasks: indirect object

[3] We adopt the ACDC optimized for KL diveregence instead of task-specific metrics because it's shown to perform better in Conmy et al. [31].

[4] We consider EAP but not EAP-IG because EAP-IG performs comparably to EAP on the tasks we evaluate [53], but with a runtime increase by a constant factor in theory.

Algorithm 2 Building a circuit using CD-T for a specified task

Input: datapoint x, precomputed mean activations μ
Denote $a_x(s)$ as the activation of s on input x
Denote $\mu(s)$ as the precomputed mean activation of s on an appropriate distribution
Initialize C to store the circuit
Initialize T to be the output of the model
repeat
    for each attention head s upstream of T **do**
        Run the model from x to s
        Compute $\beta_s = a_x(s) - \mu(s)$, $\gamma_s = \mu(s)$
        Propagate decomposition to target T using CD-T to get β_T, γ_T
        Compute relevance $R(s, T)$ using equation 4.16
    end for
    Select a set S of source nodes with highest $R(s, T)$
    $C \leftarrow C \cup S$
    repeat
        for each node n in C **do**
            $C' \leftarrow C \setminus \{n\}$
            if $C'(x) > C(x)$ **then**
                $C \leftarrow C'$
            end if
        end for
    until C has not changed since the last iteration
    if $|C(x) - M(x)| < \epsilon$ **then**
        return C
    end if
    if $C(x)$ has not improved since last iteration **then**
        return C
    end if
    $T \leftarrow S$
until no upstream attention heads for T remain
return C

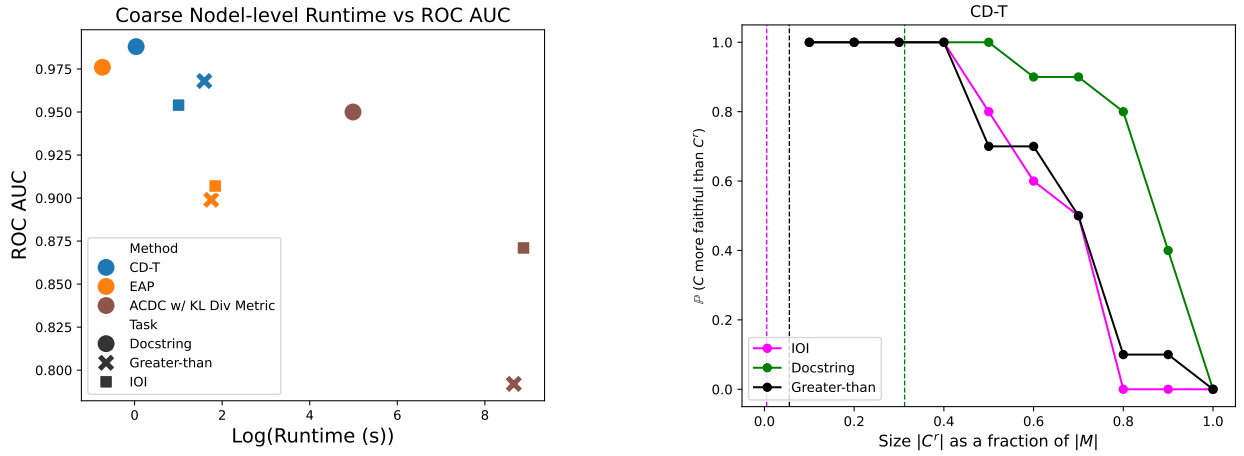


Figure 4.1: Left: Log algorithm runtime and ROC AUC comparison. Each dot represents an average measurement on one task with specific methods differentiated by colors.; Right: The relative faithfulness of CD-T circuits compared to random circuits from the reference distribution of varying sizes (x-axis). Dotted vertical lines indicate the actual size of the circuits. C denotes a CD-T circuit. C^r denotes a random circuit. M denotes the full model.

identification (IOI) [193], greater-than (Greater-than) [52], and docstring completion (Docstring) [58] (see Appendix A.3 for details). They are three standard evaluation tasks to benchmark circuit discovery methods as prior work [52, 193, 58] has *manually* found circuits in language models explaining for those tasks, which often serve as an imperfect reference standard for circuit discovery work [31, 172, 124] due to the lack of ground-truth circuits. For both baselines, we perform node-level patching to be comparable since CD-T is a node-level patching technique. All experiments are conducted on an NVIDIA A100 GPU.

Runtime and the recovery of manual circuits

Leveraging the *manually discovered* circuits as a reference standard, we measure how much overlap there is between the recovered circuits and the reference circuits using ROC AUC. Specifically, we measure the ROC AUC by sweeping through a range of thresholds that determine cutoff scores to keep nodes in a circuit. For CD-T, we test by varying the percentile of top nodes to extract in every iteration, in the range of $[90, 99]$. At the same time, we also compute the average algorithm runtime across runs when sweeping through the thresholds for each method. Although CD-T supports arbitrary granular representations of the circuit, the runtime is measured by building a "coarse node-level" circuits by not splitting into sequence positions to be comparable with EAP and ACDC because EAP doesn't support measuring node importance at sequence positions, and ACDC takes an impractical amount of time to achieve sequence-position splitting. To demonstrate CD-T's capability to identify circuits responsible for underlying algorithm for tasks with great efficiency, we plot the algorithm

runtime against the ROC AUC and report the result in Figure 4.1-Left. From Figure 4.1-Left we can see CD-T outperforms the baselines by achieving high ROC AUC and low runtime at the same time. EAP obtains similar runtime as CD-T but performs slightly worse in recovering *manual* circuits for Greater-than and IOI. Even under this coarse nodel-level abstraction, ACDC is the least efficient as it often takes *hours* to identify a circuit, yet with no better ROC AUC on all tasks compared to other methods. To explain the better ROC AUC performance of CD-T, we find that CD-T is especially good at identifying negative heads and supporting heads, such as the negative name-mover heads and backup name-mover heads in IOI, which other algorithms struggle to do (more discussion in Appendix A.3).

Probability of identified circuits to be more faithful than random circuits

Given a circuit and a task, one can evaluate how well the circuit performs on the task by measuring the task-specific metric, such as the logit difference on IOI. However, in mechanistic interpretability, instead of aiming to identify the best-performing circuits on the task-specific metric, an ideal circuit should be able to *faithfully replicate the full model behavior*. Following prior work [124, 108], faithfulness is defined as the proportion of the full model's performance that a circuit explains, relative to the baseline performance when no specific input information is provided. Given a task, faithfulness is computed as $\frac{m(C)-m(\emptyset)}{m(M)-m(\emptyset)}$, where $m(C)$, $m(\emptyset)$, and $m(M)$ are the average of the task-specific metric performance over the dataset for the circuit, the corrupted model with all heads ablated, and the full model, respectively. To complement the imperfect nature of the *manual* circuits, leveraging faithfulness, here we provide another evidence to validate CD-T's mechanism perservation of the original model. We adopt a circuit hypothesis testing framework proposed in Shi et al. [158] to test whether a circuit preserves the original model's performance by measuring the probability of a circuit C to be more faithful to the original model M, compared to random circuits C^r of the same size sampling from a reference distribution. This test verifies that the circuit is not a simple lucky draw from the distribution of random circuits, and ensures that it is better than at least a fraction q^* of random circuits. In our experiment, we sample random circuits from all possible nodes in the full model, and we repeat the process for 10 times. Figure 4.1-Right shows the results of the best CD-T circuits (as measured by faithfulness and among different granularity) on the 3 tasks. Our best circuit for IOI is from attention heads at specific sequence positions, while for Greater-than and Docstring, they are from attention heads at a coarse level without the splitting. The results show CD-T circuits are 80% ($q^* = 0.8$) more faithful than random circuits of up to around 60% of the original model's size for IOI and Greater-than, and even those of up to 80% of the original model's size for Docstring, with just a small percentage of nodes in the CD-T circuits (0.4% of full model size for IOI, 5.5% for Greater-than, and 31.3% for Docstring). This finding suggests the faithfulness of CD-T circuits is not due to random chance, and that a more fine-grained abstraction of circuits sometimes is necessary to obtain better faithfulness with an extremely small and specific set

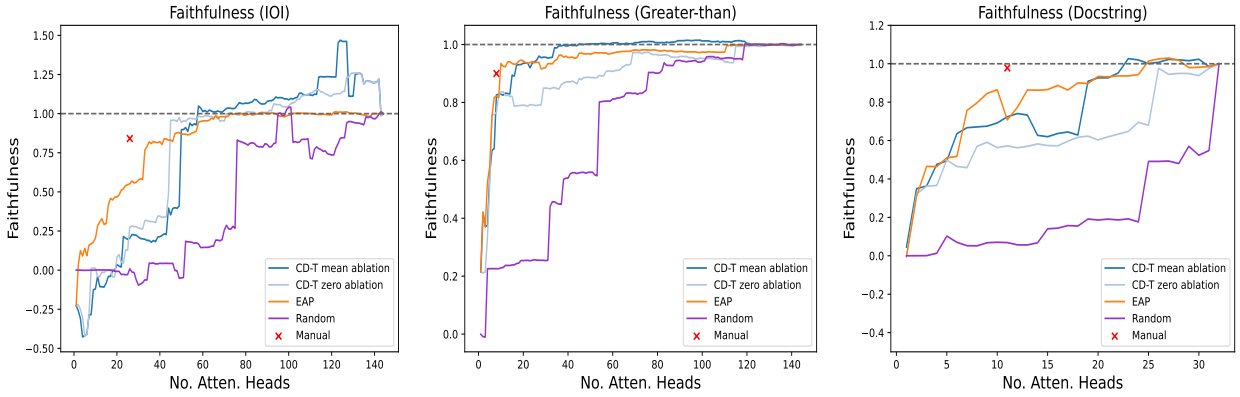


Figure 4.2: Faithfulness of CD-T circuits, EAP circuits, and randomly selected circuits of equivalent size for IOI, Greater-than and Docstring tasks. CD-T circuits obtains the full model's performance (faithfulness of 1) faster than EAP as attention heads are added in order of importance.

of nodes, as in the IOI case.

Identified circuits match full model performance

Faithfulness of circuits of varying sizes

To understand how faithfulness changes as the size of the circuits grows, we compute node importance before pruning for each method [5], and track faithfulness performance as we add circuit nodes in order of importance, while ablating all other nodes with corrupted activations (see Appendix A.3). We sample attention heads in the original model one by one in random order to construct the random baseline, and report the results in Figure 4.2. Since the goal of mechanistic interpretability is to replicate the original model's behavior, namely achieving faithfulness = 1 [6], with the smallest amount of nodes possible, this is naturally the first question we ask. First, we find CD-T circuits are able to perfectly replicate model behavior (faithfulness = 1) using fewer nodes than EAP for all the three tasks. Second, manual circuits outperform both CD-T and EAP of the same size, indicating room of improvement of advanced circuit discovery methods. Finally, we compare the effect of ablation methods on CD-T circuits' performance by including CD-T circuits obtained from zero-ablation in the plot, and find that mean-ablation tend to yield higher faithfulness compared to zero-

[5] ACDC is omitted in this experiment because of the prohibitive amount of runtime in hours to obtain individual node importance.

[6] Faithfulness can go beyond 1 because there are both positively and negatively contributing heads in the model. The original model performance relies on a certain dynamic of interactions between them. When the dynamic is not perfectly replicated and there's extra positively contributing effect, the circuit might give a higher prediction score than the original model would give, which results in a faithfulness > 1.

ablation on Greater-than and Docstring, except for IOI where they are quite comparable. Mean-ablation is often viewed as a preferred way of ablation because it preserves a general concept of the task, hence is less destructive than zero-ablation. Similar finding is also discussed in other circuit discovery work [31, 193].

4.5 Discussion

Although CD-T is broadly applicable to a network's internal components, this paper focused on circuits composed solely of attention heads; extending the approach to include heterogeneous components such as MLPs is a promising direction for future work. Additionally, CD-T currently has two methodological limitations: the relevance metric does not account for the sign of a source node's contribution, potentially valuable in interpretability settings, and some heuristic tuning remains necessary, particularly in selecting components during circuit construction. Despite these limitations, we believe CD-T makes a meaningful contribution to mechanistic interpretability by offering a practical, scalable tool for circuit discovery. We hope that our open-source implementation will enable more fine-grained and efficient analyses, accelerating progress in the field.

Part III

Real-world clinical decision support enhanced by interpretability

Chapter 5

Feature space analysis with clinician input for reliable model selection (SUFO)

5.1 Motivating the need for interpreting fine-tuned feature space for reliable model selection in healthcare

Pre-trained transformer models achieve state-of-the-art performance on a range of NLP tasks [36, 93]. As a consequence, we have witnessed their increasing adoption in the medical domain [201, 211]. While they achieve strong empirical performance, little is understood about how they obtain these results or when they lead to unreliable performance. As recent studies pointed out [144, 15, 156], in clinical settings, deploying language models (LMs) requires more than high performance on benchmark metrics – it demands transparency, robustness, and alignment with domain expertise. In other words, interpretability of the predictions is indispensable for building trust in these models for medical personnel and patients alike.

In this chapter, we propose a systematic framework that provides a practical pipeline for analyzing and interpreting models fine-tuned for particular prediction tasks, focusing on important questions about model trust and interpretability: model suitability for a task, feature space evolution during fine-tuning, and interpretation of fine-tuned features and failure modes. Our framework leverages a suite of analytic and visualization techniques to interpret the feature space of a fine-tuned model.

Distribution shift is a natural consequence of applying general-domain pre-trained models to the medical domain [42]. Mixed-domain pre-trained models that perform continual pre-training with biomedical data partially address this issue and have demonstrated improved performance on several medical tasks [50, 47, 8, 90]. Domain-specific models, which are pre-trained *from scratch* with biomedical data, further alleviate this issue by allowing for

specialized vocabularies better reflective of the medical domain. Although these models yield improved performance, their vulnerability to spelling mistakes resulting from a highly specialized vocabulary and limited pre-training data is well documented [23, 148, 91].

We use SUFO [1] to comprehensively investigate the effects of pre-training data distributions for a real-world pathology report dataset, and further support our findings with a public clinical dataset, MedNLI [149]. We evaluate five pre-trained transformer models (Subsection 5.3) of the same size but differing in pre-training corpora (general-domain/mixed-domain/domain-specific) on our five tasks (Subsection 5.3). In this setting, SUFO helps study the following instantiations of its general targets: (1) how much does in-domain pre-training help? (Section 5.4)? (2) what changes in the feature space during fine-tuning to have led to the differences in model performance (Section 5.5)? (3) how do we interpret the fine-tuned feature space and analyze their failure modes (Sections 5.6)?

We call our approach SUFO and explain below where the name SUFO stands for by making the corresponding letters bold. Each component of SUFO was chosen to yield complementary insights into each of these questions. Firstly, Supervised probing evaluates model features by directly using them for prediction with minimal fine-tuning and sheds light on the suitability of certain pre-trained model for a target task. We show that although pre-trained features in a domain-specific model may contain the most useful information, a domain-specific model can overfit to minority classes after fine-tuning, when presented with class imbalance, while mixed-domain pre-trained models are more resistant to overfitting.

Secondly, Unsupervised similarity analysis and Feature Dynamics visualization study the evolution of the learned feature spaces through fine-tuning and qualitatively disambiguate these models both through their speed of convergence and the degree to which they deviate from the pre-trained initialization. We find the benefit of in-domain pre-training is manifested in faster feature disambiguation; however, the key determinants of model performance are the closeness of pre-training and target tasks and a diverse pre-training data source enabling more robust textual modeling.

Finally, through the substantial sparsification of feature spaces induced by fine-tuning, Outlier analysis, *with clinician validation*, allows for a deeper understanding of the failure modes of these models. We observe that models pre-trained with in-domain data discover a more diverse set of challenging/erroneous reports as determined by a domain expert than a general-domain model.

SUFO may inform the practical use of these models by aiding in the selection of an appropriate pre-trained model, a quantitative and qualitative evaluation of these models through fine-tuning, and finally, an understanding of their failure modes for more reliable deployment.

[1] Code and scripts for running SUFO and experiments are available at `https://github.com/adelaidehsu/path_model_evaluation`.

5.2 Background on language models for clinical tasks and feature analysis

LMs performance on clinical tasks

Prior work has noted the benefits of including biomedical data in the pre-training corpora [201, 47, 8, 90], and the nuances of when and how to include such data [50]. Yet, a comprehensive analysis of the impact of these choices on transformer features remains elusive, and our work aims to provide this understanding to offer improved prescriptive recommendations for practitioners.

In concurrent work, Kefeli and Tatonetti [80] released a model fine-tuned with ClinicalBERT on pathology reports for primary Gleason score extractions. Tai, Kung, and Dong [174] adapts BERT to the medical domain by adding a domain-specific embedding layer and extending the vocabulary. Domain-specific models [48, 145, 96, 9] are proposed to further mitigate the problem of distribution shifts [42] with pre-training using biomedical data only. These models have shown improved performances on biomedical benchmarks [180], and many clinical tasks spanning from medical abstraction [140], drug-target interaction identification [7], to clinical classifications [178, 107]; however, their vulnerability regarding grammatical mistakes is also discussed [23, 148, 91].

Feature analysis in LMs

Most prior works have focused on token feature analysis in unsupervised LM encoders. Supervised probing models, or diagnostic classifiers, are widely used in such works to test features for linguistic phenomena [179, 100, 137] and syntactic structure [62]. With increased flexibility, unsupervised techniques are also proposed to investigate features in the same encoders. SV-CCA [142], a form of canonical correlation analysis, is used in a cross-temporal feature analysis for learning dynamics [151], while PW-CCA [114], an improved version of SV-CCA, is used to analyze transformer features under different pre-training objectives [191]. RSA [85] is increasingly used, such as in investigating the sensitivity of features to context [2], and the correspondence of natural language features to syntax [30]. In addition to the works performed on unsupervised LMs, our work builds on a line of recent works focusing on the fine-tuning effect on BERT for NLU tasks. Peters, Ruder, and Smith [136] discussed the choice of adaptation methods based on the performance of task-specific probing models at various layers. Aken et al. [6] interpreted question-answering models through cluster analysis. Structural probing, RSA and layer ablations are also used in investigating the fine-tuning process of BERT [111], and correlating features of a fine-tuned BERT to fMRI voxel features [43].

Our work improves upon feature analysis since it integrates feature analyses to enable enhanced interpretability of the fine-tuned feature spaces of transformer, and provides insights into the impact of pre-training data on fine-tuned transformers features. This integrated feature analysis pipeline SUFO allows a clearer window through which the inner workings of

fine-tuned LMs become more accessible to domain experts such as clinicians. Such domain expert engagements are indispensable for building trust in LMs and ensuring their safety in medicine.

5.3 Experiment setup

Pre-trained models

We evaluate five 110M-sized [2] encoder-based [3] transformer [188] models commonly used in clinical classifications. Here we describe the models, with an emphasis on their differences in pre-training objectives and categories of pre-training corpora.

General-domain: BERT and TNLR The popular BERT [36] architecture is based on bidirectional transformer encoder [188]. BERT is pre-trained on masked language-modeling (MLM) and next sentence prediction tasks, with a general-domain corpus (3.3B words) from BooksCorpus [220] and English Wikipedia. We use $\text{BERT}_{\text{BASE}}$ with 12 layers and 12 attention heads, and the uncased WordPiece [198] tokenization since prior work [48] has established that case does not have a significant impact on biomedical downstream tasks. The Turing Natural Language Representation (TNLR) model [10] we use has the same architecture and vocabulary as BERT. They do differ, however, in their pre-training objectives, self-attention mechanism, and data as TNLR is trained using constrained self-attention with a pseudo-masked language modeling (PMLM) [10] task on a more diverse general-domain corpus (160GB) that additionally includes OpenWebText[4], CC-News [101], and Stories [183].

Mixed-domain: BioBERT and Clinical BioBERT BioBERT [90] and Clinical BioBERT [8] are categorized as mixed-domain pre-trained models because they are pre-trained with biomedical data on top of a general-domain corpus. The version we use is obtained via continual pre-training from BERT by training on PubMed abstracts (4.5B) for additional steps. Clinical BioBERT is the result of continual pre-training from BioBERT by training additionally on MIMIC-III clinical notes (0.5B) to be more tailored for clinical tasks. The two models share the same vocabulary and architecture as BERT.

Domain-specific: PubMedBERT PubMedBERT [48] was proposed to mitigate the shortcomings in BERT's vocabulary as it cannot represent biomedical terms in full, which was found to possibly hinder the performance of general-domain and mixed-domain models on

[2]Models exhibit only small changes in vocabulary sizes (28996-30522) and the 110M parameter counts include the sizes of the word embeddings.

[3]We focus our discussion on strictly encoder-based transformers by not considering transformers in other architectures (i.e. decoder-only, encoder-decoder) to avoid introducing extra confounding factors.

[4]skylion007.github.io/OpenWebTextCorpus

downstream biomedical tasks [180, 174, 42]. Hence, this model is trained from scratch using PubMed abstracts (3.1B) only, resulting in a more specialized vocabulary for biomedical tasks. We use the uncased version of PubMedBERT with the same architecture as BERT.

Remark on differences in pre-training objectives and data The pre-training objectives and data sizes are similar for all the BERT-based models and we do not expect these differences to impact our findings. While TNLR has a different objective and self-attention mechanism which could confound our analysis, we find that the quantitative and qualitative behavior observed in its analysis in relation to the mixed and domain-specific models are similar to BERT, the other general-domain model. Thus, we believe that our conclusions are applicable to TNLR despite these differences.

Fine-tuning Data

Prostate cancer pathology reports We collected a corpus of 2907 structured pathology reports with data elements extracted from a set of free-text reports following a previously proposed preprocessing pipeline [126]. The corpus includes pathology reports for patients that had undergone radical prostatectomy for prostate cancer at the University of California, San Francisco (UCSF) from 2001 to 2018. This study was conducted under an institutional review board (IRB) approval. The reports contain an average of 471 tokens. For each document, we focus on the following 4 pathologic data elements: primary Gleason grade (Path-PG), secondary Gleason grade (Path-SG), margin status for tumor (Path-MS), and seminal vesicle invasion (Path-SV), and formed 4 classification tasks correspondingly. (Detailed description in Appendix A.4) For Path-PG and Path-SG, there are 5 labels available: `[null, 2, 3, 4, 5]`, with `null` denoting an undecided Gleason score, often due to previous treatment effects. We exclude reports with `null` and `2` Gleason scores under a doctor's suggestion as the two labels account for only 1.3% and 0.07% of the corpus, and are rarely graded in practice. [5] After the removal, the distribution of labels `3`, `4`, and `5` in Path-PG is 67%, 30%, and 3% respectively, while in Path-SG it is 39%, 53%, and 8%. Both Path-MS and Path-SV are binary classification tasks, with only two labels: `[positive, negative]`. The distribution of `positive` and `negative` in Path-MS is 26% and 74%, while in Path-SV is 13% and 87%. Our pathology reports dataset is not publicly available due to the protected patient information in the dataset; however, we provide a few anonymized report samples in Appendix A.4 as illustration.

MedNLI To support the generalizability of our conclusions, we additionally report the fine-tuning results of the models on a publicly available clinical dataset, MedNLI [149]. The objective of MedNLI is to determine if a given clinical hypothesis can be inferred from a

[5] Previously we tried to include Gleason scores `null` and `2` in the fine-tuning, but found none of the models could classify any of the two classes well due to their extremely small sample sizes. It didn't seem reasonable to discuss the models' performance on these two classes given that they couldn't even learn well.

given premise, and the dataset is labelled with three classes `[contradiction, entailment, neutral]`. We (non-uniformly) sample subsets of 6990 samples from MedNLI which reflect the different class distributions observed in the pathology report extraction tasks.

See Appendix A.4 for a full description of fine-tuning hyperparameters. Note that random weighted sampling was implemented for all tasks during fine-tuning to tackle the data imbalance.

5.4 How much does in-domain pre-training help?

In this section, we discuss realistic scenarios when in-domain pretraining [6] benefits, and more importantly, hinders, downstream task performances by analyzing performance of the pre-trained models under two most common forms of adaptation: fine-tuning and supervised probing.

Model performance: fine-tuning

We show the fine-tuned model performance on pathology reports in Table 5.1. The models have generally comparable performance on Path-SG, Path-MS, and Path-SV; however, they are distinguished by their performance on Path-PG, where serious data imbalance exists. In Path-PG, BioBERT and Clinical BioBERT still obtain relatively high accuracies, $> 93\%$, while classifying both majority *and* minority classes well (see Appendix A.4 for per-class accuracy). The general-domain models, BERT and TNLR, having accuracies 86% and 76% on Path-PG, show inferior performance to the mixed-domain models. Yet surprisingly, PubMedBERT, as a domain-specific model, also does poorly on Path-PG performing close to the general-domain models. Specifically, we find that while PubMedBERT does well on the majority classes, it struggles with the minority one.

To investigate whether this finding extends outside of our pathology report dataset, we evaluated the fine-tuning performance of PubMedBERT and Clinical BioBERT on MedNLI, where we simulated three scenarios of different class distributions: Balanced, Imbalanced (simulating class distribution in Path-SG), and Highly Imbalanced (simulating class distribution in Path-PG), and report the results in Table A.12. In the Balanced set, PubMedBERT can outperform Clinical BioBERT. However, Clinical BioBERT outperforms PubMedBERT in the Highly Imbalanced set due to PubMedBERT's inability to classify one of the minority groups well, while in the Imbalanced set, both yield comparable performance, corroborating our finding on the pathology reports. Hence, for the feature analyses in the following sections, we will focus on the pathology report dataset.

[6] In-domain pre-training includes both mixed and domain-specific pre-training, as long as biomedical data is included in the pre-training data.

Table 5.1: F1 test set performance over 3 runs. BioBERT and Clinical BioBERT perform the best on average, while PubMedBERT struggles when serious data imbalance present.

Models	Path-PG	Path-SG	Path-MS	Path-SV	Average
BERT	0.858 (0.16)	0.975 (0.02)	0.957 (0.01)	0.908 (0.03)	0.924
TNLR	0.763 (0.18)	0.995 (0.01)	0.963 (0.01)	0.932 (0.01)	0.913
BioBERT	0.933 (0.04)	0.991 (0.01)	0.959 (0.01)	0.915 (0.02)	0.950
Clinical BioBERT	0.959 (0.03)	0.992 (0.01)	0.964 (0.01)	0.920 (0.01)	0.959
PubMedBERT	0.770 (0.12)	0.984 (0.01)	0.970 (0.01)	0.928 (0.01)	0.913

Table 5.2: F1 test set performance under supervised probing over 3 runs. PubMedBERT performs the best, showing its pre-trained feature contains the most useful information for pathology reports.

Models	Path-PG	Path-SG	Path-MS	Path-SV	Average
BERT	0.371 (0.04)	0.345 (0.05)	0.678 (0.03)	0.578 (0.02)	0.493
TNLR	0.271 (0.01)	0.267 (0.06)	0.494 (0.03)	0.481 (0.01)	0.378
BioBERT	0.340 (0.04)	0.327 (0.04)	0.666 (0.03)	0.574 (0.02)	0.477
Clinical BioBERT	0.341 (0.03)	0.336 (0.04)	0.689 (0.02)	0.581 (0.01)	0.487
PubMedBERT	0.387 (0.01)	0.327 (0.03)	0.687 (0.02)	0.575 (0.01)	0.494
Random-BERT	0.339 (0.06)	0.260 (0.09)	0.529 (0.05)	0.555 (0.05)	0.421

Model performance: supervised probing

Supervised probing, where we freeze the pre-trained weights, and only train the last linear layer, is a measure of how much useful information for a downstream task is contained in the pre-trained features [6, 136, 111, 62]. We report the supervised probing performance on pathology reports in Table 5.2. For comparison, we provide baseline results on a randomly initialized BERT (Random-BERT). This normalization is necessary as even random features often perform well in probing methods. [210, 61]. Among all, PubMedBERT achieves the highest average score while the mixed-domain models and BERT, come second with average scores close to PubMedBERT, and TNLR obtains the lowest average score failing to even beat the baseline.

Discussion on effect of in-domain pre-training data

The fine-tuning and supervised probing results shown above demonstrate some subtle effects of in-domain pre-training data when it brings performance gain. That is, even under different degrees of class imbalance, *if* pre-training data is diverse enough to ensure robustness, the

gain persists. PubMedBERT is shown to contain much useful information for our tasks in its pre-trained features, possibly due to its domain-specific pre-training; however, it suffers from instability in predicting the minority class after fine-tuning. Mixed-domain models, such as BioBERT and Clinical BioBERT, not only show good performance on supervised probing, but also perform well after fine-tuning. The benefits of their mixed-domain pre-training are two-fold: first, pre-training on biomedical datasets allows for better domain-specific features more amenable to performance improvements through fine-tuning, and second, the incorporation of general-domain corpus makes them more resistant to overfitting.

5.5 What happens during fine-tuning?

In Section 5.4, we observe how fine-tuning distorts the features in PubMedBERT, making it no longer the most suitable for the pathology classification tasks after fine-tuning. This indicates a significant change in feature space through the fine-tuning process, and we investigate this change across layer and time in this section. We first leverage an unsupervised similarity analysis, to measure similarity of content of neural representations [87] across layers. Next, we explore feature dynamics, along both time and layer axes, through cluster analysis, to examine the structure and evolution of feature disambiguation during fine-tuning. Our work is, to our knowledge, the first to conduct such extensive cluster analysis on text features, as previous studies often focus on either cross-layer or cross-temporal analysis. [6, 191]

Unsupervised representational similarity analysis (RSA): changes in the feature space after fine-tuning

RSA is an unsupervised technique for measuring the similarity of two different feature spaces given a set of control stimuli. It was first developed in neuroscience [85], and has been increasingly used to analyze similarity between neural network activations [111, 2, 30]. To conduct RSA, a common set of n samples is used to create two sets of features from two models separately. For each feature set, a pairwise similarity matrix in $\mathbb{R}^{n \times n}$ is calculated with a defined distance measure. The final similarity score between the two feature spaces is computed as the Pearson correlation between the flattened upper triangular sections of the two pairwise similarity matrices. In our work, we sample random reports ($n = 1000$) from our dataset for each of the four tasks as the control stimuli. We extract activations of corresponding encoder layers at the classification token from the two versions, e.g. pre-trained vs. fine-tuned, of each model as the feature sets to compare, in an effort to examine the layer-wise change brought by the fine-tuning process. We use Euclidean distance as the defined distance measure to calculate the pairwise similarity matrix.[7]

[7] We experimented with both Euclidean distance and cosine similarity, and in practice not much difference was observed between the results.

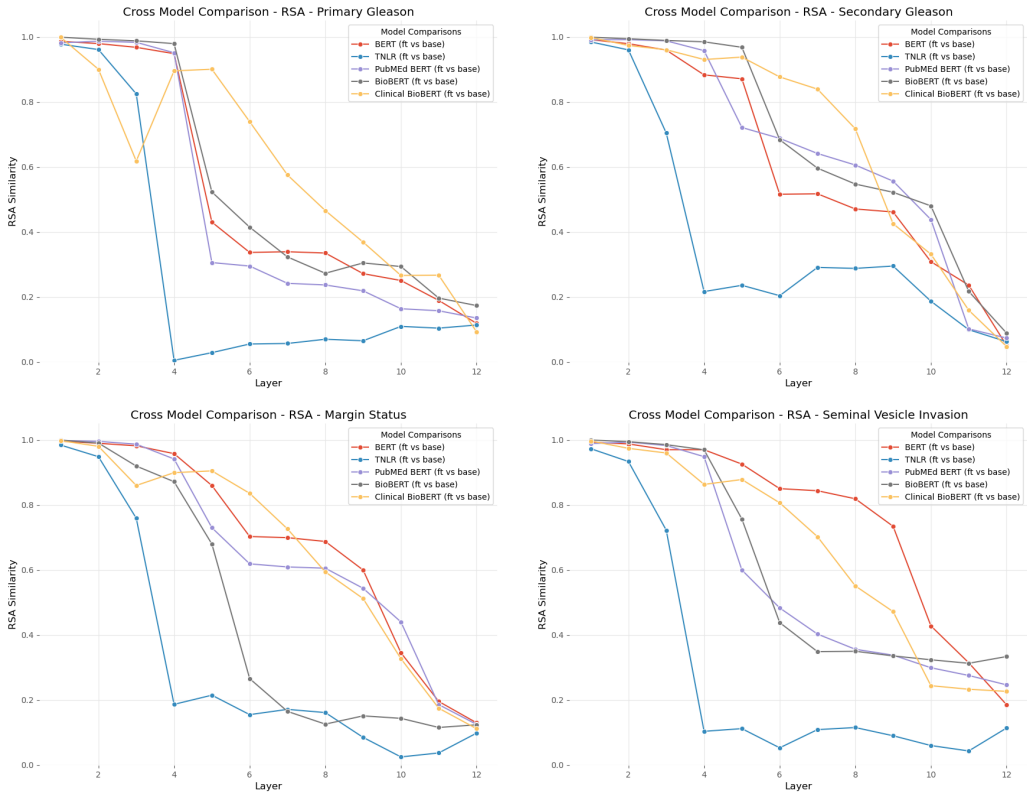


Figure 5.1: Layer-wise RSA comparing the pre-trained and fine-tuned versions of the models across four pathology classification tasks.

Results Figure 5.1 shows our RSA results comparing the pre-trained and fine-tuned versions of each of the five models. In the figure, lower values imply greater change relative to the pre-trained model. We observe a few common trends across all tasks. First, the changes generally arise in the middle layers of the network, and increase in the layers closer to the loss, with little change observed in the layers closest to the input, possibly due to vanishing gradient. Second, Clinical BioBERT on average has the smallest change across layers, or retains the most pre-trained information, while TNLR undergoes the most drastic reconfiguration, suggesting Clinical BioBERT having the pre-trained data distribution more aligned to our target task which are less distorted during fine-tuning, while that of TNLR is the most distant[8]. On average, BERT, BioBERT and PubMedBERT show moderate reconfiguration in the layers, which especially indicates the versatility of BERT's feature space for its ability to match models pre-trained using in-domain data with relatively little reconfiguration.

[8]See Appendix A.4 for a quantitative definition of the closeness between pre-training data and target data.

Feature dynamics: cluster analysis across layer and time

We use PCA to investigate feature dynamics in the models. We examine the structure and evolution of feature disambiguation during fine-tuning across two axes: layer and time. By examining whether feature disambiguation coincides with layers shown to change the most in Section 5.5, we are able to discuss how much of the change actually translates into useful information for the tasks. We extract activations of corresponding encoder layers at the classification token across all 25 checkpoints as feature sets used in this experiment.

Results Due to space limitations, we present test set feature dynamics of the five models in Appendix A.4, where we include the results from Path-PG, as we observe similar results across all the four tasks. When comparing the feature dynamics with RSA results in Section 5.5, we observe that the change in layers measured using RSA typically translate into useful information for target tasks, as we see feature disambiguation in layers often coincides with the significant drop in RSA scores for all the five models. The feature dynamics of TNLR is generally the most dissimilar with the rest. For example, in Path-PG, TNLR disambiguates the minority class, `5`, first starting in layer 4, and then the majority classes, `3` and `4`, starting in layer 7; however, we observe the opposite behavior in the remaining BERT-based models, where they disambiguate the majority class before the minority class. We suspect this difference results from the pre-training objectives and self-attention mechanisms. Examining the feature dynamics through time, we do see models leveraging in-domain pre-training, such as BioBERT, Clinical BioBERT, and PubMedBERT, disambiguate faster than general-domain models. In Path-PG, in-domain models start to disambiguate the classes at around epoch 6, while BERT and TNLR do so around epoch 9. Overall, Clinical BioBERT requires the fewest change in layers and less training epochs to disambiguate the features well. The mixing of classes shown in PubMedBERT's scatterplots on Path-PG, which was not observed in its train set feature dynamics, corroborates the overfitting problem that we see in its fine-tuning performance. From the result, we argue that the effect of in-domain pre-training is manifested in less fine-tuning epochs needed, but the quality of final feature disambiguation really affecting a model's performance is dependent on the closeness of pre-training and target tasks, and the model's resistance against overfitting, as shown in Clinical BioBERT's fast and clear feature disambiguation.

5.6 Interpretation of the fine-tuned feature space

In this section, we perform outlier analysis on the fine-tuned feature spaces of the models yielding insights into their failure modes. Our first observation is that the feature spaces undergo extensive sparsification through fine-tuning. Subsequently, we leverage this structure to identify outlier reports in the feature space and solicit expert evaluation to determine the causes of this behavior. We demonstrate that different pre-trained models exhibit qualita-

tive differences in their outlier modes and SUFO provides useful practical insight into the behavior of these models under fine-tuning.

The structure of the fine-tuned feature space

We analyze principle components (PCs) of features in the final layer classification token of the fine-tuned models. These features are important as they are used directly for prediction, and often contribute the most to performance in ablation studies [111, 6].

High sparsity We first show that the fine-tuned last layer classification token feature space is highly sparsified. We observe that, for every model across the four pathology tasks, the first two PCs explain on average 95% of the variance in the dataset. To understand how the PCs contribute to model performance, we conduct a PC probing experiment (see Appendix A.4). In the experiment, we measure model performance on reconstructed rank-k feature space by projecting onto the bottom k PCs, with k varying between 1 and 768. In particular, $k = 768$ corresponds to the full-feature space. In the PC probing result, we see the first 2 PCs contribute significantly to model performance from the surge in the performance after adding them back in at $k = 767$ and $k = 768$.

Outlier extraction The sparsified low-dimensional structure now allows us to inspect and interpret the fine-tuned features. As we will see, we do this by interpreting outlier reports that do not conform to the *typical* behavior of an input report. To extract these outliers, we construct clusters of the training set[9] on the two-dimensional singular subspace of the feature space. We observe a strong clustering phenomenon where most samples cluster based on their labels (see Figure 5.2). The main difficulties in extracting these clusters are that the one-dimensional projections onto PC1 and PC2 often exhibit significant differences in scale and distribution across the four tasks (Figure 5.2b) and furthermore, the cluster structure itself is also correspondingly different (Figure 5.2b) across tasks. To address this, we first independently extract clusters for the one-dimensional projections onto PC1 (either 2 or 3 depending on the number of labels) and PC2 (either 1 or 3). This produces intervals $\{I_{i,1}\}_{i=1}^{m_1}$ and $\{I_{i,2}\}_{i=1}^{m_2}$ where $m_1, m_2 \in [3]$ for PC1 and PC2 respectively. The clusters in the top-2 singular subspace are obtained by taking the cross product of all pairs of 1-dimensional clusters to obtain $m_1 \times m_2$ rectangles $\{I_{i,1} \times I_{j,2}\}_{i\in[m_1],j\in[m_2]}$. From these, the 2 or 3 (again depending on the number of labels) rectangles with the most datapoints are selected to obtain the final set of 2-dimensional clusters (represented by the red rectangles in Figure 5.2e). This process is illustrated in Figure 5.2. Finally, we extract as outliers all reports which do not fall into any of the clusters.

[9]We choose the training set here to ensure there are a sufficient number of datapoints to reliably recover the PCs. This, however, limits our ability to examine the test set *generalization* of the models.

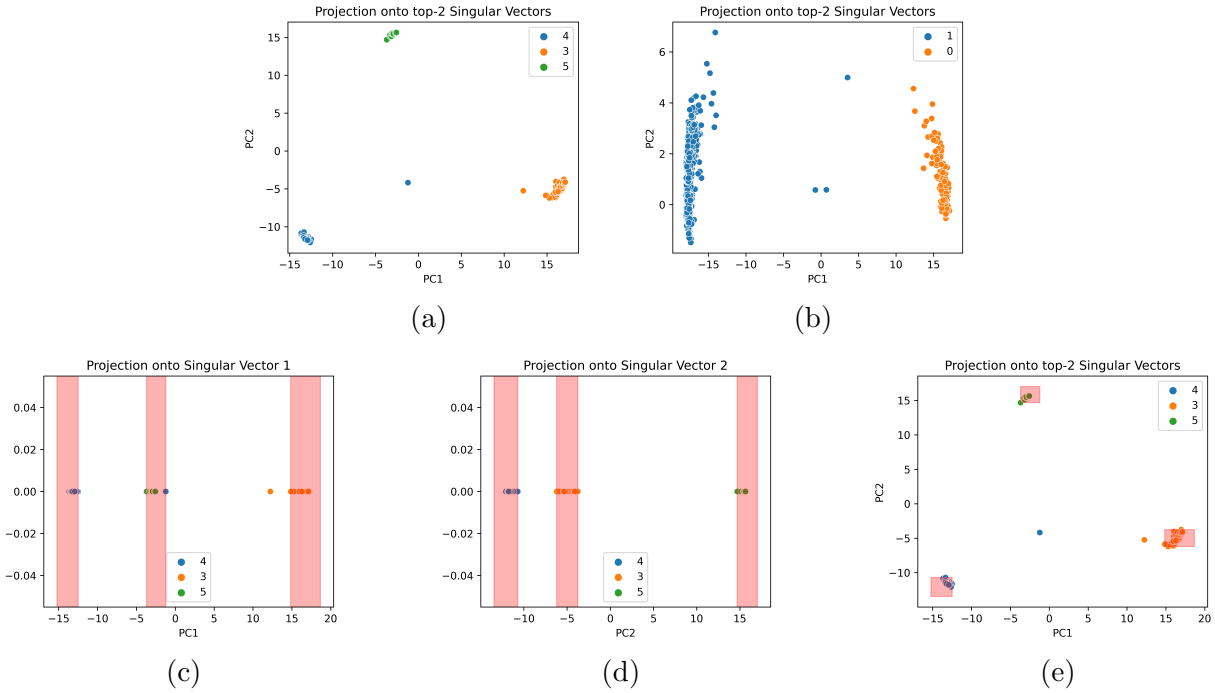


Figure 5.2: Illustration of clustering algorithm: fine-tuned BERT on Path-PG. (a) The projection onto the first two PCs. (b) A similar projection for Path-SV. Notice that the scales of the two PCs are drastically different and a naive clustering based on the Euclidean metric may not capture the variation in PC2. (c) The 3 clusters obtained solely from the projection onto PC1 with each red bar denoting the boundaries of a single cluster. (d) 3 clusters similarly obtained on PC2. (e) The final set of 3 clusters obtained by forming all possible combinations of clusters from (c) and (d) and selecting the three largest.

Domain expert evaluation

After extracting these outliers, we solicited feedback from a domain expert (a clinician in Urology) to attempt to explain their behavior. They were asked whether an outlier report would be challenging for human classification, and if so, explain why. The models were then compared on this feedback.

Common outlier modes We identified the following common outlier modes from our expert-provided feedback. Here, we restrict to the reports identified as being difficult to classify by our expert, henceforth referred to as Hard Outliers: (1) Wrongly labeled reports, (2) Inconsistent reports, (3) Multiple Sources of Information, (4) Not reported or truncated report, and (5) Boundary reports. A full description is provided in Table A.14. The distribution of these classes of outlier reports for each model is provided in Table A.15. The main difference between models is their sensitivity to (4) truncated/unreported instances.

In general, Clinical BioBERT and PubMedBERT identify more instances where the target label is not present than BERT, BioBERT, and TNLR. Hence, the sparsified feature spaces of PubMedBERT and Clinical BioBERT allow for improved detection of missing *medical information* in the pathology reports. We believe the two models extract more comprehensive features that better model the medical data than their general counterparts. On the other hand, features extracted by PubMedBERT are less robust leading to overfitting during fine-tuning. We attribute the inferior performance of the other mixed-domain model, BioBERT, compared to Clinical BioBERT to the lack of clinical data in its pre-training corpus.

5.7 Discussion

In this work, we developed SUFO, a systematic pipeline to shed light on the fine-tuned feature spaces of transformers for increased interpretability by domain practitioners, helping ensure trust in and safety of LMs in critical application domains such as medicine. In our case study investigating the impact of pre-training data on fine-tuned features for clinical note classification, we reveal the robustness of mixed-domain models under substantial class imbalance, that in-domain pre-training helps faster feature disambiguation, and improved identification of missing medical information, validated by an expert evaluation. Although this work takes positive steps towards transparent LMs for medicine, our results are limited in scale and to the setting of the clinical classification tasks. More work is needed to generalize these findings to a broader set of clinical tasks and models. Finally, optimally combining a large general-domain corpus and a smaller domain-specific one for effective pre-training is an important direction for future work.

Chapter 6

Grounding large language model agents in clinical decision rules (CDR-Agent)

6.1 Motivating the need for an intelligent CDR selection and execution system

Building on the foundation established in chapter 5, where feature space analyses and clinician insights were used to evaluate and select trustworthy models, this chapter turns to the next critical question: how can we design intelligent systems that not only support but actively assist clinical decision-making? While SUFO focused on interpretability during model selection, here we explore how interpretability can be embedded within an AI system to guide real-time decisions.

Clinical Decision Rules (CDRs) are standardized tools designed to assist clinicians by combining signs, symptoms, and clinical features into decision trees, enabling accurate and consistent, evidence-based bedside decisions [138]. Currently, over 700 CDRs span nearly all medical specialties, covering diverse clinical scenarios [125, 57]. These rules utilize patient data to generate composite scores or assessments that stratify patient risk for disease onset, progression, or clinical outcomes, helping clinicians deliver expert-level care regardless of their level of experience [64, 27].

However, across healthcare settings, clinicians face increasing time pressure and cognitive burden when evaluating patients. Whether in outpatient clinics, inpatient wards, or telemedicine encounters, clinicians must rapidly assess complex presentations with limited time and incomplete information. In these environments, tools that can surface the right decision aid at the right time can help mitigate errors and democratize care quality. This challenge is even more pronounced in high-stakes trauma care, where rapid and accurate decisions are essential to avoid the harms associated with missed critical injuries or unnecessary imaging and interventions [122]. Although trauma care has been regionalized

to concentrate expertise, injured patients often initially present to emergency departments (EDs) without trauma specialization, further emphasizing the need for universally applicable decision-making aids like CDRs [35, 135].

Despite the importance of CDRs, clinicians often face significant challenges recalling and applying them appropriately due to the sheer number of rules tailored to specific clinical conditions, organ systems, and patient populations [81]. This challenge is not confined to emergency care. Clinicians across healthcare settings routinely make rapid decisions under time constraints, often with incomplete data. Whether in prehospital, outpatient, or inpatient medicine, clinicians must balance efficiency with accuracy, making real-time access to relevant CDRs critically important.

In this chapter, we focus on trauma care as a case study – not because it is exclusive to the ED, but because it uniquely spans multiple disciplines and involves numerous CDRs across different organ systems and clinical specialities (e.g. trauma surgery, orthopedics, critical care, radiology, etc.). This allows us to explore whether an LLM-based agent can support holistic clinical reasoning by recognizing when and how to apply a diverse set of CDRs within a single patient scenario, a challenge that remains largely unmet in current AI systems.

Due to the lack of public benchmarks for validating our CDR-Agent, we evaluate its effectiveness and efficiency on two curated ED datasets: CDR-Bench and a synthetic dataset derived from Pediatric Emergency Care Applied Research Network (PECARN) [65, 86]. The synthetic dataset provides a controlled evaluation setting, where each note is generated from PECARN tabular data and contains a single CDR ground truth label with no missing values. This ensures a structured assessment of model performance in unambiguous clinical scenarios. CDR-Bench consists of real-world clinical notes from various public datasets, annotated with clinician-labeled CDRs. These notes exhibit diverse writing styles, contain noise, and often have missing values, making CDR-Bench a challenging real-world robustness test. CDR-Bench features an adaptive number of CDRs per note, better reflecting real-world clinical decision-making complexity. Together, these datasets offer a comprehensive evaluation of CDR-Agent's ability to interpret clinical narratives and apply trauma-related CDRs accurately. Both datasets are released with code and detailed documentation, ensuring reproducibility and facilitating further research in LLM-driven clinical decision support. [1]

We evaluate CDR-Agent on the two datasets against a baseline approach that directly queries the same LLM for CDR execution results using the clinical notes and a list of available CDRs. Using these datasets, we demonstrated that CDR-Agent not only selects relevant CDRs efficiently, but makes cautious yet effective imaging decisions, minimizing unnecessary interventions while successfully identifying most positively diagnosed cases, outperforming traditional LLM prompting approaches. Moreover, the entire procedure of CDR selection and execution costs 1.22 and 1.16 seconds on the synthetic data and CDR-Bench, respectively, much faster than the baseline with 4.12 and 8.7 seconds on the two datasets.

[1] Code and scripts for running CDR-Agent and the datasets are available at `https://github.com/zhenxianglance/medagent_development`.

6.2 Background on clinical decision support systems (CDSS) using LLMs

Recent advancements have highlighted the integration of Large Language Models (LLMs) into Clinical Decision Support Systems (CDSS), significantly enhancing clinical decision-making processes. LLMs augmented with external medical knowledge, such as literature databases and clinical guidelines, demonstrate superior performance compared to standalone models in various clinical tasks, including COVID-19 outpatient care, medication prescriptions, and diagnostic accuracy [196, 130, 129, 88]. Additionally, explanations generated by LLMs based on patient notes have been shown to improve clinician agreement rates, emphasizing their potential utility in clinical contexts [187]. Nevertheless, existing approaches often suffer from limited generalizability due to their application within narrow, specialized healthcare settings [132]. Particularly relevant to our work is the study by Zakka et al., which introduced an LLM-based agent with external internet access and an extensive database of clinical calculators derived from MDCalc, coupled with a ClinicalQA dataset, achieving substantial performance improvements over plain LLMs [207].

Compared to existing research focused on automated diagnosis [1, 159, 109, 186, 92], knowledge-intensive medical question answering [166, 150, 195], and medication management [129], emergency department decision-making uniquely prioritizes accuracy and efficiency amidst challenges posed by incomplete patient information. Prior studies that augment LLMs with external medical knowledge typically employ retrieval-augmented generation (RAG) [94] methods, which select relevant resources to contextualize the LLM's responses. Although these methods improve accuracy and efficiency relative to standalone models, this alone is insufficient in emergency department scenarios. Existing RAG approaches commonly lack transparency regarding how the LLM integrates retrieved information or verifies the appropriateness of selected resources, limiting interpretability. This absence of clear rationale undermines clinical trust, as emergency clinicians rely heavily on transparent, interpretable decision-making to manage rapid, accurate patient care amidst incomplete and evolving clinical information.

Differing from these studies, our research specifically focuses on a unique clinical decision-making scenario, trauma-related CDR selection and execution in emergency rooms, where both the accuracy and efficiency of the decision-making process are critical.

6.3 Problem description

To set up the task, we consider a clinical note $\boldsymbol{x}$ that describes patient information such as demographics, chief complaints, and physical exam results, and a set of predefined CDRs $\mathcal{C} = \{c_1, \cdots, c_N\}$. Each CDR c_i is a decision tree function that maps from a specific variable space $\mathcal{V}_{c_i}$ to a set of predefined outcomes. Here, the variables are the indicators in the clinical notes required by the CDR to arrive at a decision outcome. For example, the NEXUS criteria for C-spine imaging require five binary indicators: presence of focal neurologic deficit,

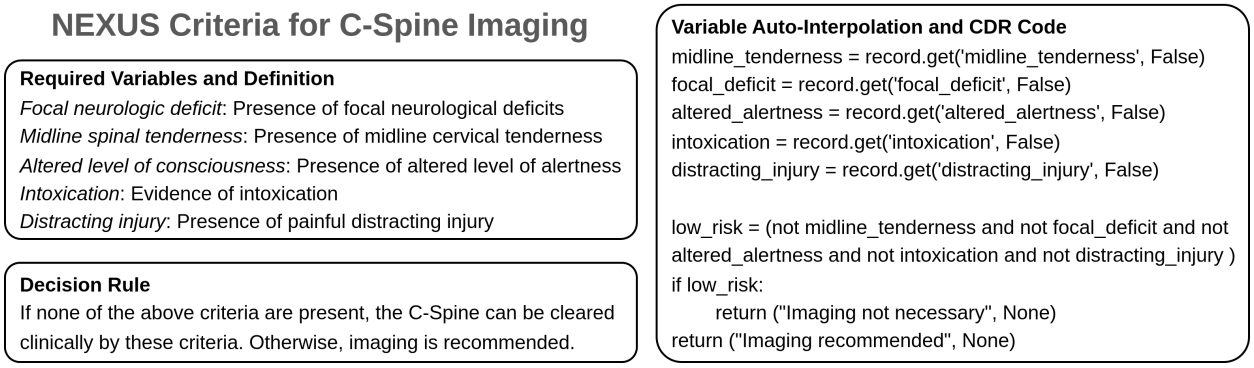


Figure 6.1: An example CDR for C-spine imaging. Top left: the variables/indicators required by the CDR and their definitions. Bottom left: the rule deciding whether the patient requires imaging. Right: the Python script for the CDR with automated imputation of missing variables.

midline spinal tenderness, altered level of consciousness, intoxication, and distracting injury, respectively, to determine an outcome that can either be "imaging recommended" (if any of the indicators present) or "imaging not necessary" (if no indicator is found), as shown in Figure 6.1. Our design objective is to build an automated system that, for any input clinical note $\boldsymbol{x}$, identifies the most appropriate CDRs from $\mathcal{C}$, if there are any, and then execute the identified CDRs to get a set of final decisions.

6.4 CDR-Agent methodology

In this section, we introduce our CDR-Agent framework proposed to address the problem described above, focusing on the design details and underlying rationales.

Overview

Our system design takes into account real-world clinical scenarios with the following challenges. First, clinical notes often vary significantly in writing style and terms, making it difficult to accurately identify relevant indicators or variables for CDRs through simple word matching. Second, some critical information may be incomplete, particularly in cases involving unconscious, disabled, or pediatric patients. Third, the execution of CDRs is prone to errors: both by humans, due to the complexity of the rules, and by automated systems such as LLMs, which may misinterpret textual rule descriptions and generate incorrect decisions.

CDR-Agent follows a three-step workflow designed to address these challenges, and we descibe the deatils of each step in the following paragraphs. The complete pipeline of CDR-Agent is shown in Figure 6.2.

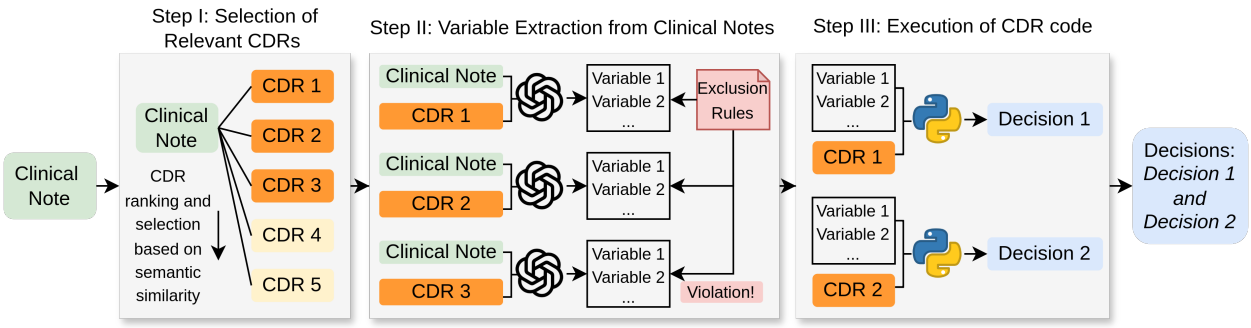


Figure 6.2: Illustration of the three-step workflow of CDR-Agent. For any input clinical note, CDR-Agent first selects a number of relevant CDRs that have high semantic similarity to the clinical note. Variables required by each selected CDR are then extracted from the clinical note using an LLM, with a set of exclusion rules applied to filter invalid CDRs. Finally, CDR-Agent executes the Python code for each valid CDR for decisions.

Step 1: CDR selection

The goal of the first step is to identify the most relevant CDRs for a given clinical note. We use *semantic similarity* as a proxy to measure the relevance between the clinical note and CDRs. To measure semantic similarity, we first converted the CDR functions into textual descriptions using GPT-4o followed by clinician inspection to ensure the correctness, and used a pretrained word embedding model, `text-embedding-ada-002` by OpenAI, to map both the clinical notes and the textual description of CDRs into a shared embedding vector space.

Formally, for each CDR c_i, we compute an embedding[2] vector $E(\boldsymbol{x})$ for the clinical note $\boldsymbol{x}$ and an embedding vector $E(\boldsymbol{t}_i)$ for the textual description $\boldsymbol{t}_i$ of the CDR, respectively. Here, the embedding model is trained by OpenAI on a large corpus of text data, including medical literature, ensuring that semantically similar content is mapped to vectors with higher cosine similarity. A straightforward approach to select CDRs would be to compute the cosine similarity $s^{(i)} = \text{cosim}(E(\boldsymbol{x}), E(\boldsymbol{t_i}))$ for each CDR and select the top-k CDRs with the highest similarity scores for a predefined k. However, setting a value of k or a naive threshold on similarity scores for CDR selection is unrealistic and challenging in practice because of the variability in similarity score ranges across different clinical notes.

As a solution, we propose an anomaly detection approach to identify for each clinical note the CDRs with abnormally large similarity scores for selection in a principled way. Specifically, for each clinical note, we fit a Gaussian distribution using the computed similarity scores. The choice of Gaussian is validated in Section 6.7. A CDR c_i is selected as an anomaly if $p(s^{(i)}|\mu, \sigma^2) > \alpha$, where μ and σ are the estimated mean and variance for the

[2]The actual computation is to map each token in the text to an embedding vector and then take the average embedding vector over the entire text.

```
You are a medical agent designed to apply Clinical Decision Rules (CDRs) to a given clinical note. The CDRs consist of code that takes variable inputs to produce a clinical decision. You will be provided with the variable names, meanings, and types associated with a CDR. Your task is to retrieve the values of these variables from a clinical note based on their descriptions.

Here are the variable descriptions: {variable_descriptions}
Here is the clinical note: {clinical_note}
The retrieved variable values should be in a list with the following format: [variable1: value1, variable2: value2, ...]

If the value of a variable cannot be determined from the clinical note, do not include this variable in the list. Only generate the list without any other information. Variable values:
```

Figure 6.3: Prompt to LLM for variable extraction from a given clinical note. The prompt includes variables required by the selected CDR and their definitions, and the formatting requirements for the extracted variable values.

Gaussian distribution, and α is a predefined significance level. Drawing inspiration from hypothesis testing, we set $\alpha = 0.05$ in all our experiments, which achieves an effective balance between minimizing false positives and enabling the detection of real outliers. If no CDR is identified as an anomaly, it indicates that none of the available CDRs are applicable to the given clinical scenario.

In addition to the above-mentioned base design, we will demonstrate in Section 6.7 that multiple features can be added to the framework to further improve the CDR selection accuracy.

Step 2: Variable extraction

CDR-Agent uses an LLM to extract the variables required by each selected CDR from the given clinical note, with a prompt shown in Figure 6.3. The key to this prompt design is the clear specification of variable *definitions* and *formatting*. For each selected CDR, the prompt includes the definition of each variable (see top-left of Figure 6.1 for example) to help the LLM accurately interpret its meaning while searching for the corresponding values from the clinical note. Additionally, the extracted variables are structured in a list format, where each variable name is followed by its corresponding extracted value.

To handle potential missing values, we include only the determined variables in the output list. In practice, absent variables can be filled in by consulting clinicians for additional information collection from patients. However, in our experiments with *automated* evaluation settings, this feature (which requires human participation) is not incorporated. Instead, we adopt a "negative" imputation approach, where missing variables are assigned default values that do not trigger positive clinical decisions, such as imaging recommendations. This practice prevents the LLM from arbitrarily assigning values that could lead to false positives. For future work, we will explore alternative imputation strategies that leverage external knowledge, such as population-level measurements, to enhance accuracy of CDR

execution. After extracting the variables, a set of exclusion rules, according to the inclusion / exclusion criteria for each CDR, is applied to filter out invalid CDRs, such as excluding adult patients for PECARN CDRs.

Step 3: CDR execution

CDR-Agent executes each valid CDR by running a predefined Python script (see the right of Figure 6.2 for an example Python script), using the variable values extracted from the clinical note in step 2. Before execution, the extracted variable values are converted to their predefined data types, including boolean, integer, float, and string, ensuring consistency and correctness. Formally, the decision outcome for a selected CDR c_i is obtained as $y_i = f_{c_i}(\{v_1^{(i)}, \cdots, v_L^{(i)}\})$, where $\{v_1^{(i)}, \cdots, v_L^{(i)}\}$ are the variable values after formatting and f_{c_i} is the CDR function. Note that any execution failure will trigger an error message, allowing the agent to prompt manual intervention to verify the information processed in earlier steps. This design ensures that only outputs based on accurate variable extraction are produced, minimizing potential errors caused by LLM hallucinations. Once all valid CDRs have been executed, their decisions are aggregated into a final list as the next-step management for the patient.

6.5 Dataset construction

As the first study to explore the automation of CDR selection and execution in ED clinical scenarios, we develop two datasets to evaluate our framework and serve as a benchmark for future research. Collectively comprising 544 patient notes, our dataset significantly surpasses prior related datasets by an order of magnitude.

CDR database Based on clinician recommendations, we selected 15 trauma-related CDRs from peer-reviewed publications to form our CDR database (see Figure 6.4 for a complete list). Each rule was manually transcribed from its original published format into a structured Python function, enabling automated execution within our system. To facilitate accurate interpretation and execution, we also provided detailed textual descriptions for each input feature and for the rule's clinical purpose. This standardized approach ensures that models evaluated on this dataset have sufficient context to correctly select and apply relevant rules. The resulting CDR database was utilized for our experiments on both CDR-Bench and the synthetic dataset, demonstrating the generalizability of our method.

Synthetic dataset PECARN provides Clinical Decision Rules (CDRs) along with corresponding tabular datasets for pediatric traumatic brain injuries (TBI) and pediatric intra-abdominal injuries (IAI). To systematically evaluate our system under controlled conditions, we generated a synthetic dataset by converting tabular patient data from PECARN into realistic free-form clinical narratives. Specifically, we randomly selected a total of 400 patients

(200 TBI and 200 IAI), of which 20% required medical intervention, ensuring representation of clinically significant scenarios. We first use templates to convert binary patient features into structured sentences, then refine these into coherent, natural-sounding clinical notes using GPT-4o. The resulting clinical narratives averaged approximately 98 tokens in length.

CDR-Bench CDR-Bench is constructed using real-world clinical notes from three well-established public datasets: MIMIC-IV [73], MedQA [72], and Augmented Clinical Notes (ACN) [20], each featuring diverse writing styles. To ensure a balanced dataset with both CDR-applicable and non-applicable cases, we filter for notes containing trauma-related keywords (e.g., fracture, injury, trauma). This step prevents an overwhelming number of notes without relevant CDRs, given the large dataset sizes.

We randomly selected 50 trauma-related notes from each source for annotation by four emergency medicine clinicians. Initially, each sample was independently labeled by two clinicians. In cases of disagreement (with an average inter-annotator agreement of 80%), a third, more senior clinician reviewed the discrepancies and provided a final adjudication. Recognizing that clinical decision-making often lacks a single ground truth, we retain all label sets from the annotators. During evaluation, we measure the model's maximum accuracy across these label sets, meaning CDR-Agent is considered correct if it aligns with any clinician's judgment. This approach reflects the inherent variability in medical decision-making while ensuring a robust evaluation of CDR selection performance.

After notes with incomplete information to label were removed, following the clinicians' feedback, the final CDR-Bench consists of 144 samples (MIMIC-IV: 47, MedQA: 47, ACN: 50). On average, each note contains 233.9 tokens. Approximately 36.8% of the notes have no applicable CDRs, ensuring a mix of relevant and irrelevant cases for robustness testing. Among the labeled samples, the average number of CDRs per note is 2.83, highlighting the multi-label nature of clinical decision-making. These facts together make CDR-Bench a challenging dataset compared to the synthetic dataset. See Figure 6.4 for a detailed breakdown of CDR label composition and token length variations across different data sources in CDR-Bench.

6.6 Experiment setup

We aim to evaluate the proposed CDR-Agent with a focus on two research questions: First, can CDR-Agent *accurately* and *efficiently* select all relevant CDRs for a given clinical note? Second, with the incorporation of LLMs, can CDR-Agent *accurately* and *efficiently* extract the required variables from the complex clinical note and execute the CDRs reliably? Note again that efficiency here refers to the computational time cost of the system.

To answer these questions, we use two sets of metrics to assess CDR selection and execution outcomes, respectively. Note that the CDR labeling for a clinical note can result in a single CDR, a set of CDRs, or "no applicable CDR". We evaluate CDR selection using three metrics: 1) **exact match (EA) accuracy**, which measures the proportion of clinical notes

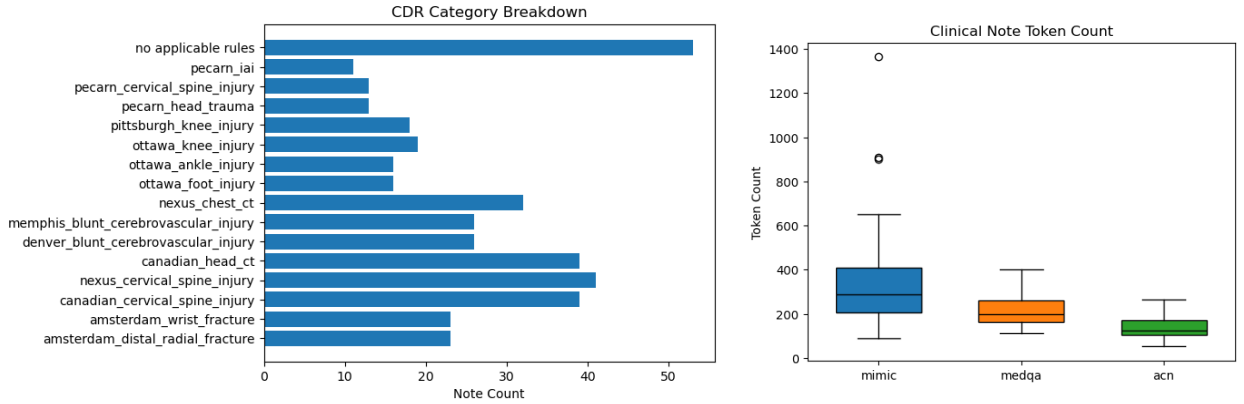


Figure 6.4: (Left) A detailed breakdown of CDR label composition in CDR-Bench. Approximately 36.8% of the notes have no applicable CDRs. (Right) Token length variations across different data sources in CDR-Bench. MIMIC-IV notes are significantly longer than those from MedQA and ACN, often containing more noise and distracting information. This highlights both the diversity captured in CDR-Bench and the challenge it presents for CDR selection.

where the selected CDR(s) exactly match the labeled CDR(s); 2) **F1-score**, which jointly assesses the precision and recall for CDR selection[3]; and 3) **selection time**, which quantifies the computational cost of CDR selection in seconds. For CDR execution, we focus on sensitivity and specificity: 1) **sensitivity**, which measures the sensitivity of outcome recommendation across all *correctly selected* CDRs; 2) **specificity,** which measures the specificity across all *correctly selected* CDRs; and 3) **execution time**, which measures the average time cost for each CDR execution, including both variable extraction and execution of Python scripts of CDRs.

To illustrate the benefits of our design over conventional LLM querying, we compare CDR-Agent with a baseline approach where an LLM is directly queried using the clinical note alongside all the CDR information to determine the final CDR execution outcomes. We report the main results in Table 6.1, where we used one of the state-of-the-art LLMs, GPT-4o, as the core LLM for both approaches. Note that for some other LLM choices, such as GPT-4, the baseline approach easily encounters maximum token limitations due to the extensive prompt length required by the inclusion of all CDRs. This issue will be further amplified as the number of CDRs increases. However, our CDR-Agent is not affected by this issue as it leverages an embedding model to generate embedding vectors to compute similarity scores for CDR selection, without the need of squeezing all CDR information in-context in a query. Under the guidance of the PCS framework [203], in addition to the main

[3]Here, we treat CDR selection for each clinical note as a binary classification problem over the set of all candidate CDRs, including "no applicable CDR". The actual positives are the applicable CDRs (or "no applicable CDR"), and a true positive is counted when a selected CDR matches the ground truth.

Table 6.1: Evaluation results for CDR-Agent on the synthetic data and CDR-Bench, compared with the baseline approach purely based on LLM querying. Evaluation metrics including the exact match (EA) accuracy, F1-score, and time cost (T_{sel}) for CDR selection, and the sensitivity, specificity, and time cost (T_{exe}) for CDR execution. The total time costs (T_{tot}) for both methods are also reported. All time costs are in *second*. "n.a." means "not applicable".

		CDR Selection			CDR Execution			
		EA Accuracy	F1-Score	T_{sel}	Sensitivity	Specificity	T_{exe}	T_{tot}
Synthetic Data	Baseline	0.420	0.735	n.a.	0.818	0.708	n.a.	4.12
	CDR-Agent	0.983	0.994	0.44	0.983	0.687	0.78	1.22
CDR-Bench	Baseline	0.426	0.608	n.a.	0.936	0.652	n.a.	8.70
	CDR-Agent	0.513	0.592	0.52	0.683	0.983	0.64	1.16

Table 6.2: Comparison of various LLM choices for CDR-Agent in CDR execution.

	Synthetic Data			CDR-Bench		
	GPT-4o	GPT-4o-mini	GPT-4	GPT-4o	GPT-4o-mini	GPT-4
Sensitivity	0.983	0.987	0.966	0.683	0.717	0.786
Specificity	0.687	0.583	0.711	0.983	0.967	1.0
T_{exe}	0.78	1.29	3.05	0.64	0.75	1.89

results, we also evaluate CDR-Agent, particularly its CDR execution performance on other LLM choices (as model choices don't affect CDR selection performance) to test the stability of the design, including GPT-4 and GPT-4o-mini, and report the results in Table 6.2.

6.7 CDR-Agent selects relevant CDRs efficiently and makes cautious yet effective decisions

Main evaluation results The comparison between CDR-Agent and the baseline approach for CDR selection and execution is presented in Table 6.1. For CDR selection, on the synthetic dataset, CDR-Agent achieves an exact match accuracy of 0.983 and an F1-score of 0.994, both significantly outperforming the baseline. On CDR-Bench, CDR-Agent achieves a superior exact match accuracy of 0.513 compared to 0.426 for the baseline, while maintaining a comparable F1-score. Additionally, CDR-Agent is highly efficient, requiring only 1.19 seconds on average to complete the entire procedure, compared to 6.41 seconds for the baseline.

For CDR execution, it is important to note that the ground truth "outcome" labels used

to compute sensitivity and specificity differ in nature between the two datasets. In the synthetic dataset (sourced from PECARN tabular data), outcome labels indicate whether a patient was actually diagnosed with TBI or IAI, regardless of whether they received an imaging intervention. However, such diagnosis outcome labels were not available in the clinical notes in our collected CDR-Bench. As a proxy, we use GPT-4o to infer whether a patient received the corresponding imaging intervention recommended by a CDR.

We suspect that this difference in outcome label definitions explains the variation in CDR-Agent's predictive sensitivity and specificity across the two datasets. On CDR-Bench, CDR-Agent achieves high specificity (0.983) and moderate sensitivity (0.683), suggesting a more conservative behavior compared to a vanilla LLM. This conservatism is often due to insufficient information in clinical notes to confidently recommend imaging. It should not be misunderstood as a failure to identify positive diagnoses. In contrast, on the synthetic dataset, where diagnosis labels are directly available, CDR-Agent achieves high sensitivity (0.983) and decent specificity (0.687), indicating strong performance in identifying true positive diagnoses. This high sensitivity is particularly desirable in medical settings, where the primary goal is to avoid missing serious or life-threatening conditions.

Overall, CDR-Agent's performance in CDR execution across both datasets highlights its ability to make cautious yet effective imaging decisions, minimizing unnecessary imaging while still capturing most positively diagnosed cases, outperforming the baseline in both respects. Moreover, CDR-Agent's performance on CDR-Bench improves further when using GPT-4 as the core LLM (Table 6.2), increasing sensitivity to 0.786 and specificity to 1.0, with only a one-second increase in time cost. In contrast, the baseline does not benefit from using GPT-4 due to its limited scalability with the number of CDRs, constrained by model input limits. In terms of computation time, CDR-Agent is over seven times faster than the baseline, demonstrating its efficiency.

In summary, CDR-Agent matches or outperforms the baseline in both CDR selection and execution, while achieving significantly lower computational cost—underscoring its suitability for real-time ED decision-making.

Design choices and further improvements In the CDR selection step, we model the similarity scores of irrelevant CDRs using a Gaussian distribution. To validate this choice, we analyze randomly sampled clinical notes by reviewing the Q-Q plots of similarity scores for all CDRs deemed irrelevant to each note. As illustrated by the example Q-Q plot on the left of Figure 6.5, the similarity scores align closely with the normal reference line (shown in red), showing that the Gaussian distribution is a reasonable choice. However, accurate estimation of the mean and variance of the Gaussian distribution requires a sufficient number of similarity scores. To improve the robustness of CDR selection, we apply repeated random truncations to clinical notes when computing similarity measures. This approach not only increases the number of samples (though dependent) for more reliable estimation of the distribution parameters, but may also reduce redundancy in the notes arising from verbose documentation. As shown on the right of Figure 6.5, we vary the number of ran-

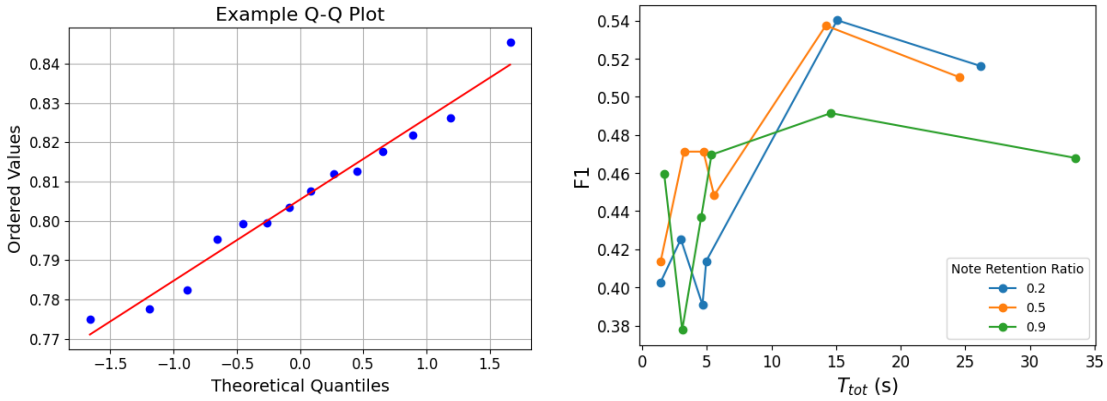


Figure 6.5: (Left) An example Q-Q plot demonstrating that a Gaussian distribution is a reasonable choice for modeling similarity scores of irrelevant CDRs. (Right) Trade-off between F1-score and computation time on a held-out set of CDR-Bench for varying numbers of random sampling iterations and note retention ratios.

dom sampling iterations across [2,5,8,10,30,50] while testing different note retention ratios. This experiment is conducted on a held-out set consisting of 20% of clinical notes randomly sampled from CDR-Bench, using the same CDR-Agent settings as in our main experiments. Across all retention ratios, increasing the number of iterations generally improves CDR selection performance, albeit at the cost of longer computation times. Notably, when focusing on smaller segments of notes (lower note retention ratio), performance improves more sharply as the number of iterations increases. These results suggest that, for real-world CDRs with noisy or verbose documentation, applying a greater number of random truncations, particularly with shorter note segments, can lead to more robust CDR selection, should the time budget allow.

Incorporation of additional knowledge To account for the variability in real-world clinical notes, where medical terms often have alternative phrasings and abbreviations, we explore enhancing CDR-Agent with additional knowledge. Specifically, we augment the textual descriptions of CDRs by appending a list of synonymous terms for key indicators, prefixed with "Keywords to consider often include:". For example, for distal radial fracture, we include alternatives such as DRF, fx distal radius, distal radius fracture, and for wrist swelling, we add swollen wrist, wrist puffiness, enlarged wrist. These keyword expansions were generated using GPT-4o with a one-shot example and later verified by a clinician. Incorporating this additional knowledge improved exact match accuracy by 4% and F1 score by 2% on CDR-Bench. This demonstrates CDR-Agent's flexibility in integrating domain-specific enhancements, making it more robust to linguistic variability in clinical text.

6.8 Discussion

In this chapter, we introduced CDR-Agent, an LLM-based system designed to autonomously select and execute Clinical Decision Rules (CDRs) based on clinical notes. To address the lack of public benchmarks for validating automated CDR selection and execution, we developed two high-quality datasets, laying the groundwork for future research in LLM-driven clinical decision support. Using these datasets, we demonstrated that CDR-Agent not only selects relevant CDRs efficiently, but makes cautious yet effective imaging decisions, minimizing unnecessary interventions while successfully identifying most positively diagnosed cases, outperforming traditional LLM prompting approaches.

While our framework is generalizable across clinical scenarios, we chose to focus on trauma care as a compelling case study due to its multidisciplinary nature and the availability of multiple, organ-specific CDRs. By bridging AI capabilities with clinical expertise, CDR-Agent has the potential to support decision-making not only in specialized trauma centers but also in diverse clinical settings where expertise may be limited. This work represents an important step toward deploying safe, transparent, and context-aware AI tools in frontline multidisciplinary trauma care.

Bibliography

[1] Mahyar Abbasian et al. *Conversational Health Agents: A Personalized LLM-Powered Agent Framework*. 2024. arXiv: 2310.02374 [cs.CL].

[2] Samira Abnar et al. "Blackbox Meets Blackbox: Representational Similarity & Stability Analysis of Neural Language Models and Brains". In: *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 191–203. DOI: 10.18653/v1/W19-4820. URL: https://aclanthology.org/W19-4820.

[3] Eldar David Abraham et al. *CEBaB: Estimating the Causal Effects of Real-World Concepts on NLP Model Behavior*. 2022. arXiv: 2205.14140 [cs.CL]. URL: https://arxiv.org/abs/2205.14140.

[4] Federico Adolfi, Martina G Vilas, and Todd Wareham. "The Computational Complexity of Circuit Discovery for Inner Interpretability". In: *arXiv preprint arXiv:2410.08025* (2024).

[5] Chirag Agarwal, Sree Harsha Tanneru, and Himabindu Lakkaraju. "Faithfulness vs. plausibility: On the (un) reliability of explanations from large language models". In: *arXiv preprint arXiv:2402.04614* (2024).

[6] Betty van Aken et al. "How Does BERT Answer Questions? A Layer-Wise Analysis of Transformer Representations". In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. CIKM '19. Beijing, China: Association for Computing Machinery, 2019, pp. 1823–1832. ISBN: 9781450369763. DOI: 10.1145/3357384.3358028. URL: https://doi.org/10.1145/3357384.3358028.

[7] Jehad Aldahdooh et al. "Using BERT to identify drug-target interactions from whole PubMed". In: *BMC bioinformatics* 23.1 (2022), p. 245.

[8] Emily Alsentzer et al. "Publicly Available Clinical BERT Embeddings". In: *Proceedings of the 2nd Clinical Natural Language Processing Workshop*. Minneapolis, Minnesota, USA: Association for Computational Linguistics, June 2019, pp. 72–78. DOI: 10.18653/v1/W19-1909. URL: https://aclanthology.org/W19-1909.

[9] Sungmin Aum and Seon Choe. "srBERT: automatic article classification model for systematic review using BERT". In: *Systematic reviews* 10.1 (2021), pp. 1–8.

[10] Hangbo Bao et al. "UniLMv2: Pseudo-Masked Language Models for Unified Language Model Pre-Training". In: *CoRR* abs/2002.12804 (2020). arXiv: 2002.12804. URL: https://arxiv.org/abs/2002.12804.

[11] Sumanta Basu et al. "Iterative Random Forests to discover predictive and stable high-order interactions". In: *PNAS* 115(8) (2018), pp. 1943–1948.

[12] David Bau et al. "Gan dissection: Visualizing and understanding generative adversarial networks". In: *arXiv preprint arXiv:1811.10597* (2018).

[13] David Bau et al. "Network dissection: Quantifying interpretability of deep visual representations". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6541–6549.

[14] David Bau et al. "Understanding the role of individual units in a deep neural network". In: *Proceedings of the National Academy of Sciences* 117.48 (2020), pp. 30071–30078.

[15] Suhana Bedi et al. "Testing and evaluation of health care applications of large language models: a systematic review". In: *JAMA* (2024).

[16] Marco Bertolini et al. "Beyond atoms and bonds: contextual explainability via molecular graphical depictions". In: (2022).

[17] Steven Bills et al. *Language models can explain neurons in language models*. https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html. 2023.

[18] Jeffrey R. Binder et al. "Where Is the Semantic System? A Critical Review and Meta-Analysis of 120 Functional Neuroimaging Studies". en. In: *Cerebral Cortex* 19.12 (Dec. 2009), pp. 2767–2796. ISSN: 1460-2199, 1047-3211. DOI: 10.1093/cercor/bhp055. URL: https://academic.oup.com/cercor/article-lookup/doi/10.1093/cercor/bhp055 (visited on 02/16/2022).

[19] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation". In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[20] Antoine Bonnet. *Augmented Clinical Notes Dataset*. Accessed: 2025-03-25. 2023.

[21] L. Breiman et al. *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks, 1984. URL: https://www.routledge.com/Classification-and-Regression-Trees/Breiman-Friedman-Stone-Olshen/p/book/9780412048418.

[22] Leo Breiman. "Random Forests". en. In: *Machine Learning* 45.1 (Oct. 1, 2001), pp. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324.

[23] Keno K Bressem et al. "Highly accurate classification of chest radiographic reports using a deep learning natural language model pre-trained on 3.8 million text reports". In: *Bioinformatics* 36.21 (July 2020), pp. 5255–5261. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btaa668. eprint: https://academic.oup.com/bioinformatics/article-pdf/36/21/5255/36150551/btaa668.pdf. URL: https://doi.org/10.1093/bioinformatics/btaa668.

[24] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].

[25] Oana-Maria Camburu et al. "e-snli: Natural language inference with natural language explanations". In: *Advances in Neural Information Processing Systems* 31 (2018).

[26] Charlotte Caucheteux, Alexandre Gramfort, and Jean-Rémi King. "Deep language algorithms predict semantic comprehension from brain activity". In: *Scientific Reports* 12.1 (2022), p. 16327.

[27] Teresa M. Chan et al. "Making Decisions in the Era of the Clinical Decision Rule: How Emergency Physicians Use Clinical Decision Rules". In: *Academic Medicine* 95.8 (2020), pp. 1230–1237. DOI: 10.1097/ACM.0000000000003099.

[28] Chaofan Chen et al. *This Looks Like That: Deep Learning for Interpretable Image Recognition*. 2019. arXiv: 1806.10574 [cs.LG]. URL: https://arxiv.org/abs/1806.10574.

[29] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. ACM, Aug. 2016, pp. 785–794. DOI: 10.1145/2939672.2939785. URL: http://dx.doi.org/10.1145/2939672.2939785.

[30] Grzegorz Chrupała and Afra Alishahi. "Correlating Neural and Symbolic Representations of Language". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 2952–2962. DOI: 10.18653/v1/P19-1283. URL: https://aclanthology.org/P19-1283.

[31] Arthur Conmy et al. "Towards Automated Circuit Discovery for Mechanistic Interpretability". In: *ArXiv* abs/2304.14997 (2023). URL: https://api.semanticscholar.org/CorpusID:258418244.

[32] Alexis Conneau et al. "What you can cram into a single vector: Probing sentence embeddings for linguistic properties". In: *arXiv preprint arXiv:1805.01070* (2018).

[33] Damai Dai et al. "Knowledge neurons in pretrained transformers". In: *arXiv preprint arXiv:2104.08696* (2021).

[34] Fahim Dalvi et al. "What is one grain of sand in the desert? analyzing individual neurons in deep nlp models". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 6309–6317.

[35] Theony Deshormes et al. "Low-Value Clinical Practices in Pediatric Trauma Care". In: *JAMA Network Open* 7.10 (Oct. 2024), e2440983. DOI: 10.1001/jamanetworkopen.2024.40983.

[36] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[37] Nelson Elhage et al. "A Mathematical Framework for Transformer Circuits". In: *Transformer Circuits Thread* (2021). URL: https://transformer-circuits.pub/2021/framework/index.html.

[38] Nelson Elhage et al. "Toy Models of Superposition". In: *Transformer Circuits Thread* (2022). https://transformer-circuits.pub/2022/toy_model/index.html.

[39] Amir Feder et al. "CausaLM: Causal Model Explanation Through Counterfactual Language Models". In: *Computational Linguistics* 47 (2020), pp. 333–386. URL: https://api.semanticscholar.org/CorpusID:218901061.

[40] Jiahai Feng and Jacob Steinhardt. "How do Language Models Bind Entities in Context?" In: *ArXiv* abs/2310.17191 (2023). URL: https://api.semanticscholar.org/CorpusID:264490454.

[41] Bruce Fischl. "FreeSurfer". In: *Neuroimage* 62.2 (2012), pp. 774–781.

[42] Shang Gao et al. "Limitations of Transformers on Clinical Text Classification". In: *IEEE Journal of Biomedical and Health Informatics* 25 (2021), pp. 3596–3607.

[43] Jon Gauthier and R. Levy. "Linking artificial and human neural representations of language". In: *Conference on Empirical Methods in Natural Language Processing*. 2019.

[44] Atticus Geiger et al. "Causal Abstractions of Neural Networks". In: *Neural Information Processing Systems*. 2021. URL: https://api.semanticscholar.org/CorpusID:235358214.

[45] Robert Geirhos et al. "Don't trust your eyes: on the (un) reliability of feature visualizations". In: *arXiv preprint arXiv:2306.04719* (2023).

[46] Nicholas W. Goldowsky-Dill et al. "Localizing Model Behavior with Path Patching". In: *ArXiv* abs/2304.05969 (2023). URL: https://api.semanticscholar.org/CorpusID:258079237.

[47] Colin Grambow, Longxiang Zhang, and Thomas Schaaf. "In-Domain Pre-Training Improves Clinical Note Generation from Doctor-Patient Conversations". In: *Proceedings of the First Workshop on Natural Language Generation in Healthcare*. Waterville, Maine, USA and virtual meeting: Association for Computational Linguistics, July 2022, pp. 9–22. URL: https://aclanthology.org/2022.nlg4health-1.2.

[48] Yu Gu et al. "Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing". In: *Association for Computing Machinery* 3.1 (Oct. 2021). ISSN: 2691-1957. DOI: 10.1145/3458754. URL: https://doi.org/10.1145/3458754.

[49] Wes Gurnee et al. *Finding Neurons in a Haystack: Case Studies with Sparse Probing*. 2023. arXiv: 2305.01610 [cs.LG].

[50] Suchin Gururangan et al. "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 8342–8360. DOI: 10.18653/v1/2020.acl-main.740. URL: https://aclanthology.org/2020.acl-main.740.

[51] Liberty S Hamilton et al. "Parallel and distributed encoding of speech across human auditory cortex". In: *Cell* 184.18 (2021), pp. 4626–4639.

[52] Michael Hanna, Ollie Liu, and Alexandre Variengien. *How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model*. 2023. arXiv: 2305.00586 [cs.CL]. URL: https://arxiv.org/abs/2305.00586.

[53] Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. *Have Faith in Faithfulness: Going Beyond Circuit Overlap When Finding Model Mechanisms*. 2024. arXiv: 2403.17806 [cs.LG]. URL: https://arxiv.org/abs/2403.17806.

[54] Kelei He et al. "Transformers in medical image analysis". In: *Intelligent Medicine* 3.1 (2023), pp. 59–78. ISSN: 2667-1026. DOI: https://doi.org/10.1016/j.imed.2022.07.002. URL: https://www.sciencedirect.com/science/article/pii/S2667102622000717.

[55] Pengcheng He et al. "DEBERTA: DECODING-ENHANCED BERT WITH DISENTANGLED ATTENTION". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=XPZIaotutsD.

[56] Zhengfu He et al. *Dictionary Learning Improves Patch-Free Circuit Discovery in Mechanistic Interpretability: A Case Study on Othello-GPT*. 2024. arXiv: 2402.12201 [cs.LG]. URL: https://arxiv.org/abs/2402.12201.

[57] J. S. Heerink et al. "Clinical decision rules in primary care: necessary investments for sustainable healthcare." In: *Primary health care research development* 24 (May 2023). DOI: 10.1017/S146342362300021X.

[58] Stefan Heimersheim and Jett Janiak. "A circuit for Python docstrings in a 4-layer attention-only transformer". In: (2023). URL: https://www.alignmentforum.org/posts/u6KXXmKFbXfWzoAXn/acircuit-for-python-docstrings-in-a-4-layer-attention-only.

[59] Lisa Anne Hendricks et al. "Generating visual explanations". In: *European conference on computer vision*. Springer. 2016, pp. 3–19.

[60] Evan Hernandez et al. "Natural language descriptions of deep visual features". In: *International Conference on Learning Representations*. 2022.

[61] John Hewitt and Percy Liang. "Designing and Interpreting Probes with Control Tasks". In: *ArXiv* abs/1909.03368 (2019).

[62] John Hewitt and Christopher D. Manning. "A Structural Probe for Finding Syntax in Word Representations". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4129–4138. DOI: 10.18653/v1/N19-1419. URL: https://aclanthology.org/N19-1419.

[63] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531 [stat.ML]. URL: https://arxiv.org/abs/1503.02531.

[64] James F. Holmes and Nathan Kuppermann. "When Do Clinical Decision Rules Improve Patient Care?" In: *Annals of Emergency Medicine* 63.3 (Mar. 2014), pp. 372–373. DOI: 10.1016/j.annemergmed.2013.09.028.

[65] James F. Holmes et al. "Identifying children at very low risk of clinically important blunt abdominal injuries". In: *Annals of Emergency Medicine* 62.2 (2013), 107–116.e2. DOI: 10.1016/j.annemergmed.2012.11.009.

[66] Or Honovich et al. "Instruction Induction: From Few Examples to Natural Language Task Descriptions". In: *arXiv preprint arXiv:2205.10782* (2022).

[67] Aliyah R Hsu et al. "Efficient Automated Circuit Discovery in Transformers using Contextual Decomposition". In: *arXiv preprint arXiv:2407.00886* (2024).

[68] Aliyah R. Hsu et al. *Diagnosing Transformers: Illuminating Feature Spaces for Clinical Decision-Making*. 2023. arXiv: 2305.17588 [cs.CL].

[69] Alexander G Huth et al. "Natural speech reveals the semantic maps that tile human cerebral cortex". In: *Nature* 532.7600 (2016), pp. 453–458.

[70] Shailee Jain and Alexander Huth. "Incorporating context into language encoding models for fMRI". In: *Advances in neural information processing systems* 31 (2018).

[71] Xiaoqi Jiao et al. *TinyBERT: Distilling BERT for Natural Language Understanding*. 2020. arXiv: 1909.10351 [cs.CL]. URL: https://arxiv.org/abs/1909.10351.

[72] Di Jin et al. "What Disease does this Patient Have? A Large-scale Open Domain Question Answering Dataset from Medical Exams". In: *arXiv preprint arXiv:2009.13081* (2020).

[73] Alistair EW Johnson et al. "MIMIC-IV, a freely accessible electronic health record dataset". In: *Scientific Data* 10.1 (2023), pp. 1–12. DOI: 10.1038/s41597-022-01772-1.

[74] Jaap Jumelet, Willem H. Zuidema, and Dieuwke Hupkes. "Analysing Neural Language Models: Contextual Decomposition Reveals Default Reasoning in Number and Gender Assignment". In: *Conference on Computational Natural Language Learning*. 2019. URL: https://api.semanticscholar.org/CorpusID:202676782.

[75] John M. Jumper et al. "Highly accurate protein structure prediction with AlphaFold". In: *Nature* 596 (2021), pp. 583–589. URL: https://api.semanticscholar.org/CorpusID:235959867.

[76] Akos Kádár, Grzegorz Chrupała, and Afra Alishahi. "Representation of linguistic form and function in recurrent neural networks". In: *Computational Linguistics* 43.4 (2017), pp. 761–780.

[77] Jean Kaddour et al. "Causal Machine Learning: A Survey and Open Problems". In: *ArXiv* abs/2206.15475 (2022). URL: https://api.semanticscholar.org/CorpusID:250144475.

[78] Justin Singh Kang et al. "SPEX: Scaling Feature Interaction Explanations for LLMs". In: *arXiv preprint arXiv:2502.13870* (2025).

[79] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. "Visualizing and understanding recurrent networks". In: *arXiv preprint arXiv:1506.02078* (2015).

[80] Jenna Kefeli and Nicholas Tatonetti. *PrimaryGleasonBERT*. https://huggingface.co/jkefeli/PrimaryGleasonBERT. 2023.

[81] Priti Kharel et al. "Awareness and use of five imaging decision rules for musculoskeletal injuries: a systematic review". In: *International Journal of Emergency Medicine* 16 (Nov. 2023). DOI: 10.1186/s12245-023-00555-4.

[82] Pieter-Jan Kindermans et al. "The (Un) reliability of saliency methods". In: *arXiv preprint arXiv:1711.00867* (2017).

[83] Pang Wei Koh and Percy Liang. "Understanding black-box predictions via influence functions". In: *arXiv preprint arXiv:1703.04730* (2017).

[84] Pang Wei Koh et al. "Concept bottleneck models". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 5338–5348.

[85] Nikolaus Kriegeskorte, Marieke Mur, and Peter Bandettini. "Representational similarity analysis - connecting the branches of systems neuroscience". In: *Frontiers in Systems Neuroscience* 2 (2008). ISSN: 1662-5137. DOI: 10.3389/neuro.06.004.2008. URL: https://www.frontiersin.org/articles/10.3389/neuro.06.004.2008.

[86] Nathan Kuppermann et al. "Identification of children at very low risk of clinically important brain injuries after head trauma: a prospective cohort study". In: *The Lancet* 374.9696 (2009), pp. 1160–1170. DOI: 10.1016/S0140-6736(09)61558-0.

[87] Aarre Laakso and G. Cottrell. "Content and cluster analysis: Assessing representational similarity in neural systems". In: *Philosophical Psychology* 13 (2000), pp. 47–76.

[88] Jacqueline Lammert et al. "Expert-Guided Large Language Models for Clinical Decision Support in Precision Oncology". In: *JCO Precision Oncology* 8 (2024), e2400478. DOI: 10.1200/PO-24-00478.

[89] Amanda LeBel et al. "A natural language fMRI dataset for voxelwise encoding models". In: *bioRxiv* (2022), pp. 2022–09.

[90] Jinhyuk Lee et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4 (Sept. 2019), pp. 1234–1240. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz682. eprint: https://academic.oup.com/bioinformatics/article-pdf/36/4/1234/48983216/bioinformatics_36_4_1234.pdf. URL: https://doi.org/10.1093/bioinformatics/btz682.

[91] Manuel Lentzen et al. "Critical assessment of transformer-based AI models for German clinical notes". In: *JAMIA Open* 5.4 (Nov. 2022). ooac087. ISSN: 2574-2531. DOI: 10.1093/jamiaopen/ooac087. eprint: https://academic.oup.com/jamiaopen/article-pdf/5/4/ooac087/47042094/ooac087_supplementary_data.pdf. URL: https://doi.org/10.1093/jamiaopen/ooac087.

[92] Alessandro Giaj Levra et al. "A large language model-based clinical decision support system for syncope recognition in the emergency department: A framework for clinical workflow integration". In: *European Journal of Internal Medicine* 131 (2025), pp. 113–120. ISSN: 0953-6205. DOI: https://doi.org/10.1016/j.ejim.2024.09.017.

[93] Mike Lewis et al. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703. URL: https://aclanthology.org/2020.acl-main.703.

[94] Patrick Lewis et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks". In: *Advances in neural information processing systems* 33 (2020), pp. 9459–9474.

[95] Aaron J. Li et al. *Improving Prototypical Visual Explanations with Reward Reweighing, Reselection, and Retraining*. 2024. arXiv: 2307.03887 [cs.LG]. URL: https://arxiv.org/abs/2307.03887.

[96] Fei Li et al. "Fine-Tuning Bidirectional Encoder Representations From Transformers (BERT)–Based Models on Large-Scale Electronic Health Record Notes: An Empirical Study". In: *JMIR Medical Informatics* 7 (2019). URL: https://api.semanticscholar.org/CorpusID:202567532.

[97] David Lindner et al. "Tracr: Compiled Transformers as a Laboratory for Interpretability". In: *ArXiv* abs/2301.05062 (2023). URL: https://api.semanticscholar.org/CorpusID:255749093.

[98] Jack Lindsey et al. "On the Biology of a Large Language Model". In: *Transformer Circuits Thread* (2025). URL: https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

[99] Frederick Liu and Besim Avci. "Incorporating Priors with Feature Attribution on Text Classification". In: *arXiv preprint arXiv:1906.08286* (2019).

[100] Nelson F. Liu et al. "Linguistic Knowledge and Transferability of Contextual Representations". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 1073–1094. DOI: 10.18653/v1/N19-1112. URL: https://aclanthology.org/N19-1112.

[101] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *ArXiv* abs/1907.11692 (2019).

[102] Josh Magnus Ludan et al. *Interpretable-by-Design Text Understanding with Iteratively Generated Concept Bottleneck*. 2024. arXiv: 2310.19660 [cs.CL]. URL: https://arxiv.org/abs/2310.19660.

[103] Scott Lundberg and Su-In Lee. "An unexpected unity among methods for interpreting model predictions". In: *arXiv preprint arXiv:1611.07478* (2016).

[104] Scott M Lundberg, Gabriel G Erion, and Su-In Lee. "Consistent Individualized Feature Attribution for Tree Ensembles". In: *arXiv preprint arXiv:1802.03888* (2018).

[105] Scott M Lundberg and Su-In Lee. "A unified approach to interpreting model predictions". In: *Advances in Neural Information Processing Systems*. 2017, pp. 4768–4777.

[106] Scott M Lundberg et al. "Explainable AI for trees: From local explanations to global understanding". In: *arXiv preprint arXiv:1905.04610* (2019).

[107] J Mantas et al. "The classification of short scientific texts using pretrained BERT model". In: *Public Health and Informatics: Proceedings of MIE 2021* 281 (2021), p. 83.

[108] Samuel Marks et al. "Sparse Feature Circuits: Discovering and Editing Interpretable Causal Graphs in Language Models". In: *ArXiv* abs/2403.19647 (2024). URL: https://api.semanticscholar.org/CorpusID:268732732.

[109] Daniel McDuff et al. *Towards Accurate Differential Diagnosis with Large Language Models*. 2023. arXiv: 2312.00164 [cs.CY].

[110] Kevin Meng et al. *Locating and Editing Factual Associations in GPT*. 2023. arXiv: 2202.05262 [cs.CL].

[111] Amil Merchant et al. "What Happens To BERT Embeddings During Fine-tuning?" In: *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*. Online: Association for Computational Linguistics, Nov. 2020, pp. 33–44. DOI: 10.18653/v1/2020.blackboxnlp-1.4. URL: https://aclanthology.org/2020.blackboxnlp-1.4.

[112] Stephen Merity et al. *Pointer Sentinel Mixture Models*. 2016. arXiv: 1609.07843 [cs.CL].

[113] Tom M. Mitchell et al. "Predicting human brain activity associated with the meanings of nouns". eng. In: *Science (New York, N.Y.)* 320.5880 (May 2008), pp. 1191–1195. ISSN: 1095-9203. DOI: 10.1126/science.1152876.

[114] Ari S. Morcos, Maithra Raghu, and Samy Bengio. "Insights on Representational Similarity in Neural Networks with Canonical Correlation". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS'18. Montréal, Canada: Curran Associates Inc., 2018, pp. 5732–5741.

[115] John X Morris et al. "Tree Prompting: Efficient Task Adaptation without Fine-Tuning". In: *arXiv preprint arXiv:2310.14034* (2023).

[116] W. James Murdoch, Peter J. Liu, and Bin Yu. "Beyond Word Importance: Contextual Decomposition to Extract Interactions from LSTMs". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=rkRwGg-0Z.

[117] W. James Murdoch et al. "Definitions, methods, and applications in interpretable machine learning". In: *Proceedings of the National Academy of Sciences* 116.44 (Oct. 2019), pp. 22071–22080. ISSN: 1091-6490. DOI: 10.1073/pnas.1900654116. URL: http://dx.doi.org/10.1073/pnas.1900654116.

[118] Seil Na et al. "Discovery of natural language concepts in individual units of cnns". In: *arXiv preprint arXiv:1902.07249* (2019).

[119] Neel Nanda et al. "Progress measures for grokking via mechanistic interpretability". In: *ArXiv* abs/2301.05217 (2023). URL: https://api.semanticscholar.org/CorpusID:255749430.

[120] Sharan Narang et al. "Wt5?! training text-to-text models to explain their predictions". In: *arXiv preprint arXiv:2004.14546* (2020).

[121] Meike Nauta, Ron van Bree, and Christin Seifert. *Neural Prototype Trees for Interpretable Fine-grained Image Recognition*. 2021. arXiv: 2012.02046 [cs.CV]. URL: https://arxiv.org/abs/2012.02046.

[122] Juliane Neumann et al. "TraumaFlow—development of a workflow-based clinical decision support system for the management of severe trauma cases". In: *International Journal of Computer Assisted Radiology and Surgery* 19 (May 2024), pp. 2399–2409. DOI: 10.1007/s11548-024-03191-2.

[123] Shinji Nishimoto et al. "Eye movement-invariant representations in the human visual system". In: *Journal of vision* 17.1 (2017), pp. 11–11.

[124] Charles O'Neill and Thang Bui. *Sparse Autoencoders Enable Scalable and Reliable Circuit Identification in Language Models*. 2024. arXiv: 2405.12522 [cs.CL]. URL: https://arxiv.org/abs/2405.12522.

[125] Jed Keenan Obra et al. "Systematic Bias in Clinical Decision Instrument Development: A Quantitative Meta-Analysis". In: *medRxiv* (2025). DOI: 10.1101/2025.02.12.25320965. eprint: https://www.medrxiv.org/content/early/2025/02/16/2025.02.12.25320965.full.pdf. URL: https://www.medrxiv.org/content/early/2025/02/16/2025.02.12.25320965.

[126] Anobel Y Odisho et al. "Natural language processing systems for pathology parsing in limited data environments with uncertainty estimation". In: *JAMIA Open* 3.3 (Oct. 2020), pp. 431–438. ISSN: 2574-2531. DOI: 10.1093/jamiaopen/ooaa029. eprint: https://academic.oup.com/jamiaopen/article-pdf/3/3/431/34283168/ooaa029.pdf. URL: https://doi.org/10.1093/jamiaopen/ooaa029.

[127] Tuomas Oikarinen et al. *Label-Free Concept Bottleneck Models*. 2023. arXiv: 2304.06129 [cs.LG]. URL: https://arxiv.org/abs/2304.06129.

[128] Christopher Olah et al. "Zoom In: An Introduction to Circuits". In: 2020. URL: https://api.semanticscholar.org/CorpusID:215930358.

[129] Jasmine Chiat Ling Ong et al. "Development and Testing of a Novel Large Language Model-Based Clinical Decision Support Systems for Medication Safety in 12 Clinical Specialties". In: *arXiv preprint arXiv:2402.01741* (2024). DOI: 10.48550/arXiv.2402.01741.

[130] David Oniani et al. "Enhancing Large Language Models for Clinical Decision Support by Incorporating Clinical Practice Guidelines". In: *Proceedings of the 2022 IEEE International Conference on Big Data (Big Data)* (2022), pp. 2523–2532.

[131] OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

[132] Rani Oomman Panicker and Ankitha Elizabeth George. "Adoption of Automated Clinical Decision Support System: A Recent Literature Review and a Case Study". In: *Archives of Medicine and Health Sciences* 11.1 (2023), pp. 86–95. DOI: 10.4103/amhs.amhs_257_22.

[133] Judea Pearl. *Causality: Models, Reasoning and Inference*. 2nd. USA: Cambridge University Press, 2009. ISBN: 052189560X.

[134] Fabian Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12.85 (2011), pp. 2825–2830. URL: http://jmlr.org/papers/v12/pedregosa11a.html.

[135] Jeffrey J. Perry and Ian G. Stiell. "Impact of clinical decision rules on clinical care of traumatic injuries to the foot and ankle, knee, cervical spine, and head". In: *Injury, Int. J. Care Injured* 37.12 (Dec. 2006), pp. 1157–1165. DOI: 10.1016/j.injury.2006.07.028.

[136] Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. "To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks". In: *Proceedings of the 4th Workshop on Representation Learning for NLP, RepL4NLP@ACL 2019, Florence, Italy, August 2, 2019*. Ed. by Isabelle Augenstein et al. Association for Computational Linguistics, 2019, pp. 7–14. DOI: 10.18653/v1/w19-4302. URL: https://doi.org/10.18653/v1/w19-4302.

[137] Matthew E. Peters et al. "Dissecting Contextual Word Embeddings: Architecture and Representation". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 1499–1509. DOI: 10.18653/v1/D18-1179. URL: https://aclanthology.org/D18-1179.

[138] Jesse M. Pines and Christopher R. Carpenter. "Clinical Decision Rules". In: *Evidence-Based Emergency Care: Diagnostic Testing and Clinical Decision Rules*. Ed. by Jesse M. Pines et al. Third. John Wiley & Sons Ltd., 2023, pp. 43–52.

[139] Nina Poerner, Benjamin Roth, and Hinrich Schütze. "Interpretable textual neuron representations for NLP". In: *arXiv preprint arXiv:1809.07291* (2018).

[140] Sam Preston et al. "Towards Structuring Real-World Data at Scale: Deep Learning for Extracting Key Oncology Information from Clinical Text with Patient-Level Supervision". In: *ArXiv* abs/2203.10442 (2022).

[141] Alec Radford et al. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[142] Maithra Raghu et al. "SVCCA: Singular Vector Canonical Correlation Analysis for Deep Learning Dynamics and Interpretability". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6078–6087. ISBN: 9781510860964.

[143] Nazneen Fatema Rajani et al. "Explain yourself! leveraging language models for commonsense reasoning". In: *arXiv preprint arXiv:1906.02361* (2019).

[144] Inioluwa Deborah Raji, Roxana Daneshjou, and Emily Alsentzer. *It's Time to Bench the Medical Exam Benchmark*. 2025.

[145] Laila Rasmy et al. "Med-BERT: pre-trained contextualized embeddings on large-scale structured electronic health records for disease prediction". In: *CoRR* abs/2005.12833 (2020). arXiv: 2005.12833. URL: https://arxiv.org/abs/2005.12833.

[146] Marco Ribeiro, Sameer Singh, and Carlos Guestrin. ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. Ed. by John DeNero, Mark Finlayson, and Sravana Reddy. San Diego, California: Association for Computational Linguistics, June 2016, pp. 97–101. DOI: 10.18653/v1/N16-3020. URL: https://aclanthology.org/N16-3020.

[147] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Model-agnostic interpretability of machine learning". In: *arXiv preprint arXiv:1606.05386* (2016).

[148] Phillip Richter-Pechanski et al. "Automatic extraction of 12 cardiovascular concepts from German discharge letters using pre-trained language models". In: *DIGITAL HEALTH* 7 (2021). SAGE Publications Sage UK: London, England. URL: https://journals.sagepub.com/doi/full/10.1177/20552076211057662.

[149] Alexey Romanov and Chaitanya Shivade. "Lessons from Natural Language Inference in the Clinical Domain". In: *arXiv:1808.06752 [cs]* (Aug. 21, 2018). arXiv: 1808.06752. URL: http://arxiv.org/abs/1808.06752 (visited on 08/27/2018).

[150] Khaled Saab et al. *Capabilities of Gemini Models in Medicine*. 2024. arXiv: 2404.18416 [cs.AI].

[151] Naomi Saphra and Adam Lopez. "Understanding Learning Dynamics Of Language Models with SVCCA". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 3257–3267. DOI: 10.18653/v1/N19-1329. URL: https://aclanthology.org/N19-1329.

[152] Elvis Saravia et al. "Carer: Contextualized affect representations for emotion recognition". In: *Proceedings of the 2018 conference on empirical methods in natural language processing*. 2018, pp. 3687–3697.

[153] Simon Schrodi et al. "Concept Bottleneck Models Without Predefined Concepts". In: *arXiv preprint arXiv:2407.03921* (2024).

[154] Sarah Schwettmann et al. "Toward a visual concept vocabulary for gan latent space". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 6804–6812.

[155] Lei Sha, Oana-Maria Camburu, and Thomas Lukasiewicz. "Learning from the Best: Rationalizing Predictions by Adversarial Information Calibration." In: *AAAI*. 2021, pp. 13771–13779.

[156] Nigam H Shah, David Entwistle, and Michael A Pfeffer. "Creation and adoption of large language models in medicine". In: *Jama* 330.9 (2023), pp. 866–869.

[157] Matthew Shen et al. "Enhancing CBMs Through Binary Distillation with Applications to Test-Time Intervention". In: *ICLR 2025 Workshop on Building Trust in Language Models and Applications*. 2025. URL: https://openreview.net/forum?id=wBygggbUV8.

[158] Claudia Shi et al. "Hypothesis Testing the Circuit Hypothesis in LLMs". In: *ICML 2024 Workshop on Mechanistic Interpretability*. 2024. URL: https://openreview.net/forum?id=ibSNv9cldu.

[159] Wenqi Shi et al. "EHRAgent: Code Empowers Large Language Models for Few-shot Complex Tabular Reasoning on Electronic Health Records". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 22315–22339. DOI: 10.18653/v1/2024.emnlp-main.1245.

[160] Taylor Shin et al. "Autoprompt: Eliciting knowledge from language models with automatically generated prompts". In: *arXiv preprint arXiv:2010.15980* (2020).

[161] Chandan Singh, W James Murdoch, and Bin Yu. "Hierarchical Interpretations for Neural Network Predictions". en. In: *International Conference on Learning Representations* (2019), p. 26. URL: https://openreview.net/forum?id=SkEqro0ctQ.

[162] Chandan Singh et al. "Augmenting Interpretable Models with LLMs during Training". In: *arXiv preprint arXiv:2209.11799* (2022).

[163] Chandan Singh et al. *Explaining black box text modules in natural language with language models*. 2023. arXiv: 2305.09863 [cs.AI]. URL: https://arxiv.org/abs/2305.09863.

[164] Chandan Singh et al. "Explaining patterns in data with language models via interpretable autoprompting". In: *arXiv preprint arXiv:2210.01848* (2022).

[165] Chandan Singh et al. "imodels: a python package for fitting interpretable models". In: *Journal of Open Source Software* 6.61 (2021), p. 3192. DOI: 10.21105/joss.03192. URL: https://doi.org/10.21105/joss.03192.

[166] Karan Singhal et al. "Large language models encode clinical knowledge". eng. In: *Nature (London)* 620.7972 (2023), pp. 172–180. ISSN: 0028-0836.

[167] Richard Socher et al. "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1631–1642. URL: https://www.aclweb.org/anthology/D13-1170.

[168] Sam Spiro. *FOURIER ANALYSIS OF BOOLEAN FUNCTIONS*. 2016. URL: https://math.uchicago.edu/~may/REU2016/REUPapers/Spiro.pdf.

[169] Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. "A Mechanistic Interpretation of Arithmetic Reasoning in Language Models using Causal Mediation Analysis". In: *Conference on Empirical Methods in Natural Language Processing*. 2023. URL: https://api.semanticscholar.org/CorpusID:258865170.

[170] Hongjin Su et al. "One Embedder, Any Task: Instruction-Finetuned Text Embeddings". In: *arXiv preprint arXiv:2212.09741* (2022).

[171] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. "Axiomatic attribution for deep networks". In: *ICML* (2017).

[172] Aaquib Syed, Can Rager, and Arthur Conmy. *Attribution Patching Outperforms Automated Circuit Discovery*. 2023. arXiv: 2310.10348 [cs.LG]. URL: https://arxiv.org/abs/2310.10348.

[173] Christian Szegedy et al. *Rethinking the Inception Architecture for Computer Vision*. 2015. arXiv: 1512.00567 [cs.CV]. URL: https://arxiv.org/abs/1512.00567.

[174] Wen-Hsin Tai, H. T. Kung, and Xin Dong. "exBERT: Extending Pre-trained Models with Domain-specific Vocabulary Under Constrained Training Resources". In: *Findings*. 2020.

[175] Sarah Tan et al. "Distill-and-compare: Auditing black-box models using transparent model distillation". In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. 2018, pp. 303–310.

[176] Yan Shuo Tan et al. "Fast Interpretable Greedy-Tree Sums". In: *Proceedings of the National Academy of Sciences* 122.7 (2025), e2310151122. DOI: 10.1073/pnas.2310151122. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.2310151122. URL: https://www.pnas.org/doi/abs/10.1073/pnas.2310151122.

[177] Jerry Tang et al. "Semantic reconstruction of continuous language from non-invasive brain recordings". In: *Nature Neuroscience* (2023), pp. 1–9.

[178] Ali S Tejani et al. "Performance of Multiple Pretrained BERT Models to Automate and Accelerate Data Annotation for Large Datasets". In: *Radiology: Artificial Intelligence* 4.4 (2022), e220007.

[179] Ian Tenney et al. "What do you learn from context? Probing for sentence structure in contextualized word representations". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=SJzSgnRcKX.

[180] Robert Tinn et al. "Fine-tuning large neural language models for biomedical natural language processing". In: *Patterns* 4.4 (2023), p. 100729. ISSN: 2666-3899. DOI: https://doi.org/10.1016/j.patter.2023.100729. URL: https://www.sciencedirect.com/science/article/pii/S2666389923000697.

[181] Hugo Touvron et al. "Llama 2: Open foundation and fine-tuned chat models". In: *arXiv preprint arXiv:2307.09288* (2023).

[182] Hugo Touvron et al. "Llama: Open and efficient foundation language models". In: *arXiv preprint arXiv:2302.13971* (2023).

[183] Trieu H. Trinh and Quoc V. Le. "A Simple Method for Commonsense Reasoning". In: *ArXiv* abs/1806.02847 (2018).

[184] Michael Tsang, Dehua Cheng, and Yan Liu. "Detecting statistical interactions from neural network weights". In: *arXiv preprint arXiv:1705.04977* (2017).

[185] Doris Y. Tsao et al. "Patches of face-selective cortex in the macaque frontal lobe". eng. In: *Nature Neuroscience* 11.8 (Aug. 2008), pp. 877–879. ISSN: 1546-1726. DOI: 10.1038/nn.2158.

[186] Tao Tu et al. *Towards Conversational Diagnostic AI*. 2024. arXiv: 2401.05654 [cs.AI].

[187] D. Umerenkov, G. Zubkova, and A. Nesterov. "Deciphering Diagnoses: How Large Language Models Explanations Influence Clinical Decision Making". In: *arXiv preprint arXiv:2310.01708* (2023). DOI: 10.48550/arXiv.2310.01708.

[188] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[189] Deepika Vemuri, Gautham Bellamkonda, and Vineeth N Bellamkonda. *Enhancing Concept-based Learning with Logic*. 2024. URL: https://openreview.net/pdf?id=FUZYp83y4M.

[190] Jesse Vig et al. "Investigating Gender Bias in Language Models Using Causal Mediation Analysis". In: *Neural Information Processing Systems*. 2020. URL: https://api.semanticscholar.org/CorpusID:227275068.

[191] Elena Voita, Rico Sennrich, and Ivan Titov. "The Bottom-up Evolution of Representations in the Transformer: A Study with Machine Translation and Language Modeling Objectives". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 4396–4406. DOI: 10.18653/v1/D19-1448. URL: https://aclanthology.org/D19-1448.

[192] Catherine Wah et al. *The Caltech-UCSD Birds-200-2011 Dataset*. Version Published. Aug. 2011.

[193] Kevin Ro Wang et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small". In: *The Eleventh International Conference on Learning Representations*. 2023. URL: https://openreview.net/forum?id=NpsVSN6o4ul.

[194] Qianru Wang et al. "Epistasis regulates genetic control of cardiac hypertrophy". In: *MedRxiv preprint https://www.medrxiv.org/content/10.1101/2023.11.06.23297858v1* (2023).

[195] Y. Wang, X. Ma, and W. Chen. "Augmenting Black-Box LLMs with Medical Textbooks for Clinical Question Answering". In: *arXiv preprint arXiv:2309.02233* (2023).

[196] Yanshan Wang, Xiaoxi Yao, and Xizhi Wu. "Transforming Large Language Models into Superior Clinical Decision Support Tools by Embedding Clinical Practice Guidelines". In: *Mayo Clinic Proceedings: Digital Health* 2.3 (2024), pp. 491–492. DOI: 10.1016/j.mcpdig.2024.05.018.

[197] Eric Wong, Shibani Santurkar, and Aleksander Madry. *Leveraging Sparse Linear Layers for Debuggable Deep Networks*. 2021. arXiv: 2105.04857 [cs.LG]. URL: https://arxiv.org/abs/2105.04857.

[198] Yonghui Wu et al. "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation". In: *CoRR* abs/1609.08144 (2016). arXiv: 1609.08144. URL: http://arxiv.org/abs/1609.08144.

[199] Zhengxuan Wu et al. "Causal Distillation for Language Models". In: *ArXiv* (2021).

[200] Xinyue Xu et al. *Energy-Based Concept Bottleneck Models: Unifying Prediction, Concept Intervention, and Probabilistic Interpretations*. 2024. arXiv: 2401.14142 [cs.CV]. URL: https://arxiv.org/abs/2401.14142.

[201] Alexander Yalunin, Dmitriy Umerenkov, and Vladimir Kokh. "Abstractive summarization of hospitalisation histories with transformer networks". In: *CoRR* (2022).

[202] Yunzhi Yao et al. "Editing large language models: Problems, methods, and opportunities". In: *arXiv preprint arXiv:2305.13172* (2023).

[203] Bin Yu and Karl Kumbier. "Veridical data science". In: *PNAS* 117(8) (2020), pp. 3920–3929.

[204] Mert Yuksekgonul, Maggie Wang, and James Zou. *Post-hoc Concept Bottleneck Models*. 2023. arXiv: 2205.15480 [cs.LG]. URL: https://arxiv.org/abs/2205.15480.

[205] Zeyu Yun et al. "Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors". In: *arXiv* (2021).

[206] Omar Zaidan and Jason Eisner. "Modeling annotators: A generative approach to learning from annotator rationales". In: *Proceedings of the 2008 conference on Empirical methods in natural language processing*. 2008, pp. 31–40.

[207] C. Zakka et al. "Almanac — Retrieval-Augmented Language Models for Clinical Medicine". In: *NEJM AI* 1 (2024), p. 25. DOI: 10.1056/AIoa2300004.

[208] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer. 2014, pp. 818–833.

[209] Rowan Zellers et al. "From recognition to cognition: Visual commonsense reasoning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 6720–6731.

[210] Kelly Zhang and Samuel Bowman. "Language Modeling Teaches You More than Translation Does: Lessons Learned Through Auxiliary Syntactic Task Analysis". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 359–361. DOI: 10.18653/v1/W18-5448. URL: https://aclanthology.org/W18-5448.

[211] Longxiang Zhang et al. "Leveraging Pretrained Models for Automatic Summarization of Doctor-Patient Conversations". In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 3693–3712. DOI: 10.18653/v1/2021.findings-emnlp.313. URL: https://aclanthology.org/2021.findings-emnlp.313.

[212] Peitian Zhang et al. "Retrieve Anything To Augment Large Language Models". In: *arXiv preprint arXiv:2310.07554* (2023).

[213] Tianyi Zhang et al. "Bertscore: Evaluating text generation with bert". In: *arXiv preprint arXiv:1904.09675* (2019).

[214] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. "Character-level Convolutional Networks for Text Classification". In: *NIPS*. 2015.

[215] Ruochen Zhao et al. "Explaining Language Models' Predictions with High-Impact Concepts". In: *arXiv preprint arXiv:2305.02160* (2023).

[216] Ruiqi Zhong et al. "Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections". In: *arXiv preprint arXiv:2104.04670* (2021).

[217] Ruiqi Zhong et al. "Describing Differences between Text Distributions with Natural Language". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 27099–27116.

[218] Ruiqi Zhong et al. "Goal Driven Discovery of Distributional Differences via Language Descriptions". In: *arXiv preprint arXiv:2302.14233* (2023).

[219] Yongchao Zhou et al. "Large Language Models Are Human-Level Prompt Engineers". In: *arXiv preprint arXiv:2211.01910* (2022).

[220] Yukun Zhu et al. "Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books". In: *The IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015.

[221] Zhiying Zhu, Weixin Liang, and James Zou. "GSCLIP: A Framework for Explaining Distribution Shifts in Natural Language". In: *arXiv preprint arXiv:2206.15007* (2022).

Appendix A

Appendices

A.1 FIGS-BD experiment details extended

Model architectures and hyperparameters

For all datasets and CBM/TBM models, we utilize code and the majority of architectures provided by the authors of respective papers. For more complex concept-to-target (CTT) portions of the CBM, we modified provided code and scripts to train and evaluate the model. For FIGS models, we have contributed to the `imodels` [165] package (specifically, the FIGS implementation) to restrict the maximum depth of trees, handle multi-output prediction tasks, and create cross validation (CV) models. We have then adapted that model for ATTIs.

CUB and Traveling Birds

The Caltech-UCSD Birds-200-2011 (CUB) dataset [192] and the TravelingBirds [84] dataset contain $n = 11,788$ photos of birds with 200 bird class labels. Every observation in the dataset comes with human-labelled annotations regarding concepts present in the image, which facilitates our ATTI experiments. We reduce the number of concepts used in the same procedure as described in Koh et al. [84]. We utilize the code, instructions, and some trained models provided by Koh et al. [84]. We modify parts of their Github repository to incorporate more complex concept-to-target models. Specifically, we include MLP with 1 hidden layer, MLP with 2 hidden layers, and a simple Transformer model (encoder-only). All MLPs have the same hidden size, set to be 250 for CUB and TravelingBirds. The Transformer model utilizes multi-headed attention [188] with 4 heads, a MLP with 1 hidden layer of hidden size 250, and then a linear classifier layer. For the input-to-concept portion of the CBM, we utilize the Inception V3 [173] model, and for the overall model, utilize the overall Joint training process with $\lambda = 0.01$. All hyperparameters regarding training are the same as in Koh et al. [84]. Due to the complicated 200 class prediction task posed by CUB and TravelingBirds, we utilize a FIGS CV model to determine the hyperparameters

that result in the strongest FIGS-BD model. We use an interpretable rule of > 0 (anpositive concept prediction results in 1, negative results in 0) to binarize concept features before FIGS distillation. We search over [125, 200] rules, [30, 40] trees, and [3, 4] max depth. For CV results in Table 2.1, the post cross-validation fitted FIGS-BD model results in 200 rules, 30 trees, and max depth of 3 for both CUB and TravelingBirds.

AGNews and CEBaB

AGNews contains $n = 7,600$ news articles and 4 class labels (world, sports, business, and sci/tech) for news topic classification. CEBaB contains $n = 1,713$ restaurant views and their corresponding ratings (1-5) from customers as labels, and we formulate it as a regression task. For both datasets, we randomly split them into $n = 1,500$ train set and $n = 250$ test set for training and evaluation. To be comparable to the original TBM [102] paper, we use GPT-4 `(GPT-4-0613)` [131] as the underlying LLM for concept generation and concept measurement in the input-to-concept portion of the TBMs. The original TBM code uses Scikit-learn [134] for training linear regression (regression task) and logistic regression (classification task) for the concept-to-target portions of the TBMs. We modify parts of their code to incorporate more complex concept-to-target models. Specifically we include MLP with 1 hidden layer, MLP with 2 hidden layers, and a simple Transformer model (encoder-only). All MLPs have the same size set to be 50 for both AGNews and CEBaB datasets. The Transformer model utilizes two blocks of multi-headed attention (4 heads) + MLP with 1 hidden layer (hidden size 52) module, and then a linear classifier layer. All hyperparameters regarding training are the same as in Ludan et al. [102], except for the refinement trial size, which we set to be 500 for training the more complicated CTT models (MLPs and the simple transformer). We use one-hot-encoding to binarize concept features before FIGS distillation. We search over [100, 200, 250] rules, [20, 30, 50] trees, and [3, 4] max depth. For the NLP results in Table 2.1, the post cross-validation fitted FIGS-BD model results in 154 rules, 50 trees, and max depth of 3 for both CEBaB and AGNews.

Full CBM and student model prediction and distillation results

Table A.1 and Table A.2 contain all teacher and student test prediction performances across a variety of teacher models and selection of student (regression) models: FIGS, XGBoost [29], Random Forest (RF) [22], and Decision Tree (DT) [21]. The teacher models vary in their concept-to-target portion, in which we consider Linear, MLP1, MLP2, and Transformer concept-to-target models. Note that FIGS-BD was trained using cross-validation, meaning that it is likely that if the teacher model is more complex, the FIGS-BD student model consists of more trees, more rules, and more depth. For CUB and TravelingBirds, we restricted XGBoost and RF to 30 trees (the same amount as the cross-validation-chosen FIGS-BD model). We depth-restricted XGBoost, RF, and DT to 3 (same as cross-validation chosen FIGS-BD model), 7 or 8, and 7 or 8, respectively, and chose the best performing model. We choose depth or 7 or 8 because there are 200 classes in the CUB and Traveling-

Bird tasks, so we need enough expressivity (and leaf nodes: $2^7 = 128, 2^8 = 256$) to achieve strong performance. For AGNews and CEBaB, we restricted XGBoost and RF to 50 trees, and depth-restricted XGBoost, RF, and DT to 3 (same as cross-validation chosen FIGS-BD model), 2 or 3, and 2 or 3, respectively, following the same logic. The results displayed consist of XGBoost, RF, and DT of depth 3, 8, and 8, respectively for CUB and TravelingBirds, and of depth 3 for all three models for AGNews and CEBaB.

On all datasets, XGBoost displays strong performance, but we note that XGBoost was not restricted in terms of number of rules (only restricted in depth and tree) and XGBoost also grows a separate estimator per class/task, for example, resulting in $30 \cdot 200 = 6000$ total trees (for CUB and TravelingBirds) with max depth 3. Thus, XGBoost is highly uninterpretable and grows highly inefficient and dense trees. On the other hand, FIGS-BD grows sparser and is a number-of-rules restricted model, consisting of only 30 trees of max depth 3 for CUB and TravelingBirds. RF and DT perform significantly worse than XGBoost and FIGS-BD, while RF is also highly uninterpretable.

Table A.1: Full CBM (teacher model) and student model test prediction performance across the image datasets. "Teacher Pred" and "Student Pred" denote teacher and student test prediction performance, respectively. Top prediction performance for each dataset and model role (teacher or student) in bold. The second-best student performance is underlined. RF and DT denote Random Forest and Decision Tree, respectively.

Dataset	**Teacher**	**Student**	**Teacher Pred**	**Student Pred**
CUB (Acc %)	CBM Linear	FIGS-BD	**79.8**	**75.9**
	-	XGBoost	-	**75.9**
	-	RF	-	64.4
	-	DT	-	50.2
	CBM MLP1	FIGS-BD	79.0	<u>73.7</u>
	-	XGBoost	-	**74.1**
	-	RF	-	65.3
	-	DT	-	51.8
	CBM MLP2	FIGS-BD	78.0	<u>72.7</u>
	-	XGBoost	-	**74.2**
	-	RF	-	65.4
	-	DT	-	48.0
	CBM Transformer	FIGS-BD	77.4	<u>72.2</u>
	-	XGBoost	-	**73.0**
	-	RF	-	66.2
	-	DT	-	51.5
TravelingBirds (Acc %)	CBM Linear	FIGS-BD	**51.8**	**47.9**
	-	XGBoost	-	<u>47.7</u>
	-	RF	-	38.4
	-	DT	-	28.5
	CBM MLP1	FIGS-BD	49.2	<u>48.5</u>
	-	XGBoost	-	**50.1**
	-	RF	-	41.5
	-	DT	-	31.5
	CBM MLP2	FIGS-BD	49.6	<u>49.1</u>
	-	XGBoost	-	**49.7**
	-	RF	-	42.0
	-	DT	-	33.7
	CBM Transformer	FIGS-BD	47.5	<u>47.1</u>
	-	XGBoost	-	**47.2**
	-	RF	-	43.4
	-	DT	-	32.2

Table A.2: Full CBM (teacher model) and student model test prediction performance across the text datasets. "Teacher Pred" and "Student Pred" denote teacher and student test prediction performance, respectively. Top prediction performance for each dataset and model role (teacher or student) in bold. The second-best student performance is underlined. RF and DT denote Random Forest and Decision Tree, respectively.

Dataset	**Teacher**	**Student**	**Teacher Pred**	**Student Pred**
AGNews (Acc %)	TBM Linear	FIGS-BD	84.8	<u>83.2</u>
	-	XGBoost	-	**86.8**
	-	RF	-	82.8
	-	DT	-	81.2
	TBM MLP1	FIGS-BD	84.4	<u>80.8</u>
	-	XGBoost	-	**83.6**
	-	RF	-	76.4
	-	DT	-	78.8
	TBM MLP2	FIGS-BD	80.8	<u>79.2</u>
	-	XGBoost	-	**82.0**
	-	RF	-	76.8
	-	DT	-	76.8
	TBM Transformer	FIGS-BD	**89.6**	**88.8**
	-	XGBoost	-	<u>88.0</u>
	-	RF	-	83.2
	-	DT	-	83.2
CEBaB (R-squared)	TBM Linear	FIGS-BD	0.761	<u>0.797</u>
	-	XGBoost	-	**0.804**
	-	RF	-	0.784
	-	DT	-	0.785
	TBM MLP1	FIGS-BD	0.837	<u>0.873</u>
	-	XGBoost	-	**0.882**
	-	RF	-	0.864
	-	DT	-	0.863
	TBM MLP2	FIGS-BD	0.808	**0.833**
	-	XGBoost	-	**0.833**
	-	RF	-	0.813
	-	DT	-	0.812
	TBM Transformer	FIGS-BD	**0.868**	<u>0.871</u>
	-	XGBoost	-	**0.877**
	-	RF	-	0.847
	-	DT	-	0.786

ATTI details extended

For CUB and TravelingBirds, as done by Koh et al. [84] for interventions, we replace the predicted concept values with the 5th quantile and 95th quantile of the predicted concept in the training data if the true concept is 0 and 1 for the original CBM (linear CTT), respectively. This is denoted as "map" in Figure 2.1. This can result in prediction performance degrading as replacing predicted values for a specific instance with training values, even if the pre-intervention and post-intervention concept values agree in some way (one could perhaps argue for equivalence in sign meaning an agreement, but there is no exact way without uncertainty to determine if a CBM predicted a concept correctly).

A.2 SASC methodology details extended

Table A.3: Statistics on corpuses used for explanation. Wikitext is used for BERT explanation and Moth stories are used for fMRI voxel explanation.

	Unique unigrams	Unique bigrams	Unique trigrams
Wikitext [112]	157k	3,719k	9,228k
Moth stories [89]	117k	79k	140k
Combined	158k	3,750k	9,334k

Prompts used in SASC The summarization step summarizes 30 randomly chosen ngrams from the top 50 and generates 5 candidate explanations using the prompt *Here is a list of phrases:\n{phrases}\nWhat is a common theme among these phrases?\nThe common theme among these phrases is ___.*

In the synthetic scoring step, we generate similar synthetic strings with the prompt *Generate 10 phrases that are similar to the concept of {explanation}:*. For dissimilar synthetic strings we use the prompt *Generate 10 phrases that are not similar to the concept of {explanation}:*. Minor automatic processing is applied to LLM outputs, e.g. parsing a bulleted list, converting to lowercase, and removing extra whitespaces.

Synthetic module results extended

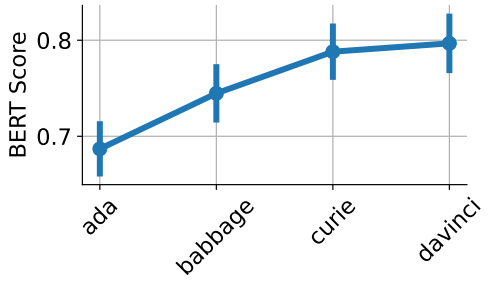


Figure A.1: The BERT score between generated explanation and groundtruth explanation generally increases as the size of the helper LLM for summarization/generation increases. Models are accessed via the OpenAI API (`text-ada-001`, `text-babbage-001`, `text-curie-001`, `text-davinci-001`, all accessed on Feb. 2023) and are in order of increasing size. BERT score for each module is computed as the maximum over the 5 generated explanations.

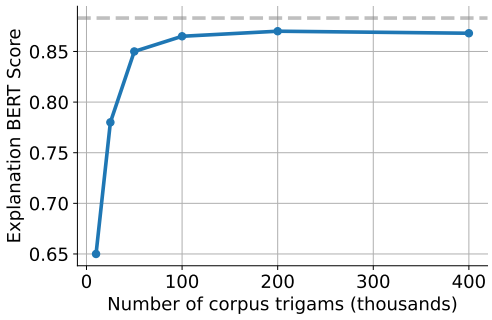


Figure A.2: Explanation BERT score for the 54 synthetic datasets as a function of corpus size. Performance plateaus around 100,000 ngrams. Corpus is created by randomly subsampling the unique trigrams in the WikiText dataset [112]. Gray dotted line shows the result when evaluating on dataset-specific corpuses, as in the *Default* setting in Table 3.1.

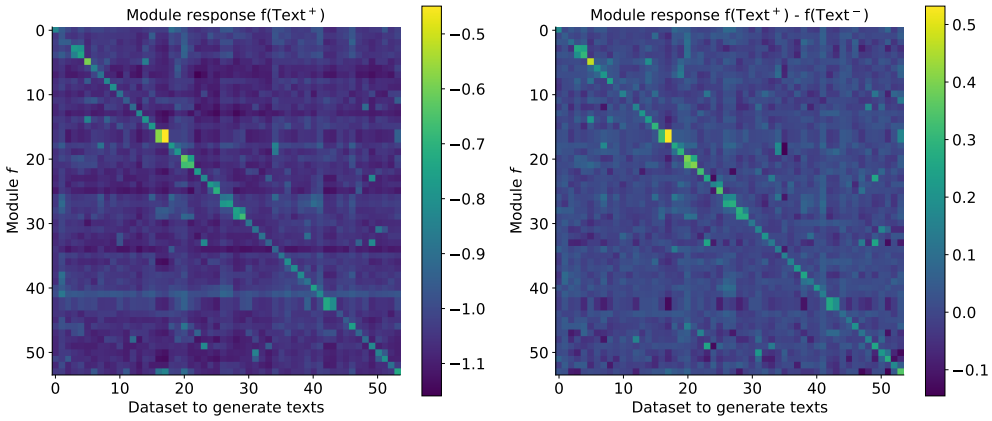


Figure A.3: Average module responses for synthetic texts that are related to the explanation (left, $f(\text{Text}^+)$) or the difference between the responses for related and unrelated texts (right, $f(\text{Text}^+) - f(\text{Text}^-)$). Responses correspond to synthetic modules in the *Default* setting. Bright diagonal on the left suggests that f selectively responses to synthetic texts generated according to the appropriate explanation. On the right, the diagonal is slightly less bright, suggesting that the module does not tend to respond more negatively to unrelated texts Text^-.

Synthetic module data details

Table A.4 showcases 54 synthetic modules and information about their underlying data corpus. Note that some modules use the same groundtruth Keyword (e.g. *environmentalism*), but that the underlying data corpus contains different data (e.g. text that is pro/anti environmentalism).

Table A.4

Module name	Groundtruth keyphrase	Dataset explanation	Examples	Unique unigrams
0-irony	sarcasm	contains irony	590	3897
1-objective	unbiased	is a more objective description of what happened	739	5628
2-subjective	subjective	contains subjective opinion	757	5769
3-god	religious	believes in god	164	1455
4-atheism	atheistic	is against religion	172	1472
5-evacuate	evacuation	involves a need for people to evacuate	2670	16505
6-terorrism	terrorism	describes a situation that involves terrorism	2640	16608
7-crime	crime	involves crime	2621	16333
8-shelter	shelter	describes a situation where people need shelter	2620	16347
9-food	hunger	is related to food security	2642	16276
10-infrastructure	infrastructure	is related to infrastructure	2664	16548
11-regime change	regime change	describes a regime change	2670	16382
12-medical	health	is related to a medical situation	2675	16223
13-water	water	involves a situation where people need clean water	2619	16135
14-search	rescue	involves a search/rescue situation	2628	16131
15-utility	utility	expresses need for utility, energy or sanitation	2640	16249
16-hillary	Hillary	is against Hillary	224	1693
17-hillary	Hillary	supports hillary	218	1675
18-offensive	derogatory	contains offensive content	652	6109
19-offensive	toxic	insult women or immigrants	2188	11839
20-pro-life	pro-life	is pro-life	213	1633
21-pro-choice	abortion	supports abortion	209	1593
22-physics	physics	is about physics	10360	93810
23-computer science	computers	is related to computer science	10441	93947
24-statistics	statistics	is about statistics	9286	86874
25-math	math	is about math research	8898	85118
26-grammar	ungrammatical	is ungrammatical	834	2217
27-grammar	grammatical	is grammatical	826	2236
28-sexis	sexist	is offensive to women	209	1641
29-sexis	feminism	supports feminism	215	1710
30-news	world	is about world news	5778	13023
31-sports	sports news	is about sports news	5674	12849
32-business	business	is related to business	5699	12913
33-tech	technology	is related to technology	5727	12927
34-bad	negative	contains a bad movie review	357	16889
35-good	good	thinks the movie is good	380	17497
36-quantity	quantity	asks for a quantity	1901	5144
37-location	location	asks about a location	1925	5236
38-person	person	asks about a person	1848	5014
39-entity	entity	asks about an entity	1896	5180
40-abbrevation	abbreviation	asks about an abbreviation	1839	5045
41-defin	definition	contains a definition	651	4508
42-environment	environmentalism	is against environmentalist	124	1117
43-environment	environmentalism	is environmentalist	119	1072
44-spam	spam	is a spam	360	2470
45-fact	facts	asks for factual information	704	11449
46-opinion	opinion	asks for an opinion	719	11709
47-math	science	is related to math and science	7514	53973
48-health	health	is related to health	7485	53986
49-computer	computers	related to computer or internet	7486	54256
50-sport	sports	is related to sports	7505	54718
51-entertainment	entertainment	is about entertainment	7461	53573
52-family	relationships	is about family and relationships	7438	54680
53-politic	politics	is related to politics or government	7410	53393

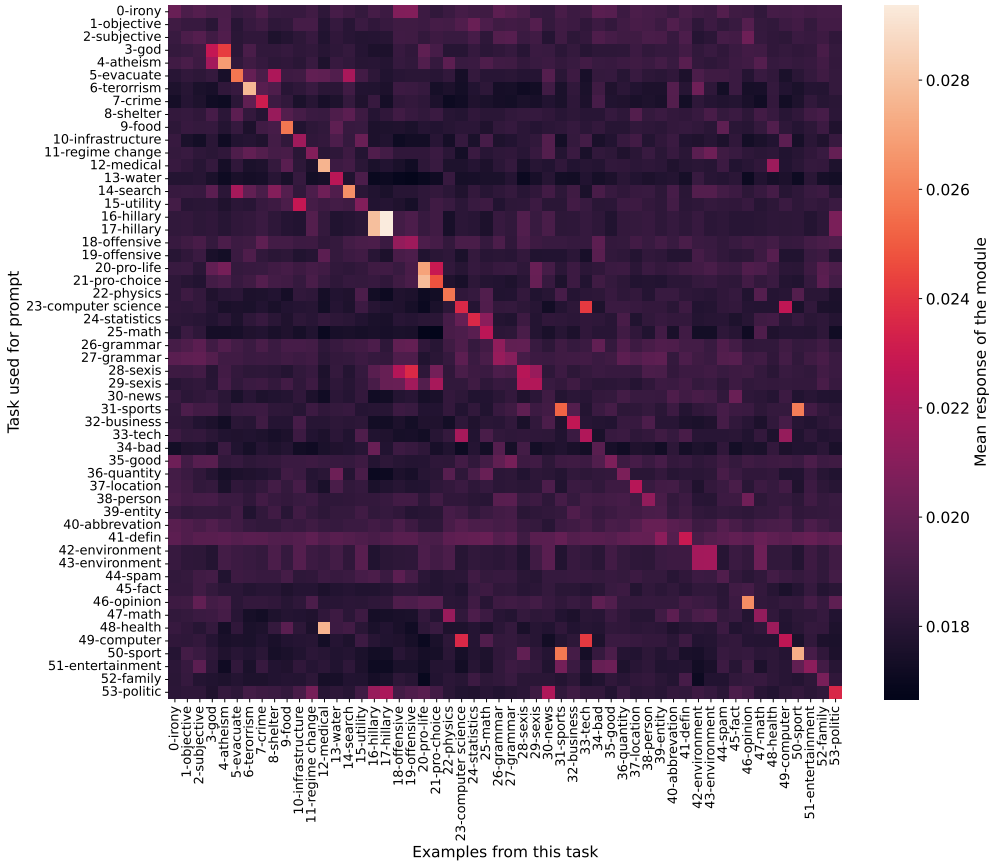


Figure A.4: Synthetic modules respond more strongly to phrases related to their keyphrase (diagonal) than to phrases related to the keyphrase of other datasets (off-diagonal). Each value shows the mean response of the module to 5 phrases and each row is normalized using softmax. Each module is constructed using Instructor [170] with the prompt *Represent the short phrase for clustering:* and the groundtruth keyphrase given in Table A.4. Related keyphrases are generated manually.

BERT interpretation

Details on fitting transformer factors Pre-trained transformer factors are taken from [205]. Each transformer factor is the result of running dictionary learning on a matrix X described as follows. Using a corpus of sentences S (here wikipedia), embeddings are extracted for each input, layer, and sequence index in BERT. The resulting matrix X has size $\left(\underbrace{\text{num\_layers}}_{\text{13 for BERT}} \cdot \sum_{s \in S} \text{len(s)}\right) \times \underbrace{d}_{\text{768 for BERT}}$. Dictionary learning is run on X with 1,500

dictionary components, resulting in a dictionary $D \in \mathbb{R}^{1,500 \times d}$. Here, we take the fitted dictionary released by [205] trained on the WikiText dataset [112].

During our interpretation pipeline, we require a module which maps text to a scalar coefficient. To interpret a transformer factor as a module, we specify a text input t and a layer l. This results in $len(t)$ embeddings with dimension d. We average over these embeddings, and then solve for the dictionary coefficients, to yield a set of coefficients $A \in \mathbb{R}^{1500}$. Finally, specifying a dictionary component index yields a single, scalar coefficient.

Extended BERT explanation results Tables A.6, A.8, and A.7 show explanations for modules selected by linear models finetuned on text-classification tasks.

Table A.5: Fraction of top logistic regression coefficients that are relevant for a downstream task (extends Table 3.5). Averaged over 3 random seeds; parentheses show standard error of the mean.

	Emotion	AG News	SST2
Top-10	0.50 ±0.08	1.00 ±0.00	0.80 ±0.14
Top-15	0.47 ±0.05	0.98 ±0.03	0.69 ±0.13
Top-20	0.42 ±0.09	0.98 ±0.02	0.55 ±0.10

Table A.6: SASC explanations for modules selected by 25-coefficient linear model on *SST2* for a single seed. Green shows explanations deemed to be relevant to the task.

Layer, Factor index	Explanation	Linear coefficient
(0, 783)	something being incorrect or wrong	-862.82
(0, 1064)	negative emotions and actions, such as hatred, violence, and disgust	-684.27
(1, 783)	something being incorrect, inaccurate, or wrong	-577.49
(1, 1064)	hatred and violence	-499.30
(0, 157)	air and sequencing	463.80
(9, 319)	a negative statement, usually in the form of not or nor	-446.58
(0, 481)	harm, injury, or damage	-441.98
(8, 319)	lack of something or the absence of something	-441.04
(10, 667)	two or more words	424.48
(2, 783)	something that is incorrect or inaccurate	-415.56
(0, 658)	thrice	-411.26
(0, 319)	none or its variations (no, not, never)	-388.14
(0, 1402)	dates	-377.74
(0, 1049)	standard	-365.83
(3, 1064)	negative emotions or feelings, such as hatred, anger, disgust, and brutality	-360.47
(4, 1064)	negative emotions or feelings, such as hatred, anger, and disgust	-357.35
(5, 152)	geography, history, and culture	-356.10
(0, 928)	homelessness and poverty	-355.05
(2, 691)	animals and plants, as many of the phrases refer to species of animals and plants	-351.62
(0, 810)	catching or catching something	350.98
(0, 1120)	production	-350.01
(0, 227)	a period of time	-345.72
(2, 583)	government, law, or politics in some way	-335.40
(2, 1064)	negative emotions such as hatred, disgust, and violence	-334.87
(4, 125)	science or mathematics, such as physics, astronomy, and geometry	-328.55

Table A.7: SASC explanations for modules selected by 25-coefficient linear model on *AG News* for a single seed. Green shows explanations deemed to be relevant to the task.

Layer, Factor index	Explanation	Linear coefficient
(5, 378)	professional sports teams	545.57
(4, 378)	professional sports teams in the united states	542.25
(3, 378)	professional sports teams	515.37
(0, 378)	names of sports teams	508.73
(6, 378)	sports teams	499.62
(2, 378)	professional sports teams	499.57
(1, 378)	professional sports teams	492.01
(7, 378)	sports teams	468.66
(8, 378)	sports teams or sports in some way	468.39
(11, 32)	activity or process	461.46
(12, 1407)	such	450.70
(5, 730)	england and english sports teams	427.33
(12, 104)	people, places, and events from history	425.49
(10, 378)	locations	424.71
(6, 730)	sports, particularly soccer	424.24
(12, 730)	sports	415.21
(4, 396)	people, places, or things related to japan	-415.13
(10, 659)	sports	410.89
(4, 188)	history in some way	404.24
(12, 1465)	different aspects of life, such as activities, people, places, and objects	403.77
(0, 310)	end with the word until	-400.10
(5, 151)	a particular season, either of a year, a sport, or a television show	396.41
(12, 573)	many of them contain unknown words or names, indicated by <unk	-393.27
(12, 372)	specific things, such as places, organizations, or activities	-392.57
(6, 188)	geography	388.69

Table A.8: SASC explanations for modules selected by 25-coefficient linear model on *Emotion* for a single seed. Green shows explanations deemed to be relevant to the task.

Layer, Factor index	Explanation	Linear coefficient
(0, 1418)	types of road interchanges	581.97
(0, 920)	fame	577.20
(6, 481)	injury or impairment	566.44
(5, 481)	injury or impairment	556.58
(0, 693)	end in oss or osses	556.53
(12, 1137)	ownership or possession	-537.45
(0, 663)	civil	524.88
(6, 1064)	negative emotions such as hatred, disgust, disdain, rage, and horror	523.41
(3, 872)	location of a campus or facility	-518.85
(5, 1064)	negative emotions and feelings, such as hatred, disgust, disdain, and viciousness	489.25
(0, 144)	lectures	482.85
(0, 876)	host	479.18
(0, 69)	history	-467.80
(0, 600)	many of them contain the word seymour or a variation of it	464.64
(0, 813)	or phrases related to either measurement (e.g	-455.11
(1, 89)	caution and being careful	451.73
(11, 229)	russia and russian culture	-450.28
(0, 783)	something being incorrect or wrong	448.55
(12, 195)	dates	442.14
(12, 1445)	breaking or being broken	439.81
(0, 415)	ashore	-438.22
(0, 118)	end with a quotation mark	437.66
(1, 650)	mathematical symbols such as >, =, and)	-437.28
(4, 388)	end with the sound ch	-437.15
(0, 840)	withdrawing	-436.38

fMRI module interpretation

fMRI data and model fitting

This section gives more details on the fMRI experiment analyzed in Section 3.6. These MRI data are available publicly [89, 177], but the methods are summarized here. Functional magnetic resonance imaging (fMRI) data were collected from 3 human subjects as they listened to English language podcast stories over Sensimetrics S14 headphones. Subjects were not asked to make any responses, but simply to listen attentively to the stories. For encoding model training, each subject listened to at approximately 20 hours of unique stories across 20 scanning sessions, yielding a total of ∼33,000 datapoints for each voxel across the whole brain. For model testing, the subjects listened to two test stories 5 times each, and one test story 10 times, at a rate of 1 test story per session. These test responses were averaged across repetitions. Functional signal-to-noise ratios in each voxel were computed using the mean-explainable variance method from [123] on the repeated test data. Only voxels within 8 mm of the mid-cortical surface were analyzed, yielding roughly 90,000 voxels per subject.

MRI data were collected on a 3T Siemens Skyra scanner at University of Texas at Austin using a 64-channel Siemens volume coil. Functional scans were collected using a gradient echo EPI sequence with repetition time (TR) = 2.00 s, echo time (TE) = 30.8 ms, flip angle = 71°, multi-band factor (simultaneous multi-slice) = 2, voxel size = 2.6mm x 2.6mm x 2.6mm (slice thickness = 2.6mm), matrix size = 84x84, and field of view = 220 mm. Anatomical data were collected using a T1-weighted multi-echo MP-RAGE sequence with voxel size = 1mm x 1mm x 1mm following the Freesurfer morphometry protocol [41].

All subjects were healthy and had normal hearing. The experimental protocol was approved by the Institutional Review Board at the University of Texas at Austin. Written informed consent was obtained from all subjects.

All functional data were motion corrected using the FMRIB Linear Image Registration Tool (FLIRT) from FSL 5.0. FLIRT was used to align all data to a template that was made from the average across the first functional run in the first story session for each subject. These automatic alignments were manually checked for accuracy.

Low frequency voxel response drift was identified using a 2nd order Savitzky-Golay filter with a 120 second window and then subtracted from the signal. To avoid onset artifacts and poor detrending performance near each end of the scan, responses were trimmed by removing 20 seconds (10 volumes) at the beginning and end of each scan, which removed the 10-second silent period and the first and last 10 seconds of each story. The mean response for each voxel was subtracted and the remaining response was scaled to have unit variance.

We used the fMRI data to generate a voxelwise brain encoding model for natural language using the intermediate hidden states from the the 18th layer of the 30-billion parameter LLaMA model [182], and the 9th layer of GPT [141]. In order to temporally align word times with TR times, Lanczos interpolation was applied with a window size of 3. The hemodyanmic response function was approximated with a finite impulse response model using 4 delays at -8,-6,-4 and -2 seconds [69]. For each subject x, voxel v, we fit a separate

encoding model $g_{(x,v)}$ to predict the BOLD response $\hat{B}$ from our embedded stimulus, i.e. $\hat{B}_{(x,v)} = g_{(x,v)}(H_i(\{S\}))$.

To evaluate the voxelwise encoding models, we used the learned $g_{(x,v)}$ to generate and evaluate predictions on a held-out test set. The GPT features achieved a mean correlation of 0.12 and LLaMA features achieved a mean correlation of 0.17. These performances are comparable with state-of-the-art published models on the same dataset that are able to achieved decoding [177].

To select voxels with diverse encoding, we applied principal components analysis to the learned weights, $g_{(x,v)}$, for GPT across all significantly predicted voxels in cortex. Prior work has shown that the first four principal components of language encoding models weights encode differences in semantic selectivity, differentiating between concepts like *social*, *temporal* and *visual* concepts. Consequently, to apply SASC to voxels with the most diverse selectivity, we found voxels that lie along the convex hull of the first four principal components and randomly sampled 1,500 of them (500 per subject). The mean voxel correlation for the 1,500 voxels we study is 0.35. Note that these voxels were selected for being well-predicted rather than easy to explain: the correlation between the prediction error and the explanation score for these voxels is 0.01, very close to zero.

Evaluating top fMRI voxel evaluations

Table A.9 shows two evaluations of the fMRI voxel explanations. First, similar to Figure 3.4, we find the mean explanation score remains significantly above zero. Second, we evaluate beyond whether the explanation describes the fitted module and ask whether the explanation describes the underlying fMRI voxel. Specifically, we predict the fMRI voxel response to text using only the voxel's explanation using a very simple procedure. We first compute the (scalar) negative embedding distance between the explanation text and the input text using Instructor [170][1]. We then calculate the spearman rank correlation between this scalar distance and the recorded voxel response (see Table A.9). The mean computed correlation is low[2], which is to be expected as the explanation is a concise string and may match extremely few ngrams in the text of the test data (which consists of only 3 narrative stories). Nevertheless, the correlation is significantly above zero (more than 15 times the standard error of the mean), suggesting that these explanations have some grounding in the underlying brain voxels.

[1]The input text for an fMRI response at time t (in seconds) is taken to be the words presented between $t-8$ and $t-2$.

[2]For reference, test correlations published in fMRI voxel prediction from language are often in the range of 0.01-0.1 [26].

Table A.9: Evaluation of fMRI voxel explanations. For all metrics, SASC is successful if the value is significantly greater than 0. Errors show standard error of the mean.

Explanation score	Test rank correlation
$1.27\sigma_f$ ± 0.029	0.033 ± 0.002

fMRI results when using WikiText Corpus

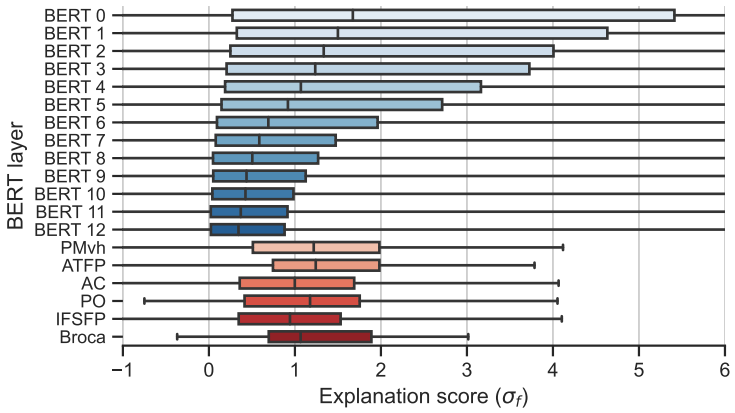


Figure A.5: Results in Figure 3.4 when using WikiText as the underlying corpus for ngrams rather than narrative stories.

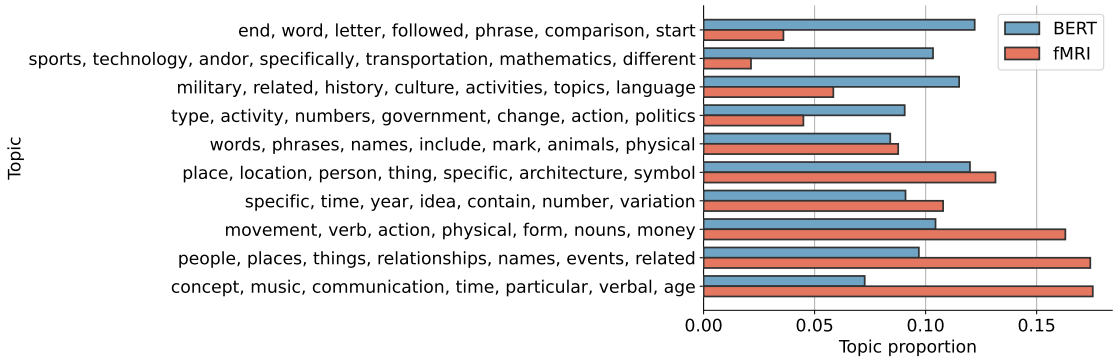


Figure A.6: Results in Figure 3.5 when using WikiText as the underlying corpus for ngrams rather than narrative stories.

A.3 CD-T circuit discovery details extended

Algorithm details

Heuristics

In addition to the greedy pruning presented in Algorithm 2 as a refinement step, here we describe other heuristic elements to the algorithm.

- The threshold for determining which set S of highest-contributing nodes may be varied: it is possible to pick the top N nodes or fraction of nodes, or to automatically detect

outliers. Empirically, we find that the distribution of node contributions varies quite severely, so finding a heuristic which works in all cases is actually an object of future work. In this paper, we set the threshold by varying the percentile of top nodes to extract, in the range of [90, 99] to obtain the ROC AUC in section 4.4.

- Empirically we find, for a fixed target node, the relevance scores of source nodes in different layers to this target node may have different expected magnitudes, due to the numerical effects of propagation through the network. To account for this, we normalize the relevance scores by dividing by the average magnitude across scores found in a given layer.
- To avoid the risk of propagation equations becoming numerically unstable after a large number of iterations, especially if the relevant and irrelevant constituents differ in sign at a specific index, we set the value of one of rel/irrel at this position to 0 and the other to the sum of the two terms.

Complexity analysis

To provide more clarity, the computational complexity of the algorithm satisfy the following properties:

- Each decomposition (of a set of target nodes with respect to a set of source nodes) requires cost in FLOPs similar to one forward pass of the model (and often less, since values prior to the source nodes can be cached and values after the target nodes do not need to be calculated).
- The core of the algorithm is a loop, where in each iteration, we search over all nodes which can potentially have high relevance to the target nodes. (This means that heuristically excluding some nodes from the search can potentially significantly decrease cost in FLOPs, and cost in FLOPs increases linearly with respect to the granularity with which we separate the nodes in the model.)
- The memory footprint of a single decomposition, including the forward pass, is a small (less than 3) constant multiple of the cost of a forward pass; the only added costs are to keep track of the relevant and irrelevant constituent tensors separately, as well as bookkeeping of components of the target decomposition metric.
- The cost (in FLOPs or memory) of performing the analysis on a set of input examples is linear in the number of input examples, since the same set of computations needs to be done with respect to each example.

Experiment details

Here we describe each task and how the experiments are performed in more detail; we also specify the exact model components which comprise the circuits we analyzed in the main

paper. In this section, we use the convention of specifying attention heads in a model by the tuple (layer index, head index). For the IOI task, we add a third entry for output of the head at a specific sequence position, and follow their nomenclature for semantically labeling the sequence positions.

Indirect Object Identification (IOI) [193]

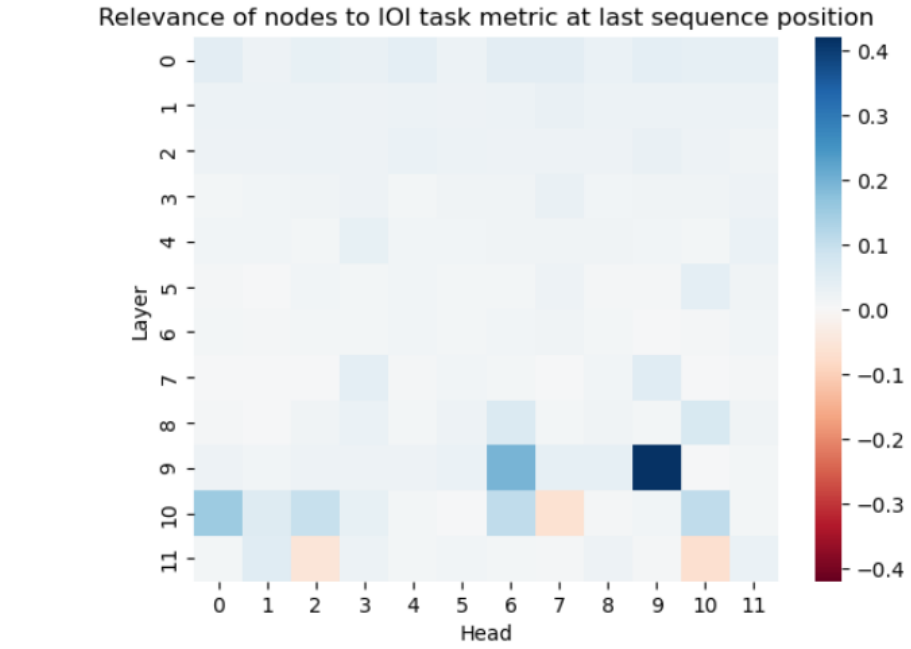
The indirect object identification (IOI) task is to predict the indirect object in a sentence with two entities, such as identifying "Mary" in the sequence "When Mary and John went for a walk, John gave an apple to ...". The objective originally defined for this task, which we also use, is the difference between the predicted logit of the indirect object (IO) token and the subject (S) token. Wang et al. [193] carefully design their experiment and dataset code so that mean ablation occurs using the mean activations over the corrupted "ABC dataset", which replaces the three name tokens in the task with random names. Likewise, when setting the decomposition of a node, we set the "relevant" component to the deviation from the mean activations on the ABC dataset. This task was performed on GPT2-small. GPT-2 correctly performs this task about 99 percent of the time, so we did not take special measures to account for those samples where it doesn't in our circuit analysis. We identify circuits using 25 IOI samples drawn from mixed templates, and mean ablation is conducted using the corrupted ABC dataset. Another set of 100 IOI samples are used in evaluation. Manual circuit of IOI compared against in the ROC AUC experiment is: [(2, 2), (4, 11), (0, 1), (3, 0), (0, 10), (5, 5), (6, 9), (5, 8), (5, 9), (7, 3), (7, 9), (8, 6), (8, 10), (10, 7), (11, 0), (9, 9), (9, 6), (10, 0), (9, 0), (9, 7), (10, 1), (10, 2), (10, 6), (10, 10), (11, 2), (11, 9)], which is from Figure 2 in Wang et al. [193].

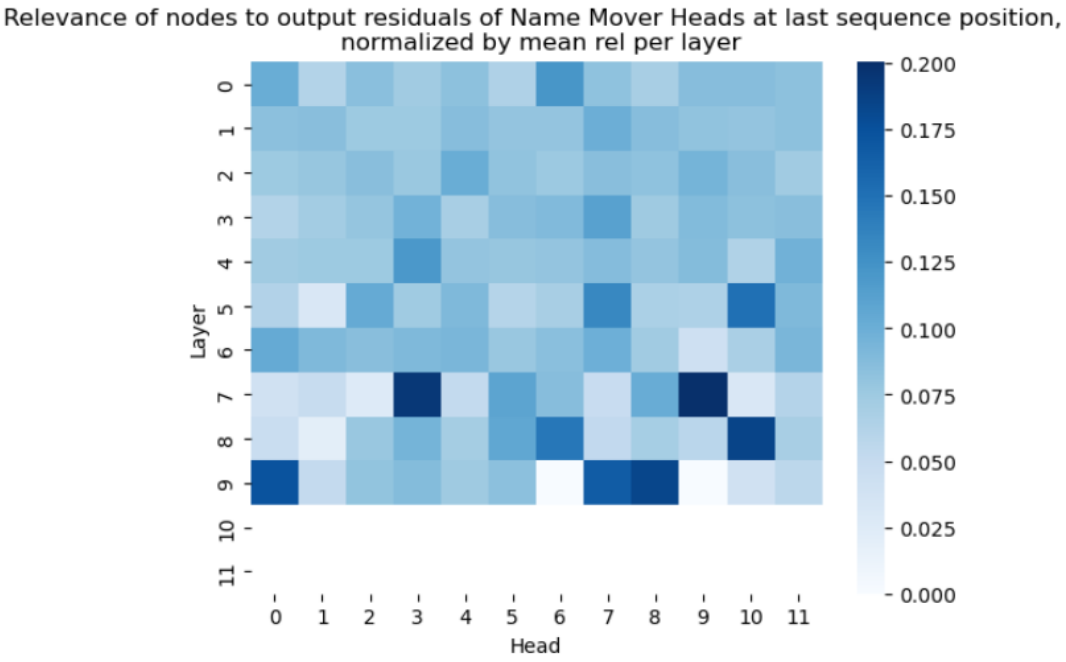
Details of Circuit Analysis For this task, in order to provide some intuitions about what CD-T calculates, we provide a number of heatmaps of the relevance scores at specific positions during various iterations of the circuit analysis. *In this section, we don't follow our automated circuit discovery algorithm exactly, but instead partially follow Wang et al. [193]'s analysis to decide what sequence positions to search over and visualize.*

The first iteration of the algorithm finds the Name Mover Heads: (9, 9, end), (10, 0, end), and (9, 6, end); the Negative Name Mover Heads: (10, 7, end), (11, 10, end); and some Backup Name Mover Heads: (10, 2, end), (10, 6, end), (10, 10, end), described by Wang et al. [193]

Following Wang et al. [193]'s analysis further, and deviating from the normal course of our circuit-finding algorithm, we compute the relevance of nodes to the Name Mover Heads on just the end position, and find what they named the "S-Inhibition Heads", at (8, 10, end), (7, 9, end), and (7, 3, end), though there are some other relevant-looking contenders:

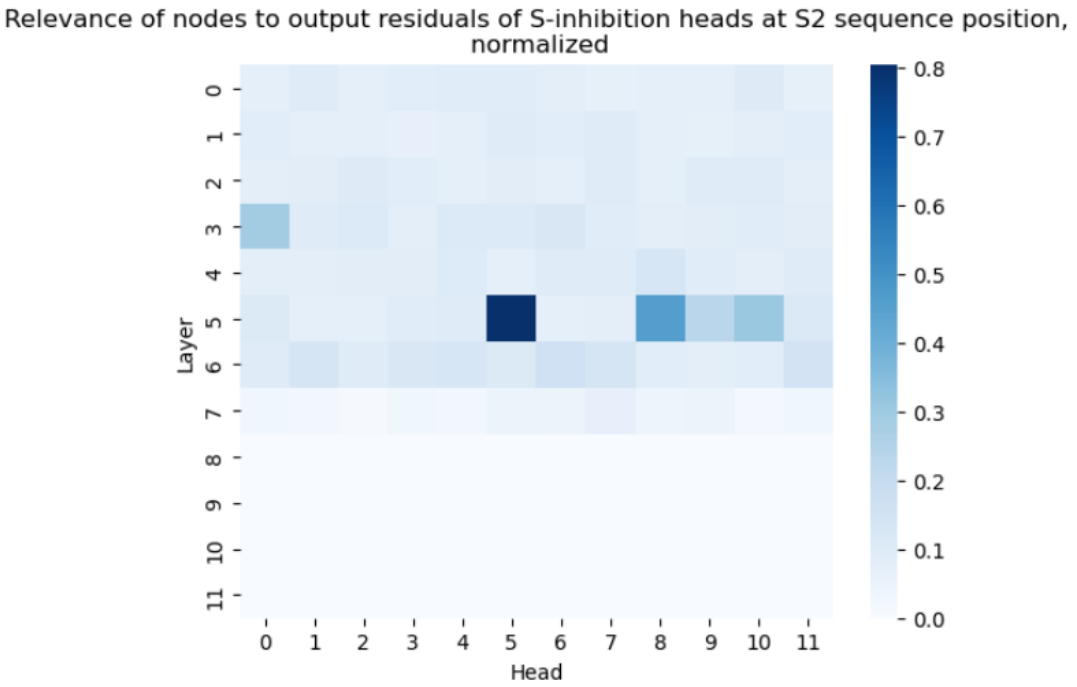
Continuing to follow their analysis, Wang et al. [193]'s analysis further, we compute the relevance of nodes to the S-Inhibition Heads at the S2 position, and find our first minor disagreement with their process: though we find two of what they named the Induction Heads at (5, 5, S2), (5, 8, S2), (5, 9, S2) we don't find the one at (6, 9, S2) but instead find

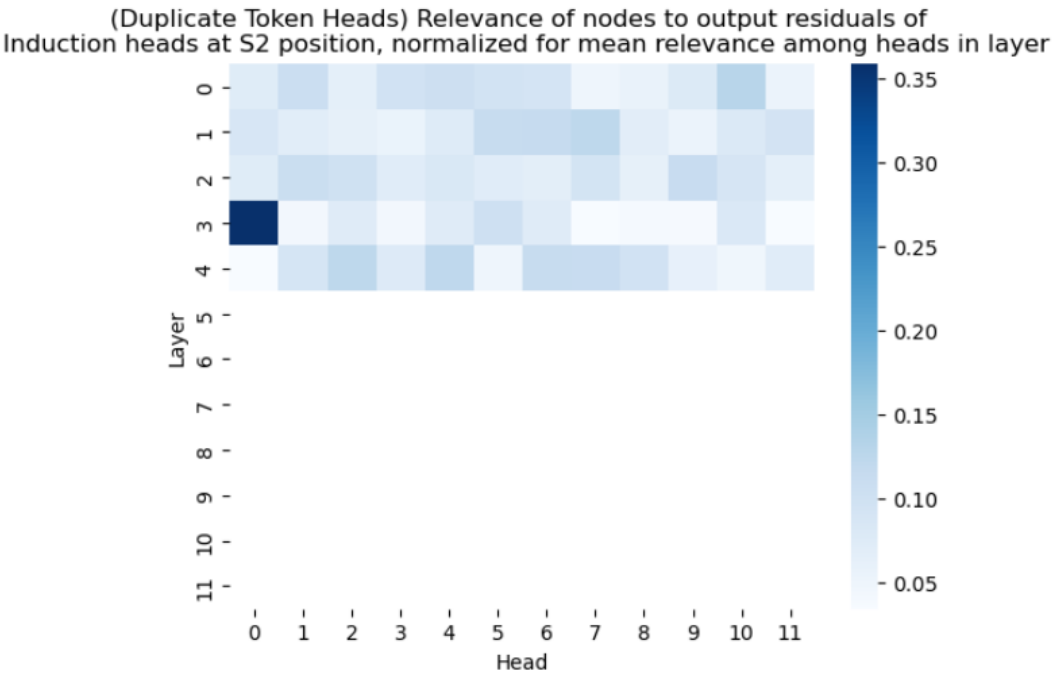




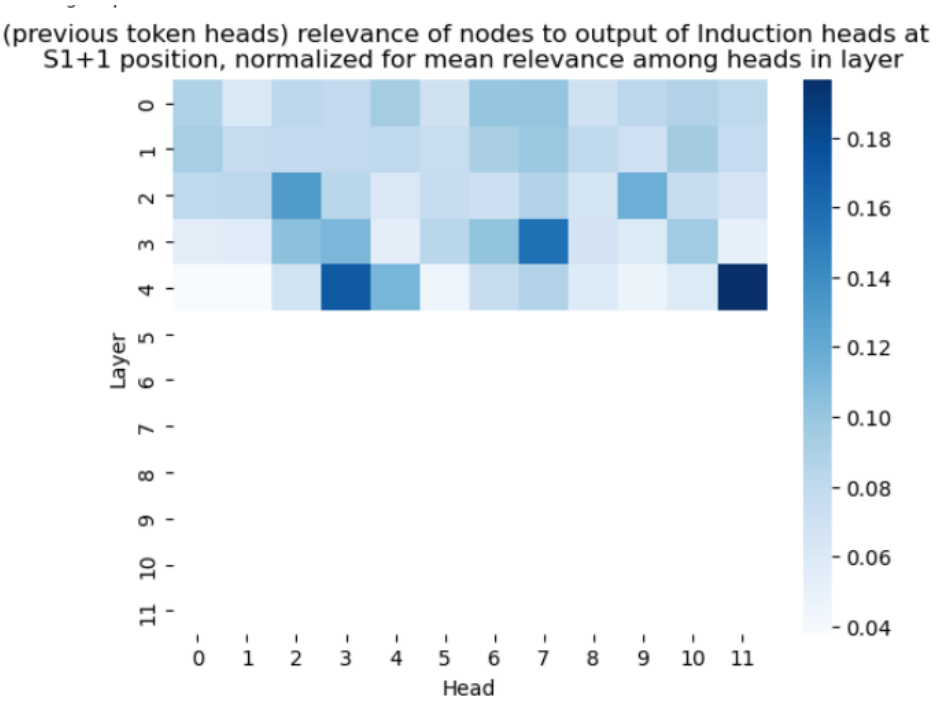
one at (5, 10, S2), and find a head their later analysis called a Duplicate Token Head, at (3, 0, S2).

Next we compute the relevance of nodes to the Induction Heads at the S2 position, and find that (3, 0, S2) mostly drowns out the signal of the other Duplicate Token Heads:





Finally, we compute the relevance of nodes to the Induction Heads at the S1+1 position, and find (4, 11, S1+1), with the other Previous Token Head they found at (2, 2, S1+1) a top contender, though there are other heads not accounted for in their analysis:



Overall, there is significant but not perfect agreement with the results and analysis of the IOI paper. We also attempted analysis by computing relevance to intermediate matrices (i.e., the key, query, value vectors) in the attention calculation, but found the plots to be qualitatively similar, possibly due to the fact that CD-T by design propagates relevances from these vectors to the attention head outputs as well.

Another fact of note is that the scales on these plots vary substantially. It would be desirable to find an interpretable normalization method to put these on the same scale with some intrinsic meaning, or otherwise explain the causes of this phenomenon.

Greater-Than [52]

The Greater than task is to predict the last two digits in an incomplete sentence following the template "The [noun] lasted from the year XXYY to the year XX". And we expect the model to assign higher probability to years greater than YY. The objective originally defined for this task, which we also use, is the sum of probabilities assigned to tokens corresponding to greater years, minus the sum of probabilities assigned to tokens corresponding to lesser years. (Some probability is assigned to tokens which don't correspond to numbers at all.) For our "mean-ablation", we simply take the mean over the activations over 100 negative datapoints (impossible completions, with the ending year preceding the starting century), and as above, when setting the decomposition at a source node, define the relevant component to be the deviation from the mean activation over this distribution. This task was performed on GPT2-small. GPT-2 correctly performs this task about 99 percent of the time, so we

did not take special measures to account for those samples where it doesn't in our circuit analysis. We identify circuits using a random sample of 100 datapoints provided by Hanna, Liu, and Variengien [52], and mean ablation is conducted using the negative impossible completion samples. Another set of 100 samples are used in evaluation.

It should be noted that our result and the results in Hanna, Liu, and Variengien [52] are not directly comparable, since they also attempt to investigate the influence of MLPs and their resulting circuit removes most of the MLPs in GPT-2. For the comparisons found in the main paper, we have compared our result (with all MLPs) to the circuit which is obtained by taking all the attention heads named in Hanna, Liu, and Variengien [52] (also with all MLPs), which is a circuit distinct from the one found in their work.

Manual circuit of Greater-than compared against in the ROC AUC experiment is: [(5, 1), (5, 5), (6, 1), (6, 9), (7, 10), (8, 8), (8, 11), (9, 1)].

Independently of the comparison, it remains true that keeping the relatively small proportion of attention heads in our circuit results in recovering almost all of GPT-2's capability on this task; see below.

	Task-specific metric	Correct guess rate
Full model	0.817	0.992
All attention heads ablated	-2.095	0
Their circuit	0.768	0.989
Their circuit (most MLPs ablated)	0.727	N/A
Our circuit	0.761	0.981

Docstring [58]

The goal of the Docstring task is to predict the next variable name in a Python docstring. For example, given a function with variable names LOAD, SIZE, FILES, and LAST, the task is to predict the word after one :PARAM. According to docstring conventions, this should be the variable name in the function definition which follows the most recent variable name to appear after :PARAM. The objective originally defined for this task is the logit assigned to the correct variable name, minus the max logit assigned to all other variable name tokens found in the function signature. For our "mean-ablation", we use their `random_random` dataset which randomize both the variable names in the function definition and in the docstring of prompts, and correspondingly, when setting the decomposition at a source node, define the relevant component to be the deviation from the mean activation over this distribution. We identify circuits using a 100 datapoints sampling for the dataset provided by Heimersheim and Janiak [58], and mean ablation is conducted using the corrupted `random_random` dataset. Another set of 100 samples are used in evaluation. This task was performed on a 4-layer attention-only transformer trained on natural language and Python code (`attn-only-4l`) released with the TransformerLens library for the express purpose of facilitating mechanistic interpretability research. Another complication is that the toy model only guesses the correct

token between 60 and 65 percent of the time. To account for this, we perform our circuit analysis and evaluation on the subset of input examples for which the model performs the task correctly.

Our circuit consists of the nodes (3, 0), (3, 6), (1, 4), (0, 5), (0, 0), (1, 0), (1, 4), (0, 1), (2, 3), (1, 2). Heimersheim and Janiak [58] find the circuit (0, 2), (0, 4), (0, 5), (1, 2), (1, 4), (2, 0), (3, 0), (3, 6) with their initial analysis, and heuristically observe that three heads help to obtain the "augmented circuit" (0, 2), (0, 4), (0, 5), (1, 2), (1, 4), (2, 0), (3, 0), (3, 6), (1, 0), (0, 1), (2, 3).

Manual circuit of Docstring compared against in the ROC AUC experiment is: [(0, 2), (0, 4), (0, 5), (1, 2), (1, 4), (2, 0), (3, 0), (3, 6)].

	Task-specific metric	Correct guess rate
Full model	3.661	1.0
All attention heads ablated	-2.095	0
Their circuit	2.884	0.54
Augmented circuit	3.524	0.66
Our circuit	3.235	0.57

A.4 SUFO experiment details extended

Description of extracted pathologic data elements

Table A.10: Description of the 4 extracted pathologic data elements.

Data elements	Description
Primary Gleason grade	A whole number from 1 to 5 representing the primary score given to a specimen based on the Gleason grading system to measure tumor aggressiveness.
Secondary Gleason grade	A whole number from 1 to 5 representing the secondary score given to a specimen based on the Gleason grading system to measure tumor aggressiveness.
Margin status for tumor	To evaluate surgical margins, the entire prostate surface is inked after removal. The surgical margins are designated as "negative" if the tumor is not present at the inked margin, and "positive" if tumor is present.
Seminal vesicle invasion	Invasion of tumor into the seminal vesicle. It is marked as "negative" if no invasion is present in the seminal vesicle, and "positive" if invasion is present.

Anonymized pathology report examples

- **Example 1:** "synoptic comment for prostate tumors " 1. type of tumor : adenocarcinoma small acinar type. " 2. location of tumor : both lobes. 3. estimated volume of tumor : 3. 5 ml. 4. gleason score : 4 + 3 = 7. 5. estimated volume ¿ gleason pattern 3 : 2 ml. 6. involvement of capsule : present (e. g. slide b6). 7. extraprostatic extension : not identified. 8. status of excision margins for tumor : negative. status of excision margins for benign prostate glands : positive (e. g. slide b4). 9. involvement of seminal vesicle : not identified. 10. perineural infiltration : present (e. g. slide b11). " 11. prostatic intraepithelial neoplasia (pin) : present high - grade (e. g " slide b4). 12. ajcc / uicc stage : pt2cnxmx ; stage ii if no metastases are identified. 13. additional comments : none. final diagnosis : " a. prostate left apical margin : benign prostatic tissue. " " b. prostate and seminal vesicles resection : prostatic adenocarcinoma " gleason score 4 + 3 = 7 ; see comment."

- **Example 2:** "synoptic comment for prostate tumors - type of tumor : small acinar adenocarcinoma. - location of tumor : - right anterior midgland : slides b3 - b5. - right posterior midgland : slides b6 - b8. - left anterior midgland : slides b12 - b14. - left posterior midgland : slides b9 - b11. - left and central bladder bases : slides b16 - b17 - estimated volume of tumor : 10 cm3. " - gleason score : 7 ; primary pattern 3 secondary pattern 4. " - estimated volume ¿ gleason pattern 3 : 40 %. " - involvement of capsule : tumor invades capsule but does not extend beyond " " capsule (slides b5 b8 b18). " - extraprostatic extension : none. - margin status for tumor : negative. - margin status for benign prostate glands : negative. - high - grade prostatic intraepithelial neoplasia (hgpin) : present ; extensive. - tumor involvement of seminal vesicle : none. - perineural infiltration : present. - lymph node status : none submitted. - ajcc / uicc stage : pt2cnx. final diagnosis : " a. prostate left base biopsy : fibromuscular tissue no tumor. " " b. prostate radical prostatectomy : " " 1. prostatic adenocarcinoma gleason grade 3 + 4 score = 7 involving " " bilateral prostate negative margins ; see comment. 2. " seminal vesicles with no significant pathologic abnormality."

Fine-tuning

We fine-tune the models to perform single-label classification for all tasks. We add a linear layer followed by a softmax function to the model output on the classification token. The datasets are divided into 71% training, 18% validation, and 11% test, with label distribution in each set resembling the distribution in the full datasets. Best model checkpoints are selected based on validation set performances, and are used in all experiments. For pathology reports, we evaluate the models against macro F1 as each class accounts for equal importance, while we report accuracy for MedNLI. We set the encoder sequence length to 512 tokens for pathology reports, and 256 tokens for MedNLI, which allows us to encode the full length of the majority of the datasets.

Prostate cancer pathology reports We use consistent fine-tuning hyperparameters for all models and all the four tasks, as we observe the validation set performance is not very sensitive to hyperparameter selection (less than 1% F1 performance change). We use an AdamW optimizer with a 7.6×10^{-6} learning rate, 0.01 weight decay, and a 1×10^{-8} epsilon. We also adopt a linear learning rate schedule with a 0.2 warm-up ratio. We fine-tune for a maximum of 25 epochs with a batch size of 8 and evaluate every 50 steps on the validation set. Each model is fine-tuned on a single NVIDIA Tesla K80 GPU, and average fine-tuning time is around 3 hours.

MedNLI We use consistent fine-tuning hyperparameters for all models, as we observe the validation set performance is not very sensitive to hyperparameter selection (less than 1% accuracy change). We use an AdamW optimizer with a per-layer learning rate decay schedule (1×10^{-4} as the starting learning rate, and 0.8 as the decay factor), 0 weight decay, 1×10^{-6} epsilon, and a 0.1 warm-up ratio. We fine-tune for a maximum of 10 epochs with a batch size of 32 and evaluate every epoch on the validation set. Each model is fine-tuned on a single NVIDIA GeForce GTX TITAN X GPU, and the fine-tuning time on average is less than 1 hours.

Per-class accuracy on Path-PG and Path-SG

Table A.11: Per-class accuracy of the five models on Path-PG and Path-SG, averaged across three runs (all stds are $< 5\%$ so we omit it to save spaces). PubMedBERT performs poorly when classifying the minority class `5` in the highly imbalanced Path-PG dataset, while it obtains descent performance across all classes in the slightly more balanced Path-SG dataset.

	Path-PG			Path-SG		
Models \Labels	`3`	`4`	`5`	`3`	`4`	`5`
BERT	0.99	0.94	1.00	0.98	0.98	0.97
TNLR	0.97	0.87	1.00	0.99	0.99	0.99
BioBERT	0.99	0.97	0.94	0.99	0.99	0.99
Clinical BioBERT	0.99	0.98	1.00	0.99	0.99	0.98
PubMedBERT	0.99	0.92	**0.67**	0.98	0.99	0.97

Fine-tuning results on MedNLI

Table A.12: Per-class accuracy and overall accuracy of PubMedBERT and Clinical BioBERT on MedNLI across three runs, where three scenarios are evaluated: Balanced ('C':'E':'N'=34%:33%:33%), Imbalanced ('C':'E':'N'=39%:53%:8%), and Highly Imbalanced ('C':'E':'N'=67%:30%:3%).

	Balanced		Imbalanced		Highly Imbalanced	
Labels \Models	PubMedBERT	Clinical BioBERT	PubMedBERT	Clinical BioBERT	PubMedBERT	Clinical BioBERT
Contradiction ('C')	0.88 (0.03)	0.76 (0.03)	0.76 (0.03)	0.70 (0.02)	0.80 (0.01)	0.79 (0.03)
Entailment ('E')	0.75 (0.02)	0.71 (0.02)	0.71 (0.03)	0.70 (0.02)	0.34 (0.02)	0.62 (0.05)
Neutral ('N')	0.77 (0.05)	0.72 (0.01)	0.33 (0.16)	0.32 (0.01)	0.04 (0.03)	0.04(0.02)
Accuracy	0.83 (0.01)	0.73 (0.01)	0.70 (0.02)	0.71 (0.01)	0.71 (0.01)	0.76 (0.03)

Quantifying the closeness between pre-training data and target data

We use perplexity of pre-trained models on target tasks to define the closeness between pre-training data and target data. The lower the perplexity means the closer the two data distributions should be.

Table A.13: Perplexity of the five models on pathology reports.

	BERT	TNLR	BioBERT	Clinical BioBERT	PubMedBERT
Perplexity	1.111	1.115	1.113	1.110	1.103

Principle component experiment details

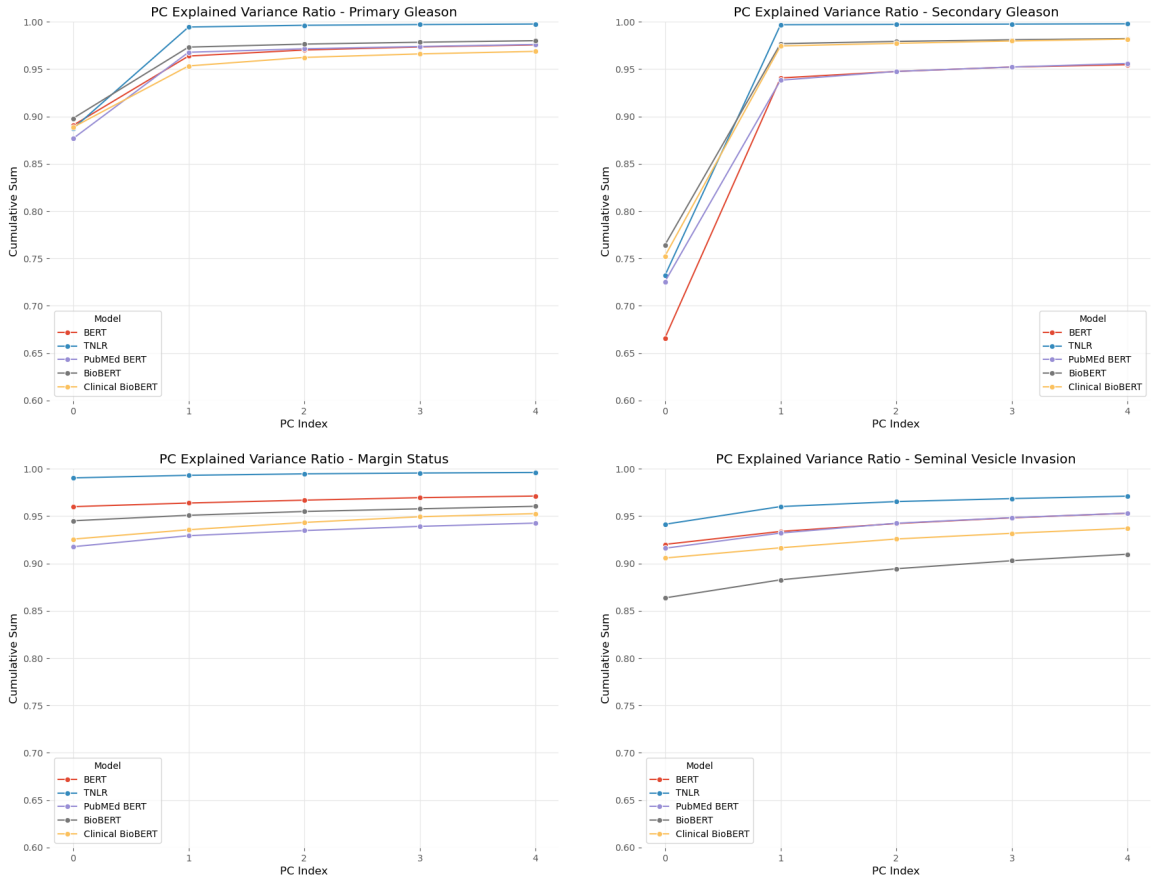


Figure A.7: The first two PCs in the fine-tuned last layer classification token feature spaces of all the models explain on average 95% of the dataset variance across the 4 tasks.

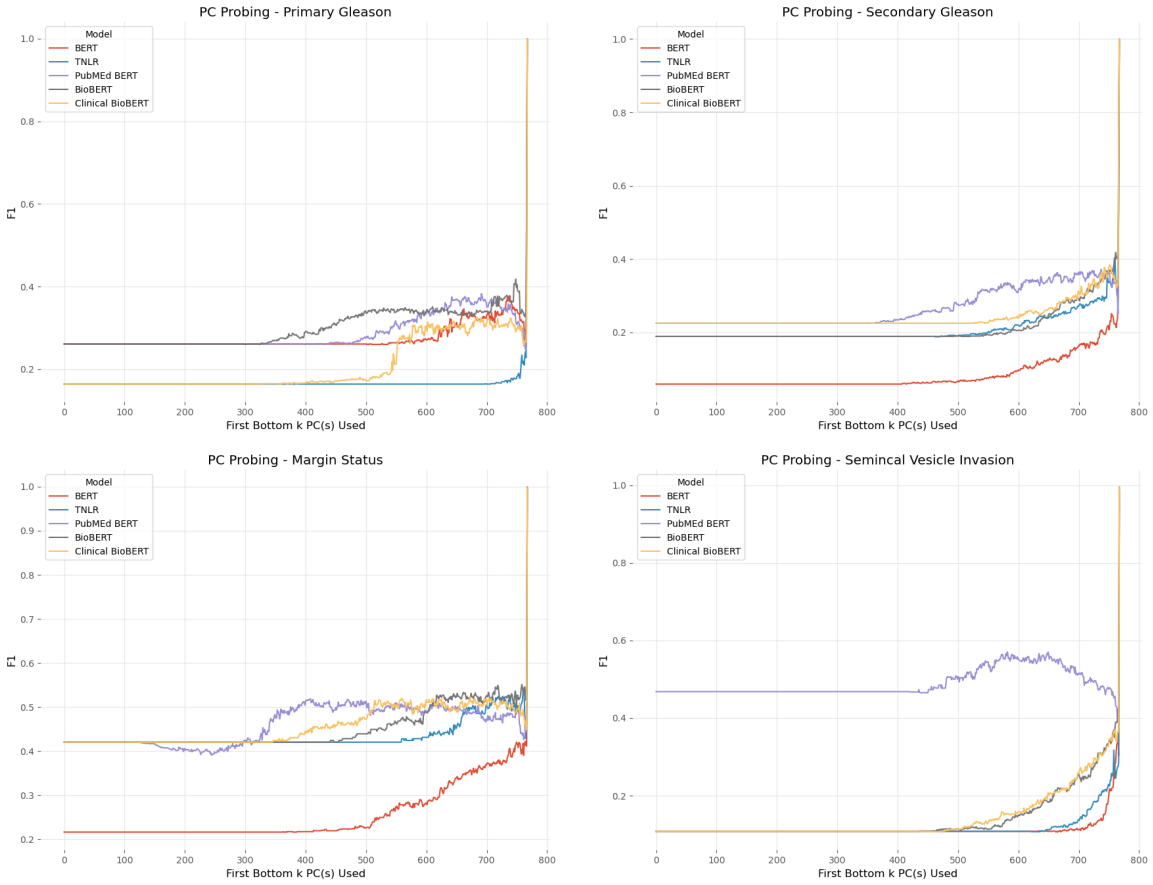


Figure A.8: Filling back in the first two PCs, at the last two steps, $k = 767$ and $k = 768$, yields significant model performance gain.

Outlier distributions

The full categorization of the types of outliers obtained from our expert evaluation is provided in Table A.14, while the distribution of these classes of outlier reports for each model are provided in Table A.15.

Table A.14: Description of categories of hard outliers

Outlier Category ID	Category Name	Category Description
1	Wrongly labeled report	These are reports for which the provided annotation is incorrect. For example, a report with `null` Gleason score corresponding to a scenario where a Gleason score cannot be assigned is wrongly included with another label.
2	Inconsistent report	For these reports, there exists inconsistent declarations of the target attribute (say, Primary Gleason score) in two different parts of the report.
3	Multiple Sources of Information	These reports contain multiple sources of information which are composed to produce one final label. One such instance of such an outlier (for the Secondary Gleason label) contained scores from five tumor nodules which were then combined to give one final composite score. A classifier must learn to distinguish the true final score from those that were used to obtain it.
4	Not reported or truncated report	These are reports for which the target attribute is either not reported or the report is truncated before entry into the database.
5	Boundary reports	These reports feature scenarios where the target attribute is hard to determine precisely or requires some interpretation of the provided information. For instance, one such report presents a Gleason score with a combined value of 7 with the other information in the report requiring the classifier to deduce that the Gleason score is $3 + 4$.

Table A.15: A distribution of Hard Outliers for each model categorized according to the 5 outlier types.

Outlier Type	BERT	BioBERT	Clinical BioBERT	PubMedBERT	TNLR
1	0	0	1	1	1
2	0	1	0	1	2
3	2	0	1	1	1
4	0	1	3	5	1
5	4	0	3	3	2
Total	6	2	8	11	7

Feature dynamics details

Here we present comprehensive sets of feature scatterplots along layers 1 to layer 12 (top-down) and selected epochs in the order of $1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25$ (left-right) of the 5 models, as we observe the models typically show the most rapid performance gain from epoch 1 to 10, and marginal increase afterwards. We include the plots from Path-PG, as we observe similar trend in the results of all the 4 tasks

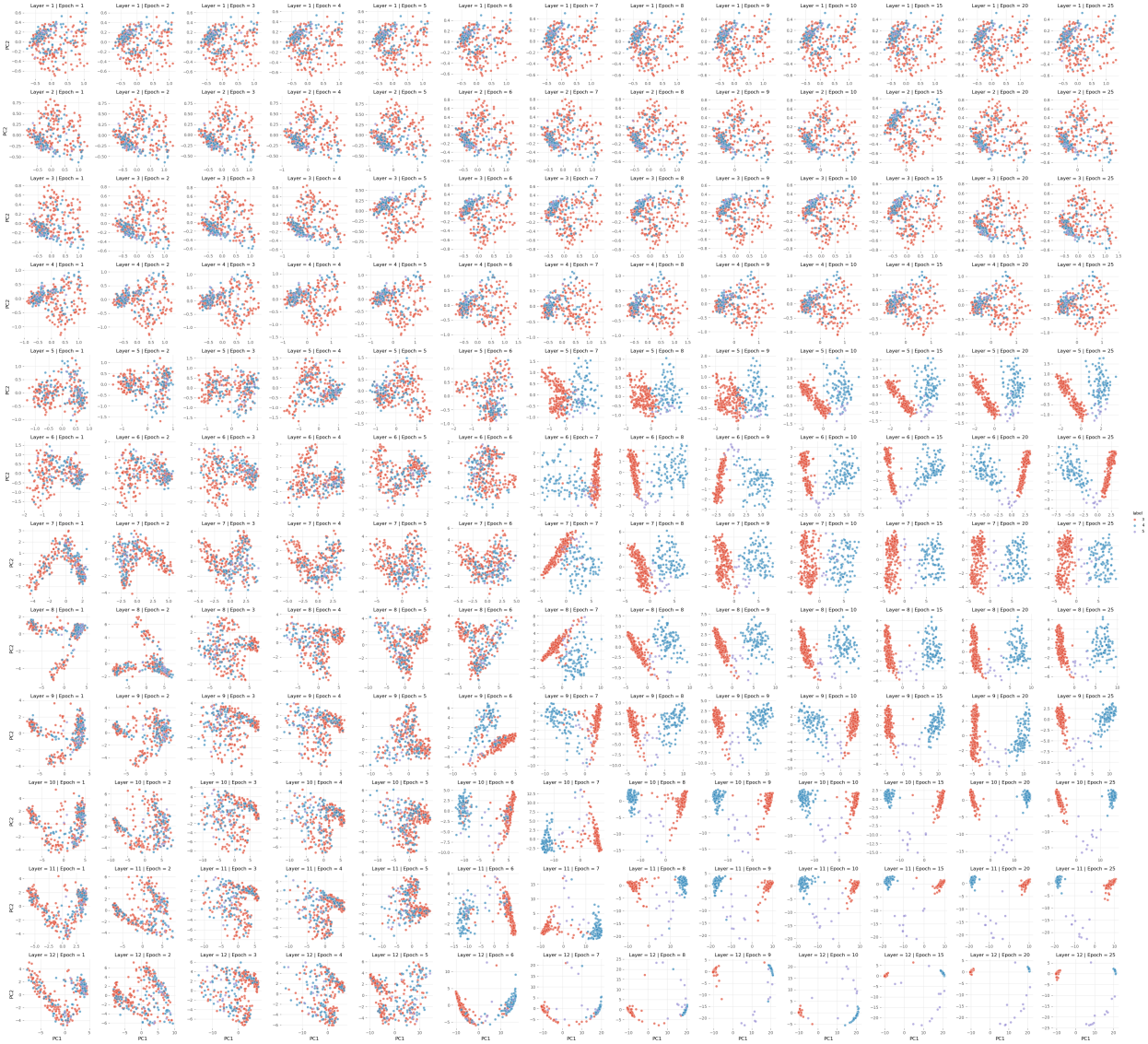


Figure A.9: Path-PG: BERT. Layer 1 to 12 from top to bottom. Epoch 1 to 25 (sampled) from left to right.

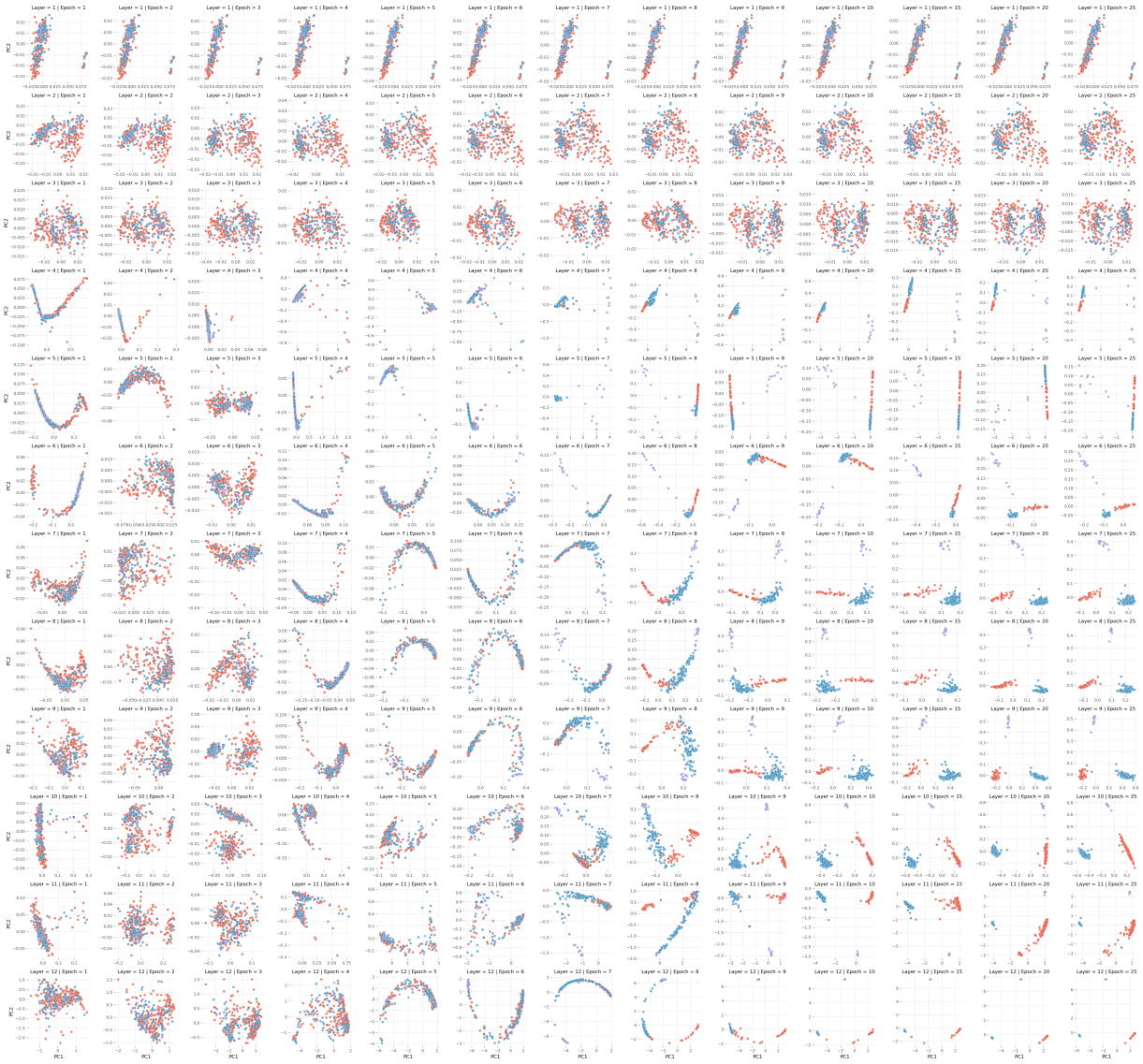


Figure A.10: Path-PG: TNLR. Layer 1 to 12 from top to bottom. Epoch 1 to 25 (sampled) from left to right.

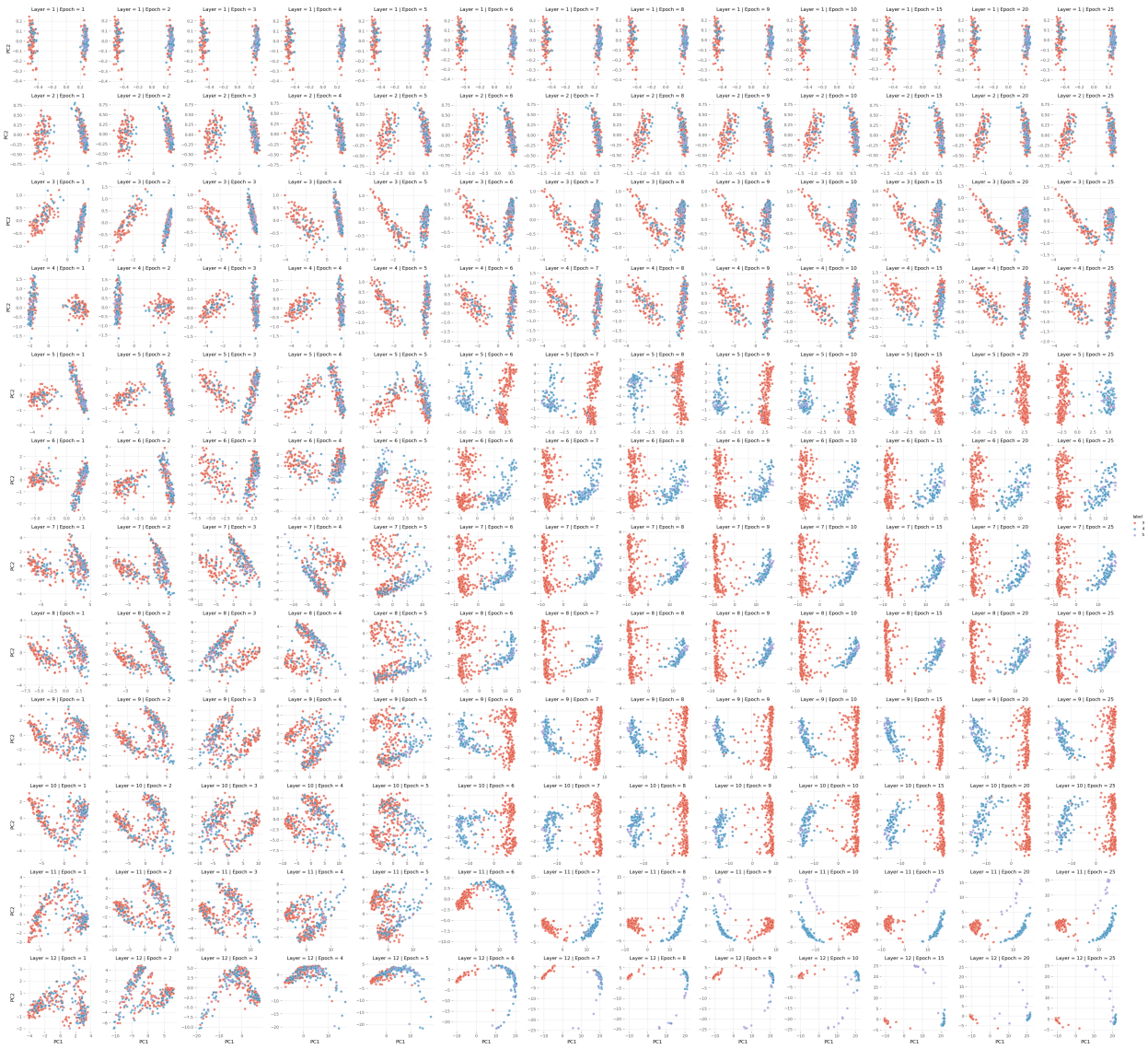


Figure A.11: Path-PG: BioBERT. Layer 1 to 12 from top to bottom. Epoch 1 to 25 (sampled) from left to right.

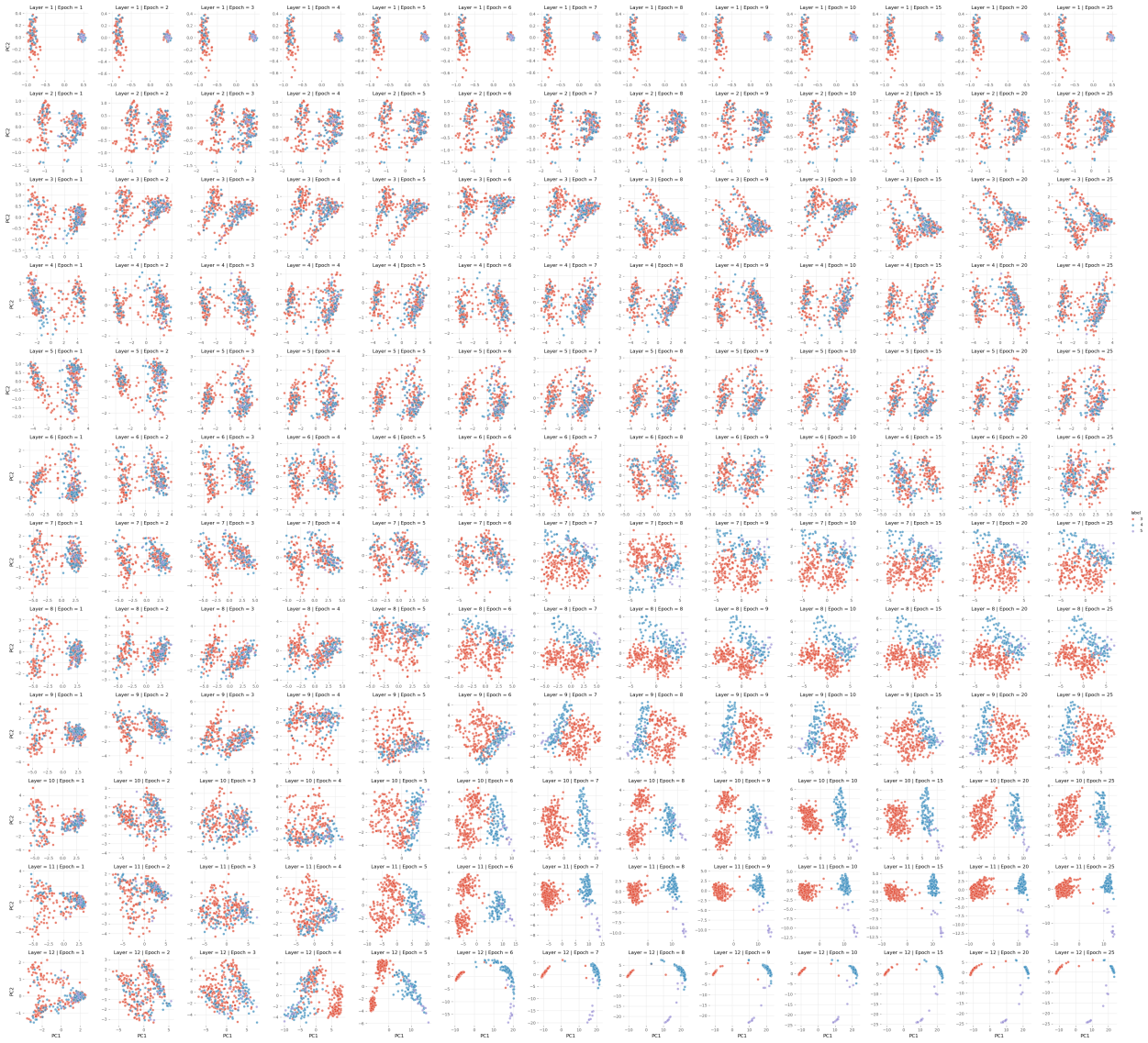


Figure A.12: Path-PG: Clinical BioBERT. Layer 1 to 12 from top to bottom. Epoch 1 to 25 (sampled) from left to right.

Figure A.13: Path-PG: PubMedBERT. Layer 1 to 12 from top to bottom. Epoch 1 to 25 (sampled) from left to right.