Understanding and Imposing Structure in Latent Spaces

by

Sotirios Panagiotis Chytas

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN–MADISON

2026

Date of final oral examination: 12/15/2025

The dissertation is approved by the following members of the Final Oral Committee:

Yong Jae Lee, Professor , Computer Sciences
Frederic Sala, Assistant Professor, Computer Sciences
Dimitris Papailiopoulos, Associate Professor, Electrical & Computer Engineering
Grigoris Chrysos, Assistant Professor, Electrical & Computer Engineering
Vikas Singh (Advisor), Professor, Biostatistics

$$1.01^{365} = 37.8$$
$$0.99^{365} = 0.03$$

ACKNOWLEDGMENTS

As you set out for Ithaka hope your road is a long one, full of adventure, full of discovery. Laistrygonians, Cyclops, angry Poseidon—don't be afraid of them: you'll never find things like that on your way as long as you keep your thoughts raised high, as long as a rare excitement stirs your spirit and your body. Laistrygonians, Cyclops, wild Poseidon—you won't encounter them unless you bring them along inside your soul, unless your soul sets them up in front of you.

Hope your road is a long one. May there be many summer mornings when, with what pleasure, what joy, you enter harbors you're seeing for the first time; may you stop at Phoenician trading stations to buy fine things, mother of pearl and coral, amber and ebony, sensual perfume of every kind— as many sensual perfumes as you can; and may you visit many Egyptian cities to learn and go on learning from their scholars.

Keep Ithaka always in your mind. Arriving there is what you're destined for. But don't hurry the journey at all. Better if it lasts for years, so you're old by the time you reach the island, wealthy with all you've gained on the way, not expecting Ithaka to make you rich.

Ithaka gave you the marvelous journey. Without her you wouldn't have set out. She has nothing left to give you now. And if you find her poor, Ithaka won't have fooled you. Wise as you will have become, so full of experience, you'll have understood by then what these Ithakas mean.

— C. P. CAVAFY

I am grateful to the committee for the time and effort they devoted to the preparation and evaluation of this work. I thank Frederic Sala, Yong Jae Lee, Dimitris Papailiopoulos, Grigoris Chrysos, and Vikas Singh for their insights, suggestions, thoughtful questions, and advice.

I would also like to thank all the people who were there for me, in one way or another, throughout these years. Old and new friends, labmates, fellow TAs, students, professors, relatives, and my parents have all contributed to this experience. I am grateful to each of them for the time we shared, their support, and the influence they had on me.

Finally, I would like to sincerely thank my advisor, Vikas Singh, for his guidance and impact. His help extended well beyond the strict boundaries of academia, encompassing broader lessons about thinking, planning, acting, and more. The PhD journey is, as the name suggests, a journey; an adventure that can make you wiser, more experienced, and ultimately change you in many ways. Throughout this process, Vikas was always there; a gentle, corrective force across multiple fronts.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

We are witnessing a profound transformation in how information is processed and how machines interact with humans. Since the breakthrough convolutional neural networks of 2011–2012, the field of Deep Learning has produced a vast array of architectures, each tailored to specific tasks and requirements. At their core, these models operate within high-dimensional latent vector spaces: abstract, opaque representations that have traditionally been treated as incidental artifacts of multilayer architectures. Yet this characterization may be misguiding and misleading.

This thesis focuses on understanding and leveraging the structure and capabilities of these latent spaces. We investigate the tools and methodologies that can illuminate their behavior and demonstrate how these spaces can be purposefully exploited across applications ranging from image harmonization to hallucination mitigation and synthetic data generation. We show how theoretical frameworks of applied Mathematics offer plausible, grounded explanations for the behavior of modern deep models; explanations that translate directly into practical algorithms enabling measurable improvements in performance.

This thesis's contributions demonstrate that latent spaces are not merely byproducts of deep architectures but fundamental computational domains. Computational domains that, under appropriate theoretical frameworks, can be understood, guided, and harnessed, opening new avenues for practical innovation in modern machine learning.

1 INTRODUCTION

In 1968, Stanley Kubrick envisioned an intelligent computational system capable of autonomous decision-making and of distinguishing between collective and individual goals [1]. In 1984, James Cameron depicted a dystopian future in which intelligent, self-sustaining systems turned against their creators and brought about humanity's downfall [2]. Earlier, in the 1960s, Isaac Asimov introduced the Three Laws of Robotics, outlining the ethical framework within which any intelligent artificial system should operate, and imagined a future where an all-seeing machine might even reverse the entropy of the universe [3; 4]. Decades later, in Iron Man [5], Tony Stark's assistant was not a human being but an artificial system capable of understanding natural language commands in real time and acting accordingly. Although the future remains uncertain, such ideas are no longer confined to the realm of science fiction. The field of Artificial Intelligence (AI), or, in its more technical manifestations, Machine Learning (ML) and Deep Learning (DL), has endowed humanity with unprecedented capabilities, fundamentally transforming the ways in which we interact, process information, and engage with technology.

Machine Learning is concerned with the study of diverse input signals and modalities, such as images, text, or attributes describing individuals and objects, and with the design of models capable of associating these inputs with specific quantities, such as the price of a house or the toxicity level of a comment. Broadly speaking, an ML model provides a structured mathematical mapping from input features to an output quantity. In the simplest case, that of linear regression [6], the target variable $y \in \mathbb{R}$ can be estimated as

$$y = w_1 x_1 + w_2 x_2 + \cdots + w_m x_m \tag{1.1}$$

where $\{w_i\}_{i=1}^m$ are (learnable) parameters, $\{x_i\}_{i=1}^m$ are the input features that correspond to measured quantities, such as the height and the weight of a person, the

number of occurrences of a specific word, the location of the depicted object, and more, while y corresponds to an appropriate target variable, such as whether an email is spam or not, whether a person is of high risk of developing diabetes, or the color and shape of the depicted object.

An exceptionally diverse collection of modeling approaches has been proposed, each designed to capture different aspects of the underlying structure of data [7]. From simple extensions of linear regression, formulated as

$$y = \phi(w_1 x_1 + w_2 x_2 + \cdots + w_m x_m), \tag{1.2}$$

where $\phi : \mathbb{R} \to \mathbb{R}$ is an appropriately chosen non-linear function, to non-parametric models such as k-nearest neighbors (kNN) [8], which compute weighted averages of the top-k elements in a held-out training set, and to decision trees, which represent cascades of conditional statements (Figure 1.1), the breadth of ML methods reflects the field's persistent effort to approximate complex aspects of reality through formal, algorithmic models.

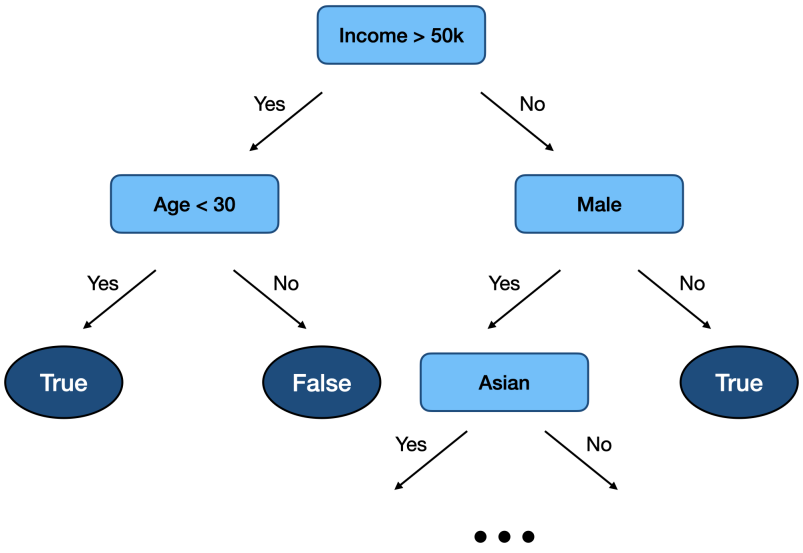


Figure 1.1: An example of a decision tree. Each internal node corresponds to a *True/False* question regarding one or more input features.

A fundamental characteristic of the above approach is its reliance on numerical features. Designing informative and discriminative features, as well as processing them appropriately, has historically been one of the most challenging, time-consuming, and non-trivial aspects of successful model development [9; 10; 11; 12]. Although certain standardized practices (e.g., feature scaling) have become well-established, the broader process of deriving meaningful representations and deciding which features to retain remains as much an art as it is a science. This challenge is particularly evident for input modalities that are not inherently numerical, such as images or natural language.

Recent advances, under the umbrella of Deep Learning, have largely mitigated these limitations, enabling the emergence of powerful models, such as ResNet and Transformers [13; 14]. The pivotal shift from classical Machine Learning to Deep Learning lies in the integration of feature extraction (also known as representation learning) into the model training process itself (Figure 1.2). Rather than relying on manually engineered feature extractors, Deep Learning models learn to derive task-relevant representations directly from data. For instance, convolutional neural networks (CNNs) [15] replaced handcrafted visual descriptors, multilayer perceptrons (MLPs) [16] supplanted manually designed kernels, and learnable token embeddings superseded traditional n-gram approaches [17; 18]. Combined with the computational advantages provided by modern GPUs, this paradigm shift facilitated the scaling of increasingly general and powerful models, reinforcing the now-common adage that "more data is better than better algorithms". Ultimately, this transition demonstrated that models can learn to map complex, high-dimensional inputs (e.g., images or text) onto well-behaved, lower-dimensional manifolds with minimal human intervention.

The concept of a manifold, along with its related notions of *latent spaces* and *latent representations*, has become central to contemporary ML/DL theory and practice. The idea that complex, high-dimensional observations can be faithfully represented within a lower-dimensional, structured space has inspired a vast array of models and algorithms (e.g., [19; 20; 21]). These latent spaces serve as the hid-

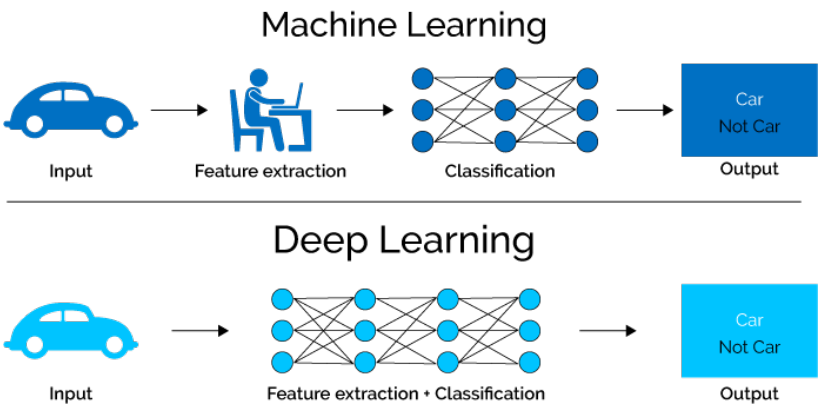


Figure 1.2: Machine Learning vs Deep Learning. Deep Learning incorporates in its training loop, and automates, the data preprocessing step, allowing for models that can handle complicated input modalities, such as images and text.

den spaces upon which models learn to organize, abstract, and reason about data. In particular, feature extraction through latent space mapping using pretrained models has become a standard methodology and, in many settings, a preferred alternative to fully end-to-end training. The emergence of Large Language Models (LLMs) has further strengthened this paradigm, with general-purpose text encoders [22] now forming the backbone of diverse text-based systems, ranging from semantic similarity tasks to Retrieval-Augmented Generation (RAG) frameworks [23; 24].

From a conceptual standpoint, latent spaces capture the long-standing aspiration of finding representations that disentangle the underlying factors of variation governing the observed world. Within this perspective, learning a latent representation is not merely a computational necessity but a philosophical one: it reflects the pursuit of models capable of internal abstraction, compression, and reasoning. The structure and geometry of these latent manifolds greatly influence a model's capacity to generalize, transfer, and generate coherent outputs.

Building upon these foundations, this thesis explores several aspects of latent representations, aiming to deepen our understanding of their formation, manipulation, and utility. In particular, it investigates how auxiliary or complementary

signals can be integrated during training to yield latent spaces that more accurately reflect underlying semantic or physical relationships. Moreover, it examines how mathematically motivated fusion mechanisms can be designed to combine embeddings from different sources in a principled manner, enhancing coherence and robustness without resorting to excessively large or potentially biased training regimes.

1.1 Manifold Learning

Manifold learning addresses the problem of discovering structure in high-dimensional data by assuming that the data are generated (or concentrated) near a smooth low-dimensional manifold embedded in a much higher-dimensional ambient space. This assumption, often called the *manifold hypothesis* [25], is central in many methods: it suggests that although each datum lives in $\mathbb{R}^D$, its intrinsic degrees of freedom d are far smaller ($d \ll D$), and that one may exploit this redundancy to obtain more efficient, geometrically meaningful representations. In empirical domains such as image and speech recognition, sensor data, and natural language embeddings, there is strong evidence that much of the observed variability is constrained by underlying latent parameters (pose, lighting, phoneme content, etc.) rather than generic high-dimensional noise (e.g., Figure 1.3) [26]. More formally, the mathematical framework supporting manifold learning posits that there is a smooth, compact d-dimensional Riemannian manifold $\boldsymbol{M} \subset \mathbb{R}^D$, perhaps with boundary, and that the observed data points $x_1, \ldots, x_n$ are sampled from a distribution supported on (or near) $\boldsymbol{M}$.

Among the landmark algorithms in manifold learning, Isomap [19] was among the first to target global geometry, approximating geodesic distances via shortest paths on a neighborhood graph and embedding the data using classical multi-dimensional scaling. Locally Linear Embedding [27] instead focuses on local linear structure, representing each point as a combination of its neighbors and preserving these reconstruction weights in the embedding. Laplacian Eigenmaps

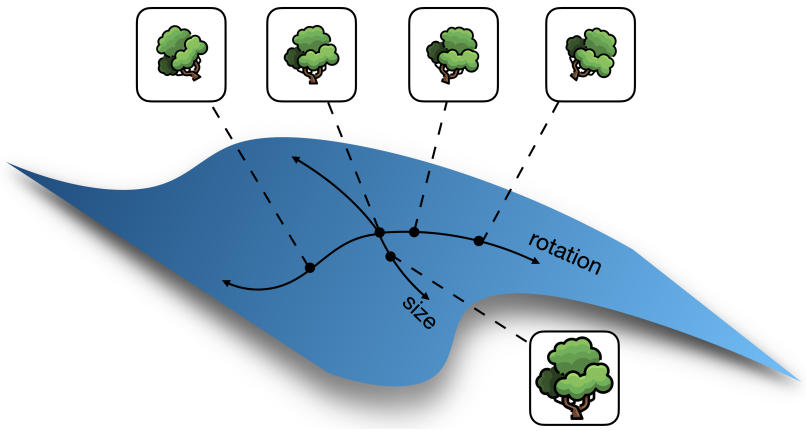


Figure 1.3: An example of a two-dimensional manifold, where the first dimension corresponds to rotation and the second to size of the displayed object in an image.

[20] and related spectral methods rely on the graph Laplacian to enforce locality and smoothness on the manifold. More recently, UMAP [21] constructs a fuzzy topological graph and optimizes a low-dimensional embedding that balances local neighborhood fidelity with aspects of global structure. Despite these successes, manifold learning faces practical and theoretical challenges, including the need for scalable approximations in large-scale settings [28; 29] and the difficulty of extending embeddings to unseen data without additional out-of-sample mechanisms [30].

1.2 Manifolds in Deep Learning

While, as already mentioned, a wide range of manifold learning algorithms have been proposed in the literature, an important development in recent years is the realization that deep learning models themselves can be understood as powerful implicit manifold learners. Classical methods such as Isomap [19], Locally Linear Embedding [27], and Laplacian Eigenmaps [20] were designed to explicitly discover a low-dimensional manifold that faithfully represents the geometry of high-dimensional data. These methods share the premise that real-world data, despite

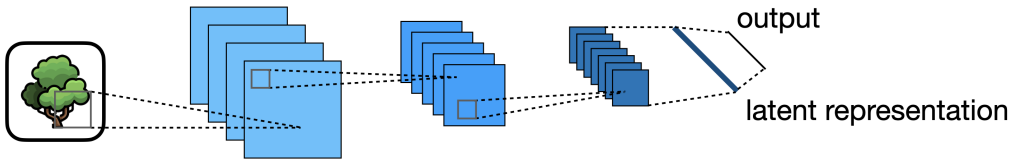


Figure 1.4: Manifold learning emerging as an intrinsic property of training Deep Neural Networks.

being embedded in a high-dimensional ambient space, typically lies on or near a manifold of much lower intrinsic dimensionality. However, such approaches often operate in an unsupervised manner, rely heavily on local neighborhood graphs, and lack scalability to modern, large-scale datasets.

Deep neural networks, by contrast, achieve a related goal implicitly. When trained for tasks such as classification, regression, reconstruction, or generative modeling, these networks simultaneously learn a representation of the input data that captures its essential structure. The optimization of supervised loss functions such as cross-entropy or mean squared error drives the model to organize intermediate features in a way that separates classes, preserves relevant invariances, and discards nuisance variations. In doing so, the network constructs latent spaces in which the geometry of the data is transformed into a representation more suitable for the downstream task (Figure 1.4). Remarkably, this process effectively uncovers the manifold structure of the data without the model being explicitly instructed to do so. Thus, manifold learning in deep networks emerges as a byproduct of the learning process itself [31; 32; 33].

Beyond providing empirical performance gains, the manifold learning viewpoint also establishes a deeper theoretical connection between classical geometric insights and modern practices. The learned latent spaces of deep models can be studied with tools from geometry and topology, offering a bridge between data analysis and representation learning. In fact, recent research has begun to formalize this connection by examining the curvature, dimensionality, and connectivity of latent manifolds induced by neural networks, linking them to notions of generalization, robustness, and interpretability [34]. From this perspective, DL

can be seen as not only solving specific predictive tasks but also performing an implicit form of unsupervised geometric organization of data.

This geometric organization allows us to conceptualize unconventional input modalities, such as images, audio signals, or words, as high-dimensional points embedded in a structured and smooth space. Such a viewpoint enables powerful algebraic operations on learned representations, like the ones presented in Chapter 3. For instance, [35] demonstrated that semantic relationships can be captured through vector arithmetic, with expressions such as

$$\mathbf{v}_{\text{queen}} = \mathbf{v}_{\text{king}} + (\mathbf{v}_{\text{woman}} - \mathbf{v}_{\text{man}}) \tag{1.3}$$

holding approximately in the induced latent space, where $\mathbf{v}_{\text{x}}$ denotes the latent representation of word x. This idea has also evolved into the notion of latent steering, where the addition or subtraction of concept vectors (e.g., $\mathbf{v}_{\text{toxicity}}$) can systematically modulate the behavior of large language models, enabling applications such as detoxification, sentiment control, and factual alignment [36; 37; 38; 39]; an idea explored in depth in Chapter 7.

Beyond simple linear operations such as addition or subtraction, a number of works have proposed richer forms of structure for latent spaces. A prominent example is Rotary Positional Embeddings (RoPE) [40], where latent vectors are rotated in embedding space according to their relative position within a sequence—providing a smoother and more expressive positional inductive bias than additive encodings. While such "advanced" compositions demonstrate that nontrivial transformations can offer practical benefits, designing composition rules that are expressive, stable, and computationally efficient remains far from straightforward, as analyzed in Chapter 4.

This challenge has motivated a growing line of research, inspired by Symbolic AI and Geometric Algebra [41; 42; 43], that seeks to endow latent spaces with structured, rigorous composition operations [44; 45; 46; 47]. These approaches introduce algebraic rules enabling entities and relations to be combined in principled ways. For instance, an object like a "blue car with red mirrors" may be

represented as

$$(\mathbf{v}_{\text{blue}} \otimes \mathbf{v}_{\text{car}}) \oplus (\mathbf{v}_{\text{red}} \otimes \mathbf{v}_{\text{mirrors}}), \tag{1.4}$$

where $\otimes$ and $\oplus$ denote well-designed composition and aggregation operators (either learned or fixed). Such formulations transcend the expressive limits of purely additive or multiplicative schemes and point toward a more algebraically grounded foundation for compositional reasoning in latent spaces, allowing for novel, powerful solutions such as the ones presented Chapters 5 and 6.

1.3 Contributions and Thesis Scope

This thesis focuses on formal ways to understand and operate on the latent spaces of trainable or fixed models and the multiple practical benefits such approaches can offer. Using tools from applied mathematics, we derive solutions to challenging, important problems in Machine Learning and Computer Vision that are more efficient and expand the capabilities of the existing proposals across multiple axes.

In this thesis we present latent-based approaches which, while grounded in concrete mathematical formalizations, focus on improving the existing state of the art across multiple fronts. For example, we will present ways to formalize and impose additional constraints in a model so that it better reflects the underlying real-world scenario (Chapters 3 and 4). Or, how can someone think about Foundational Models and how operating in their latent space can improve their performance across multiple fronts, from hallucination alleviation, to guardrailing, and simpler prompt engineering Chapters 4 and 7). Even further, we will show how ideas from Symbolic AI can be fused with current models and lead to solutions with better performance profiles in terms of runtime, data requirements, and more (Chapters 5 and 6).

More specifically, our contributions are grounded in three complementary theoretical frameworks: (a) **Category Theory** [48; 49], a diagrammatic and formal

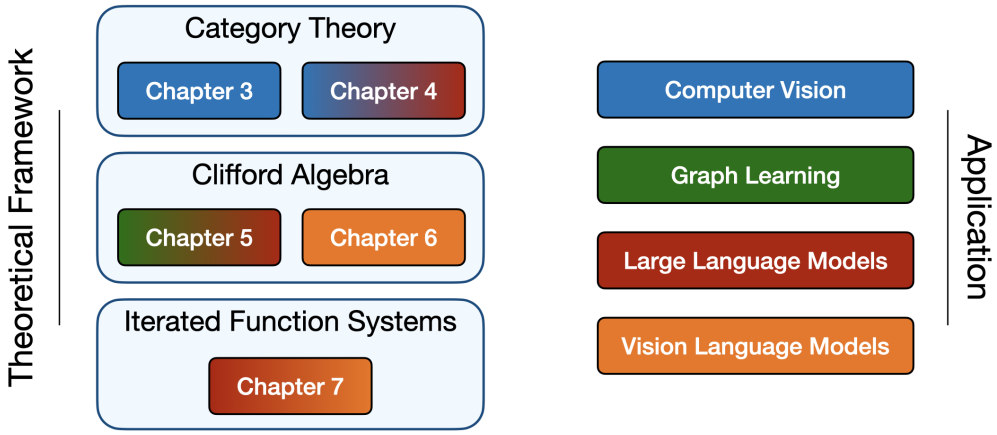


Figure 1.5: Overall scope and organization.

mathematical language that allows us to reason about structural properties independently of model or data specifics; (b) **Clifford (Geometric) Algebra** [43; 50], which allows us to realize these structural properties, by providing principled tools for composing and decomposing representations far beyond standard operations such as vector addition or matrix–vector products; and (c) **Iterated Function Systems** [51; 52], which offer an explanatory model of the representation compositions and decompositions in pretrained large language and multimodal models, enabling simple, training-free algorithmic interventions.

Together, these frameworks endow us with new analytical and algorithmic capabilities; capabilities that translate into practical solutions with appealing properties across a wide range of applications, from classical tasks such as image harmonization and graph learning to modern AI challenges, including LLM-based synthetic data generation and hallucination reduction in VLMs.

In the following sections, we introduce each problem domain, outline its unique challenges and opportunities, and preview how the ensuing chapters address them through the lens of these theoretical tools.

1.3.1 Chapter 3. Pooling Image Datasets with Multiple Covariate Shift and Imbalance

Small sample sizes are common in many disciplines, which necessitates pooling roughly similar datasets across multiple institutions to study weak but relevant associations between images and disease outcomes. Such data often manifests a shift/imbalance in covariates (i.e., secondary non-imaging data). Controlling for such nuisance variables is common within standard statistical analysis, but the ideas do not directly apply to overparameterized models. Consequently, recent work has shown how strategies from invariant representation learning provide a meaningful starting point, but the current repertoire of methods is limited to accounting for shifts/imbalances in just a couple of covariates at a time.

In Chapter 3, we show how viewing this problem from the perspective of Category theory provides a simple and effective solution that completely avoids elaborate multi-stage training pipelines that would otherwise be needed. We show the effectiveness of this approach via extensive experiments on real datasets. Further, we discuss how this style of formulation offers a unified perspective on at least 5+ distinct problem settings, from self-supervised learning to matching problems in 3D reconstruction.

1.3.2 Chapter 4. Understanding Multi-compositional learning in Vision and Language models via Category Theory

Pre-trained large language models (and multi-modal models) offer excellent performance across a wide range of tasks. Despite their effectiveness, we have limited knowledge of their internal knowledge representation.

In Chapter 4, we examine the latent structure of these models. To get started, we use the classic problem of Compositional Zero-Shot Learning (CZSL) as an example, and first provide a structured view of the latent space that any general model (LLM or otherwise) should nominally respect. We obtain a practical solution to the CZSL problem that can deal with both Open and Closed-World

single-attribute compositions as well as multi-attribute compositions with relative ease, where we achieve performance competitive with methods designed solely for that task (i.e., adaptations to other tasks are difficult). Then, we extend this perspective to the analysis of existing LLMs and ask to what extent they satisfy our axiomatic definitions. Our analysis shows a mix of interesting and unsurprising findings, but nonetheless suggests that our criteria are meaningful and may yield a more structured approach for potential incorporation in training such models, strategies for additional data collection, and diagnostics beyond visual inspection.

1.3.3 Chapter 5. FoGE: Fock Space inspired encoding for graph prompting

Recent results show that modern Large Language Models (LLM) are capable of understanding and answering questions about structured data such as graphs. Existing proposals often use some description of the graph to create an "augmented" prompt fed to the LLM. For a chosen class of graphs, if a well-tailored graph encoder is deployed to play together with a pre-trained LLM, the model can answer graph-related questions well. Current solutions to graph-based prompts range from graph serialization to graph transformers.

In Chapter 5, we show that the use of a parameter-free graph encoder based on Fock space representations, a concept borrowed from physics, is remarkably versatile in this problem setting. The simple construction, with a few small adjustments, can provide rich and informative graph encodings for a wide range of different graphs. We investigate the use of this idea for prefix-tuned prompts, leveraging the capabilities of a pre-trained, frozen LLM. The modifications lead to a model that can answer graph-related questions – from simple graphs to proteins to hypergraphs – effectively and with minimal, if any, adjustments to the architecture. Our work significantly simplifies existing solutions and generalizes well to multiple different graph-based structures effortlessly.

1.3.4 Chapter 6. ReCo: Reminder Composition Mitigates Hallucinations in Vision-Language Models

Vision Language Models (VLMs) show impressive capabilities in integrating and reasoning with both visual and language data. But these models make mistakes. A common finding – similar to LLMs – is their tendency to hallucinate, i.e., generate plausible sounding text which is not grounded in the visual input, or at worst, is contradictory. A growing consensus attributes this behavior to an over-reliance on language – especially as the generation progresses, the model suffers from a "fading memory effect" with respect to the provided visual input.

In Chapter 6, we study mechanisms by which this behavior can be controlled. Specifically, using ideas from geometric algebra and relational compositions, we propose the addition of a small, trainable module (named ReCo) on top of any VLM – no other modification is needed. We show that such a lightweight module is able to mitigate the fading memory effect on three of the most widely used VLMs (InstructBLIP, LLaVA, MiniGPT4), where we see performance improvements on multiple benchmarks. Additionally, we show that our module can be combined with many of the other approaches for reducing hallucination where we achieve improved results for each one.

1.3.5 Chapter 7. Concept Attractors in LLMs and their Applications

Large language models (LLMs) often map semantically related prompts to similar internal representations at specific layers, even when their surface forms differ widely. We show that this behavior can be explained through Iterated Function Systems (IFS), where layers act as contractive mappings toward concept-specific Attractors.

In Chapter 7, we leverage this insight and develop simple, training-free methods that operate directly on these Attractors to solve a wide range of practical tasks, including language translation, hallucination reduction, guardrailing, and

synthetic data generation. Despite their simplicity, these Attractor-based interventions match or exceed specialized baselines, offering an efficient alternative to heavy fine-tuning, generalizable in scenarios where baselines underperform.

Finally, in Chapter 8, we summarize the contributions of this thesis and discuss the future directions of our research.

2 BACKGROUND

This chapter serves as a reminder, or an introduction, to the basic building blocks used in all of the subsequent chapters. First, in Section 2.1 we set the basic notation conventions followed throughout this thesis. Following this, in Sections 2.2 to 2.4, we present the basic mathematical frameworks used to ground our propositions. Finally, in Section 2.5, we present the basic principles and architecture of Large Language and Multimodal Models.

2.1 Notation

We use bold uppercase letters to denote matrices ($\mathbf{A}, \mathbf{B}, \cdots$) and bold lowercase letters for vectors ($\mathbf{a}, \mathbf{b}, \cdots$), which we assume are column vectors by default. For scalars we use regular letters ($a, b, \cdots, A, B, \cdots$). The lowercase ones are used to denote iterating indices or specific elements and Objects of a Category, while the uppercase ones are used to denote size and dimension. Further, we use $\mathbb{A}, \mathbb{B}, \cdots$ to denote sets or spaces, and $\mathfrak{A}, \mathfrak{B}, \cdots$ for Categories. Functions and mappings are denoted with letters $\mathcal{A}, \mathcal{B}, \cdots$. To distinguish between Functors and Morphisms in Categories, we use the superscript $\cdot^f$ for Functors. We preserve $\mathbb{R}$ for the space of real numbers, $\mathbb{Z}$ for the space of integers, and $\mathbb{C}$ for the space of complex numbers. Finally, in many cases, the elements of a set $\mathbb{S}$ of cardinality D will be denoted as $s_1, s_2, \cdots, s_D$, instead of D unique letters.

For a matrix $\mathbf{A}$, we use $[\mathbf{A}]_i$ or $[\mathbf{A}]_{i,:}$ for its i-th row and $[\mathbf{A}]_{:,j}$ for its j-th column vector. Additionally, $[\mathbf{A}]_{i,j}$ denotes the j-th element of the i-th row. Similarly, we use $[\mathbf{a}]_i$ to denote the i-th element of a vector $\mathbf{a}$. For a set $\mathbb{A}$ of vectors, we use the notation $\mathbf{a}_i$ to denote the i-th vector of the set.

Given two D-dimensional vectors $\mathbf{a}, \mathbf{b}$, their inner product is defined as:

$$\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^T \mathbf{b} = \sum_{i=1}^{D} [\mathbf{a}]_i [\mathbf{b}]_i \tag{2.1}$$

while the inner product of two matrices $\mathbf{A}, \mathbf{B}$ of dimension $D \times T$ is defined as:

$$\langle \mathbf{A}, \mathbf{B} \rangle = \sum_{i=1}^{D} \sum_{j=1}^{T} [\mathbf{A}]_{i,j} [\mathbf{B}]_{i,j} \tag{2.2}$$

2.2 Category Theory

Category theory offers a way to study abstract "structures" [53; 54] by focusing on how entities relate to one another rather than on their internal composition. At its core lies the notion of a **Category**, a mathematical setting that captures systems through their interactions. A Category consists of two components: **(a)** a collection of **Objects**, representing individual entities such as images, vector spaces, or even datasets; and **(b)** a collection of **Morphisms** (or "arrows") connecting those Objects, representing transformations, mappings, or relationships between them. Each Object has a distinguished identity Morphism, and Morphisms can be composed when they are compatible. Formally, for a Category $\mathfrak{C}$ and any three Objects $a, b, c \in \mathfrak{C}$, the existence of Morphisms $\mathcal{F} : a \to b$ and $\mathcal{G} : b \to c$ ensures the existence of their composite $\mathcal{G} \circ \mathcal{F} : a \to c$, itself a Morphism in $\mathfrak{C}$.

Remark 2.1. *Assume a Category $\mathfrak{S}$ with at least three Objects s_1, s_2, s_3. The composition of two Morphisms $\mathcal{F} : s_1 \to s_2$ and $\mathcal{G} : s_2 \to s_3$ is a well-defined Morphism and it is denoted as $\mathcal{G} \circ \mathcal{F} : s_1 \to s_3$ (we should read it as $\mathcal{G}$ after $\mathcal{F}$).*

Example 2.2. *Consider the Category of Sets $\mathfrak{S}$, in which the Objects are sets (denoted $\mathbb{S}_i$) and the Morphisms correspond to functions (or matchings) between sets. For three sets $\mathbb{S}_1, \mathbb{S}_2, \mathbb{S}_3$, assuming $\exists \mathcal{F} : \mathbb{S}_1 \to \mathbb{S}_2$, $\mathcal{G} : \mathbb{S}_2 \to \mathbb{S}_3$ then $\exists \mathcal{H} = \mathcal{G} \circ \mathcal{F} : \mathbb{S}_1 \to \mathbb{S}_3$.*

In the Category of Sets $\mathfrak{S}$, $id_{\mathbb{S}_i} : \mathbb{S}_i \to \mathbb{S}_i$ *corresponds to the identity function:* $id_{\mathbb{S}_i}(s) = s,\ \forall s \in \mathbb{S}_i,\ \forall \mathbb{S}_i \in \mathfrak{S}$.

A key philosophical move that Category Theory makes is to shift attention away from the Objects themselves and toward the Morphisms between them. Instead of analyzing the internal structure of a, b, and c, one studies how these Objects *interact*. This perspective enables a unifying language that cuts across domains: functions between sets, linear maps between vector spaces, convolutions between feature maps, or even data-processing pipelines can all be viewed as Morphisms once the appropriate Category is chosen. The emphasis on relationships rather than constructional details often reveals structural commonalities between otherwise disparate mathematical or computational systems. As a result, Category Theory provides a powerful abstraction mechanism, one that is especially valuable in areas like machine learning, where the mappings between representations often matter more than the representations themselves.

Another distinctive feature of the categorical viewpoint is its *diagrammatic* nature. Categories are naturally visual: Objects are represented as nodes and Morphisms as arrows, and the axioms of composition and identity become simple statements about paths in these diagrams. Commutative diagrams encode algebraic constraints in an intuitive way, often replacing lines of symbolic manipulation. This visual language is not merely pedagogical; it plays a central role in categorical reasoning. Entire proofs can sometimes be reduced to the assertion that "this diagram commutes", making arguments both more concise and more transparent.

Finally, Category Theory scales gracefully to increasingly sophisticated notions of structure. From basic Categories one can abstract further to *Functors* (structure-preserving maps between Categories [54, p. 13], *Natural Transformations* (Morphisms between Functors [54, p. 16]), and more elaborate constructions such as monoidal Categories [54, p. 156], limits and colimits [54, p. 104], and adjunctions [54, p. 77]. Each layer enriches the expressive power of the theory while preserving its core relational philosophy. These tools have become funda-

mental in modern mathematics, theoretical computer science, and increasingly in machine learning, where they offer a principled way to reason about composition, modularity, and equivalence of representations. In this sense, Category Theory serves not just as an abstract mathematical framework, but as a conceptual technology for organizing complex systems.

Definition 2.3 (Functor)**.** *Functors* $\mathcal{F}^f : \mathfrak{S} \to \mathfrak{T}$ *give a relationship between the Objects and the Morphisms of two different Categories: the source Category* $\mathfrak{S}$ *and the target Category* $\mathfrak{T}$*. The conditions below hold,*

(i) $\mathcal{F}^f(id_s) = id_{\mathcal{F}^f(s)} \quad \forall\, s \in \mathfrak{S}$

(ii) $\mathcal{F}^f(\mathcal{G} \circ \mathcal{H}) = \mathcal{F}^f(\mathcal{G}) \circ \mathcal{F}^f(\mathcal{H}) \quad \forall\, \mathcal{H} : s_1 \to s_2, \mathcal{G} : s_2 \to s_3,\ in\ \mathfrak{S}$

A Functor [54, p. 13] can therefore be understood as a structure-preserving translation between Categories. It maps each Object of the source Category $\mathfrak{S}$ to an Object of the target Category $\mathfrak{T}$, and each Morphism in $\mathfrak{S}$ to a corresponding Morphism in $\mathfrak{T}$, in a way that strictly respects the categorical structure. The two conditions in Definition 2.3 guarantee that identity Morphisms are preserved and that the order and meaning of composition is maintained. In other words, a Functor does not merely map elements; it transports the entire relational fabric of one Category into another without distorting the way arrows compose. This is why Functors are often described as "homomorphisms of Categories" or as the primary mechanism for comparing different mathematical worlds.

Functors also enable abstraction across levels: rather than analyzing the internal details of transformations within $\mathfrak{S}$, one studies how those transformations behave under their image in $\mathfrak{T}$. This proves especially powerful when the target Category encodes a simpler, more algebraic, or more familiar structure. For example, assigning to each topological space its corresponding fundamental group, or to each graph its associated adjacency matrix, are instances of Functors that extract structured invariants. In applied settings, Functors act as pipelines that convert complex Objects (such as datasets, images, or graphs) into representations that are easier to analyze or compute with, all while preserving the relationships

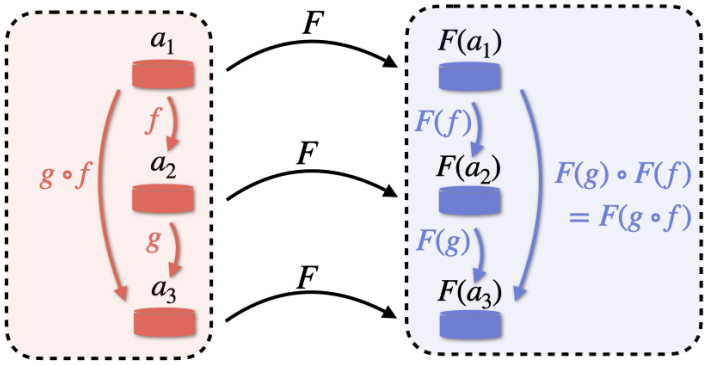


Figure 2.1: Two Categories $\mathfrak{A}$, $\mathfrak{B}$ and the mapping a Functor $\mathcal{F}^f : \mathfrak{A} \rightarrow \mathfrak{B}$ induces.

between them. Thus, Functors extend the categorical philosophy by providing a disciplined way to move between representational layers without losing essential structural information.

Example 2.4. *Consider the set of ImageNet [55] images and $\mathbb{R}^n$ as two categories. An image encoder, e.g., ResNet [13], which maps each image to an embedding vector in $\mathbb{R}^n$ is an example of a Functor from the Category of Images to the Category of Vectors.*

Example 2.5. *Consider the Category $\mathfrak{Eng}_{to}\mathfrak{Fr}$ where each Object represents a sentence in English or French, and there is a unique Morphism (translation) between corresponding English and French sentences. Mapping this Category to an embedding space involves finding a Functor $\mathcal{E}nc^f$ that preserves the structure. Assuming $\mathcal{E}nc^f$ has an inverse Morphism ($Dec^f = \mathcal{E}nc^{f^{-1}}$), we recover the standard language translation model.*

Remark 2.6. *The reader will see why the specification above is general, but useful. The Functor $\mathcal{F}^f$ takes us from the first Category to the next in a way that the structure of the source Category (Objects* and *arrows) is fully preserved in the target Category because of the two conditions above.*

The notion of a Product Category arises naturally once Functors are in view. While a single Functor allows us to translate structure from one Category into another, many constructions in mathematics and applications require reasoning about *multiple* Categories simultaneously. For instance, one may wish to track how pairs of Objects evolve under paired transformations, or to describe processes whose state is inherently composite (e.g., an image paired with its label, a graph paired with a signal, or a dataset paired with metadata). To formalize such settings, Category Theory introduces the *Product Category*, which "packages" two Categories together in a way that preserves their internal structures while enabling joint manipulation [54] (pp. 36-40).

Crucially, the Product Category is designed so that its structure interacts cleanly with Functors. The canonical projection mappings from a product to its components are themselves Functors, reflecting the fact that the product behaves like a categorical analogue of the Cartesian product of sets. These projection Functors extract the first and second components of each Object and Morphism, and they play an essential role in many higher-level constructions, including universal properties, limits, and bifunctors. In this sense, Product Categories occupy an important conceptual bridge: they are simple to define, yet they provide the scaffolding needed to move from reasoning about single Categories to reasoning about structured pairs of them.

Definition 2.7 (Product Category)**.** *For two Categories* $\mathfrak{A}, \mathfrak{B}$*, we define the Product Category* $\mathfrak{A} \times \mathfrak{B}$ *as the Category with the properties:*

(i) *Its Objects are pairs* (a, b) $\forall a \in \mathfrak{A}, b \in \mathfrak{B}$

(ii) *Its Morphisms are the pairs* $(\mathcal{F}, \mathcal{G}) : (a_1, b_1) \to (a_2, b_2)$ $\forall \mathcal{F} : a_1 \to a_2, \mathcal{G} : b_1 \to b_2$

(iii) *The composition of Morphisms is defined element-wise as* $(\mathcal{F}_2, \mathcal{G}_2) \circ (\mathcal{F}_1, \mathcal{G}_1) = (\mathcal{F}_2 \circ \mathcal{F}_1, \mathcal{G}_2 \circ \mathcal{G}_1)$ *for all composable Morphisms.*

For each Product Category [54, p. 36] $\mathfrak{A} \times \mathfrak{B}$, we can also define two Functors $\mathcal{P}^f_{\mathfrak{A}} : (\mathfrak{A} \times \mathfrak{B}) \to \mathfrak{A}$, $\mathcal{P}^f_{\mathfrak{B}} : (\mathfrak{A} \times \mathfrak{B}) \to \mathfrak{B}$ that correspond to the "projections":

- $\mathcal{P}^f_{\mathfrak{A}}\big((a, b)\big) = a$; $\mathcal{P}^f_{\mathfrak{A}}\big((\mathcal{H}, \mathcal{G})\big) = \mathcal{H}$

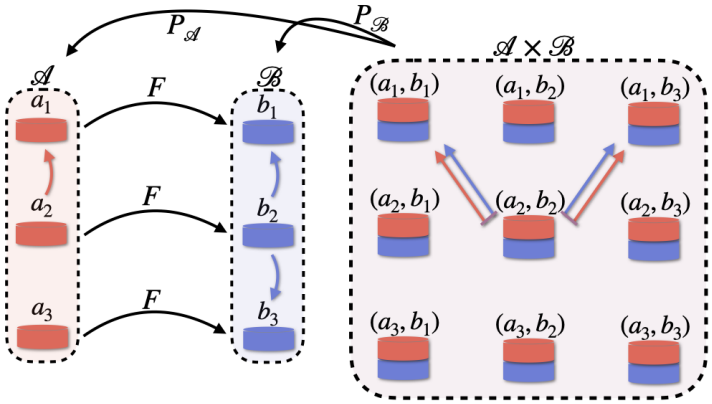


Figure 2.2: A Functor $\mathcal{F}^f : \mathfrak{A} \to \mathfrak{B}$ and the Product Category $\mathfrak{A} \times \mathfrak{B}$ shown graphically (self-loops omitted). The Functors $\mathcal{P}^f_{\mathfrak{A}}, \mathcal{P}^f_{\mathfrak{B}}$ also map the Morphisms between $(a_2, b_2) \to (a_1, b_3)$ and $(a_2, b_2) \to (a_1, b_1)$, i.e., the Morphisms $a_2 \to a_1$ and $b_2 \to b_1, b_2 \to b_3$.

- $\mathcal{P}^f_{\mathfrak{B}}\big((a, b)\big) = b$; $\mathcal{P}^f_{\mathfrak{B}}\big((\mathcal{H}, \mathcal{G})\big) = \mathcal{G}$

for each Object and Morphism of $\mathfrak{A} \times \mathfrak{B}$ respectively.

Why Category Theory? Category Theory is used in formal methods for reasoning about types and structure [56]. With functional programming approaches gaining prominence in deep learning, practical uses of Category theory can become feasible. A Category-Theory-based formulation streamlines the creation of compositional constructs. Ongoing efforts in formulating deep learning concepts via Category Theory, while nascent, are yielding interesting new algorithms [57; 58; 59; 60; 61; 62]. Category Theory allows cleaner and more general results, while maintaining the mathematical correctness and rigor desired.

2.3 Clifford Algebra

Category theory is a rigorous mathematical language that allows us to argue about concepts, structures, and relationships in a formal and principled way; in spirit, it functions like an architectural blueprint for Machine Learning models. Its great strength lies in its level of abstraction: categorical tools are deliberately detached

from the specifics of any particular learning architecture, enabling us to reason about the design space of models, the transformations between them, and the ways in which complex systems can be composed. However, this abstraction is also a limitation. Category theory does not itself prescribe how to construct individual models, how to represent data, or how to select concrete operations and hyperparameters. To understand and manipulate the internal mechanics of models, we require a mathematical framework that operates at a more concrete, representational level.

Clifford Algebra (or *Geometric Algebra*) [43; 42] provides precisely such a framework. Whereas Category Theory offers a bird's-eye conceptual view, Clifford Algebra functions at the level of vectors, transformations, and geometric structure. It extends linear algebra by introducing a rich algebraic system, typically denoted as a ring $\mathcal{G}$ equipped with two operations, $\oplus$ and $\otimes$ [63; 42]. The elements of $\mathcal{G}$ (called *multivectors*) form a graded structure: $\mathcal{G}$ contains scalars (or 0-vectors), ordinary vectors (or 1-vectors), as well as higher-degree elements such as 2-vectors, 3-vectors, and so on. The subset of $\mathcal{G}$ consisting of all k-vectors is typically denoted by $\mathcal{G}_k$. These graded components interact through the *geometric product*, an operation that simultaneously generalizes the dot product and the wedge product, capturing both metric and orientation information in a single algebraic object.

This representational richness makes Clifford Algebra well-suited for modeling transformations that are inherently geometric, such as rotations, reflections, projections, and the interactions between subspaces. Unlike ordinary vector spaces, where only magnitude and direction are represented, multivectors can encode higher-order geometric primitives (planes, volumes, and orientation) and manipulate them algebraically in a unified manner. This leads to compact formulations of classical operations as well as new ones that have no simple expression in standard vector algebra. In modern machine learning, such expressive structure can be leveraged to build models that reason about geometry more naturally, capture interactions between features more faithfully, and exploit symmetries at a

deeper algebraic level (e.g., [64; 65]).

In a way, Clifford Algebra complements the categorical perspective by providing a concrete representational toolkit. While Category Theory informs how learning systems may be organized and composed at a conceptual level, Clifford Algebra provides a powerful language for describing and manipulating the geometric content of data and the algebraic behavior of transformations.

Remark 2.8. *A k-vector can be defined as the geometric product of k mutually orthogonal* 1*-vectors. If $\{\mathbf{v}_1, \dots, \mathbf{v}_k\} \subset \mathcal{G}_1$ and $\mathbf{v}_i \perp \mathbf{v}_j$ for all $i \neq j$, then*

$$\mathbf{v}_1 \otimes \cdots \otimes \mathbf{v}_k \in \mathcal{G}_k. \tag{2.3}$$

Alternatively, any k-vector can be defined using the wedge product ($\wedge$), which extracts the fully antisymmetric part of a geometric product. Explicitly,

$$\mathbf{v}_1 \wedge \cdots \wedge \mathbf{v}_k = \frac{1}{k!} \bigoplus_{\sigma \in S_k} \operatorname{sign}(\sigma)\, \mathbf{v}_{\sigma(1)} \otimes \cdots \otimes \mathbf{v}_{\sigma(k)}, \tag{2.4}$$

where the sum ranges over all permutations in the symmetric group S_k [66] (p. 31). When the 1*-vectors are mutually orthogonal, the antisymmetrization does not alter the product, and the geometric and wedge products coincide:*

$$\mathbf{v}_1 \otimes \cdots \otimes \mathbf{v}_k = \mathbf{v}_1 \wedge \cdots \wedge \mathbf{v}_k \quad \Longleftrightarrow \quad \mathbf{v}_1 \perp \cdots \perp \mathbf{v}_k. \tag{2.5}$$

To formalize the construction of a Clifford algebra, we begin with a vector space $\mathbb{W}$ over a field $\mathbb{K}$ (typically $\mathbb{K} = \mathbb{R}$ or $\mathbb{C}$). The space $\mathbb{W}$ carries the usual linear operations—addition, scalar multiplication, subtraction—and is equipped with a symmetric bilinear form or inner product $\langle \cdot, \cdot \rangle$ that measures lengths and angles between vectors.

The first step is to build the tensor algebra $T(\mathbb{W})$. Conceptually, $T(\mathbb{W})$ contains all finite tensor products of elements from $\mathbb{W}$: $T(\mathbb{W}) = \mathbb{K} \oplus \mathbb{W} \oplus (\mathbb{W} \otimes \mathbb{W}) \oplus (\mathbb{W} \otimes \mathbb{W} \otimes \mathbb{W}) \oplus \cdots$, together with all linear combinations of such products. This construction freely adjoins all possible products of vectors, without imposing

any identities beyond bilinearity. For example, for $\mathbf{u}, \mathbf{v} \in \mathbb{W}$, the tensors $\mathbf{u} \otimes \mathbf{v}$ and $\mathbf{v} \otimes \mathbf{u}$ are distinct elements of $T(\mathbb{W})$, and no relation is assumed between them unless it follows purely from linearity. To obtain the Clifford algebra, we impose the geometric constraint that each vector should square to its negative squared norm. To enforce this, we introduce the ideal $I(\mathbb{W})$ of $T(\mathbb{W})$ generated by the elements $\mathbf{w} \otimes \mathbf{w} + \langle \mathbf{w}, \mathbf{w} \rangle \mathbf{1}$, where $\mathbf{1}$ denotes the multiplicative identity of $T(\mathbb{W})$. The ideal $I(\mathbb{W})$ collects all algebraic consequences of these relations and is closed under multiplication with arbitrary elements of $T(\mathbb{W})$, meaning that any tensor expression that differs by repeated use of the relation $\mathbf{w} \otimes \mathbf{w} = -\langle \mathbf{w}, \mathbf{w} \rangle \mathbf{1}$ is identified in the quotient.

Definition 2.9. *Let $\mathbb{W}$ be a vector space over a field $\mathbb{K}$ equipped with a quadratic form $Q : \mathbb{W} \to \mathbb{K}$. The **Clifford algebra** of $(\mathbb{W}, Q)$, denoted $\mathfrak{Cl}(\mathbb{W}, Q)$, is the quotient algebra $\mathfrak{Cl}(\mathbb{W}, Q) = T(\mathbb{W}) \,/\, I(\mathbb{W}, Q)$.*

Forming the quotient algebra means that all elements of $I(\mathbb{W}, Q)$ are declared equal to zero. In particular, for every $\mathbf{w} \in \mathbb{W}$, the identity $\mathbf{w} \otimes \mathbf{w} = -\langle \mathbf{w}, \mathbf{w} \rangle \mathbf{1}$ holds inside $\mathfrak{Cl}(\mathbb{W}, Q)$. From bilinearity, we obtain the familiar anticommutation relation $\mathbf{uv} + \mathbf{vu} = -2\langle \mathbf{u}, \mathbf{v} \rangle \mathbf{1}$, which, in an orthonormal basis $\{\mathbf{e}_i\}$, reduces to $\mathbf{e}_i \mathbf{e}_j + \mathbf{e}_j \mathbf{e}_i = -2\delta_{ij}$. These relations are the algebraic backbone of the geometric product and encode both metric information and orientation. Standard references include [67; 43].

Remark 2.10. *A 2-vector represents an oriented plane, a 3-vector an oriented volume element, and higher-grade elements encode higher-dimensional subspaces (Figure 2.3). This stands in contrast to the inner product, which collapses two vectors to a scalar and therefore cannot retain information about their span or orientation. The geometric product simultaneously generalizes both the inner product and the wedge (outer) product, enabling compact representations of reflections, rotations, and other geometric transformations used throughout modern machine learning [68; 14; 69].*

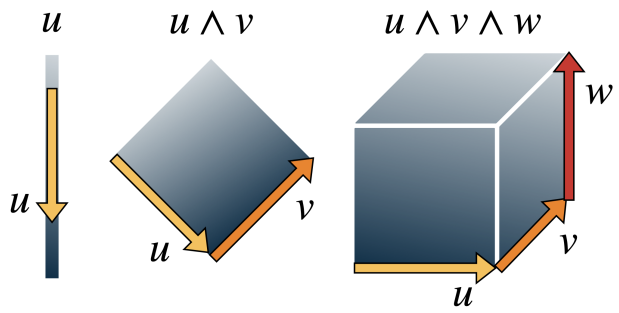


Figure 2.3: A geometric representation of multi-vectors, from 1-vector to 3-vector.

2.3.1 Vector Symbolic Architectures

Vector Symbolic Architectures (VSA), originally developed within the symbolic AI tradition, offer a computational framework for representing and manipulating structured information via high-dimensional vectors [70; 71]. In many ways, VSAs can be viewed as a practical instantiation of the algebraic principles behind Clifford Algebra: both rely on operations that preserve similarity, support compositionality, and enable distributed representations of complex structures. Whereas Clifford Algebra provides an elegant theoretical account of how symbols can be bound, merged, and transformed within graded algebras, VSAs translate these principles into concrete, scalable mechanisms for vector computation.

Motivated by early ideas in symbolic and connectionist AI [72; 73; 74; 75], VSAs operationalize the classic "bind" and "bundle" primitives within high-dimensional vector spaces. Binding typically corresponds to an invertible, pairwise interaction between vectors, whereas bundling combines multiple pieces of information into a single representation, often through superposition. These operations allow VSAs to represent variable–value pairs, sequences, hierarchical structures, or relations while maintaining the robustness and noise tolerance characteristic of distributed codes.

A rich line of work has explored the mathematical foundations of these vector operations. Holographic Reduced Representations (HRR) [45], Vector-Derived Transformation Binding (VTB) [76], and related frameworks [68] provide mech-

anistic derivations for constructing symbols, defining merge operations, and ensuring invertibility or approximate invertibility of bindings. Other studies have connected VSA operations to tensor product representations, convolution algebras, and structured latent spaces.

An intriguing direction investigates the relation between VSAs and Fock space representations, where symbolic structures are encoded via creation and annihilation operators acting on high-dimensional Hilbert spaces. This perspective has been applied to problems such as trajectory encoding and temporal pattern analysis [77], further highlighting the deep links between VSAs, algebraic frameworks, and physical interpretations of symbolic computation.

Overall, VSAs offer a bridge between the theoretical expressivity of algebraic formalisms, such as Clifford Algebra, and the empirical tractability required for large-scale machine learning systems. Their compositional operations, similarity-preserving properties, and robustness to noise make them a compelling substrate for structured reasoning in modern neural architectures [46; 78].

2.4 Iterated Function Systems

While Clifford Algebra offers a principled and expressive toolkit for reasoning about latent spaces and for manipulating high-dimensional representations through well-defined algebraic operations, these operations are ultimately prescriptive: they encode how we choose to act on representations, rather than how trained neural networks actually do so. In contrast, the internal transformations learned by deep neural networks emerge implicitly from optimization and data, and may not align with any predefined algebraic structure. This gap has motivated a variety of mathematical frameworks that aim to describe, rather than impose, the intrinsic dynamics of networks trained to completion. One such framework is that of Iterated Function Systems, which shifts the focus from explicit algebraic operators to the study of repeated, learned transformations and their long-term dynamical behavior.

For a complete metric space $(\mathbb{X}, \mathcal{D})$, an *iterated function system (IFS)* [51] is a finite collection of **contraction** mappings $\mathcal{F} = \{\mathcal{F}_i : \mathbb{X} \to \mathbb{X} \mid i = 1, 2, \ldots, N\}$, meaning that for each i there is a constant $s_i \in [0, 1)$ such that $\mathcal{D}\big(\mathcal{F}_i(x), \mathcal{F}_i(y)\big) \leq s_i\ \mathcal{D}(x, y),\ \forall x, y \in \mathbb{X}$.

The corresponding *Hutchinson operator* [52] is defined as $\mathcal{H} : \mathbb{K}(\mathbb{X}) \to \mathbb{K}(\mathbb{X})$ on the space $\mathbb{K}(\mathbb{X})$ of non-empty compact subsets of $\mathbb{X}$, endowed with the Hausdorff metric h, by

$$\mathcal{H}(\mathbb{A}) = \bigcup_{i=1}^{N} \mathcal{F}_i(\mathbb{A}), \quad \mathbb{A} \in \mathbb{K}(\mathbb{X}) \tag{2.6}$$

Theorem 2.11 ([52] (thm. 3))**.** *Under the above assumptions, there exists a unique non-empty compact set $\mathbb{A}^* \subset \mathbb{X}$ satisfying $\mathcal{H}(\mathbb{A}^*) = \mathbb{A}^*$. Moreover, for any initial non-empty compact set $\mathbb{A}_0 \subset \mathbb{X}$, the sequence defined by $\mathbb{A}_{k+1} = \mathcal{H}(\mathbb{A}_k),\ k = 0, 1, 2, \ldots$ converges (in the Hausdorff metric) to $\mathbb{A}^*$*

Thus $\mathbb{A}^*$ is the unique **attractor** (or invariant set) of the IFS, and is globally attracting for all compact seeds [51].

2.4.1 The Collage Theorem

In many practical and theoretical contexts, one aims to find an IFS whose attractor approximates a given compact target set $\mathbb{L} \subset \mathbb{X}$. The **Collage Theorem** [51] provides such a constructive route:

Theorem 2.12 (Collage Theorem [51] (sec. 3.7))**.** *Let $\mathbb{L} \subset \mathbb{X}$ be a non-empty compact set, and let $\varepsilon > 0$. Suppose there is a finite family of contraction maps $\{\mathcal{W}_i\}_{i=1}^{N}$ on $\mathbb{X}$ with contraction constant $s < 1$ such that*

$$h\Big(\mathbb{L},\ \bigcup_{i=1}^{N} \mathcal{W}_i(\mathbb{L})\Big) \leq \varepsilon. \tag{2.7}$$

Then the attractor $\mathbb{A}$ *of the IFS characterized by* $\{\mathcal{W}_i\}$ *satisfies*

$$h(\mathbb{L}, \mathbb{A}) \leq \frac{\varepsilon}{1-s}. \tag{2.8}$$

with $s = \max_i s_i$.

Hence, if a target compact set is approximately invariant under the union of the maps, then the corresponding IFS-attractor approximates the target set (in Hausdorff distance).

2.4.2 Iterated Function Systems and Machine Learning

Several recent works provide indirect yet compelling evidence that modern transformer architectures exhibit behavior reminiscent of an IFS or attractor-based representation dynamics. For instance, [79] shows that, for transformer models trained on data generated by hidden-state processes, the network's residual-stream activations linearly encode the associated "belief states". Remarkably, when the underlying belief-state geometry is nontrivial (potentially even fractal) the learned representations faithfully reflect that geometry (as predicted by theory). More broadly, frameworks such as that of [80] reinterpret layer-wise transformer updates as a discrete approximation of a continuous dynamical system. Under mild Lipschitz conditions, the dynamics converge toward a unique solution, and, when the mappings satisfy contractivity-like conditions, perturbations decay across layers, suggesting contractive, attractor-like behavior in the latent space.

Taken together, these findings suggest that transformer models may implicitly implement dynamics akin to an IFS: successive layer transformations act as contractive (or approximately contractive) maps on the latent space, pushing representations toward stable invariant sets (attractors or belief-state manifolds). This viewpoint offers a principled bridge between classical IFS theory and the internal geometry of deep learning models, and motivates interpreting latent convergence phenomena in LLMs through the lens of dynamical-systems theory (Chapter 7).

2.5 Large Language Models

In 2017, [14] proposed a new mechanism; an alternative to the intuitive but problematic Recurrent Neural Networks. The new architecture, named Transformer, fundamentally changed the way we approach time-dependent data such as text and, more importantly, allowed us to scale neural networks to sizes that had previously been unattainable. This shift led to a rapid surge of models of ever-increasing scale, from BERT [81] to the widely used Large Language Models (LLMs) such as GPT [82; 83] and Gemini [84].

At its core, a Transformer is based on the Attention mechanism; a mechanism inspired by databases that allows us to modify latent representations and capture short and long-range dependencies. Assuming a sequence of tokens $\{t_1, \cdots, t_N\}$ that is first transformed to a sequence of high-dimensional latent representations $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\} \in \mathbb{R}^{D\times N}$, Attention is based on three learnable matrices: $\mathbf{W}_K, \mathbf{W}_V, \mathbf{W}_Q$ which stand for Key, Value, and Query respectively. Assuming, for simplicity, $\mathbf{X} = [\mathbf{x}_1 : \cdots : \mathbf{x}_N]^T$, we define

$$\begin{aligned} \mathbf{K} &= \mathbf{X} \cdot \mathbf{W}_K \\ \mathbf{V} &= \mathbf{X} \cdot \mathbf{W}_V \\ \mathbf{Q} &= \mathbf{X} \cdot \mathbf{W}_Q \end{aligned} \tag{2.9}$$

and transform the latent representations as

$$\mathbf{X} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{D}}\right)\mathbf{V} \tag{2.10}$$

Remark 2.13. *Attention transforms a sequence of embeddings by replacing each one with a weighted average of all value vectors, where the weights reflect the similarity between the corresponding query and key. In this sense, it behaves like a differentiable, content-addressable lookup mechanism.*

The wide success of LLMs can be attributed mainly to two factors. First,

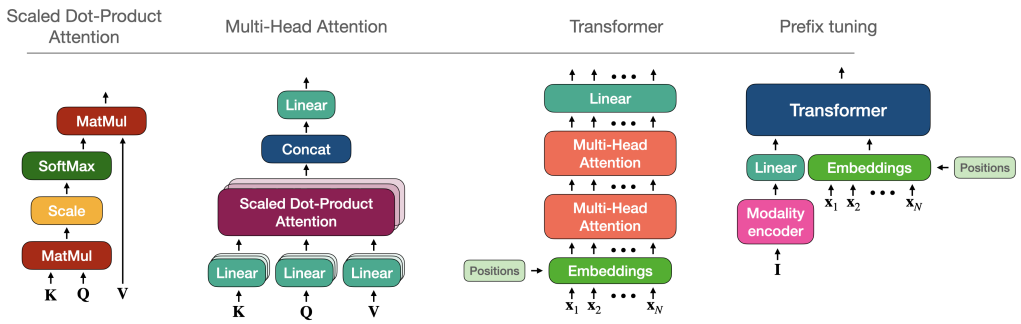


Figure 2.4: The building blocks of Transformers and Multimodal LLMs.

the extreme scalability of the attention mechanism and, consequently, of the Transformer architecture: stacking multiple layers and adding multiple heads per layer is not only feasible, but consistently improves empirical performance. Second, the adoption of self-supervised training: instead of relying on curated labeled datasets, LLMs are trained through large-scale next-token prediction on massive corpora collected from the internet [82; 85; 86; 83].

In addition to these two driving forces, the Transformer framework introduced an architectural prior that is remarkably well aligned with a broad class of sequence- and token-based tasks. Attention allows each token to interact with every other token through a learned, content-dependent weighting. This enables the model to capture long-range dependencies, latent hierarchical structures, and subtle contextual relationships; capabilities that were notoriously difficult for earlier recurrent models. The uniform block structure of the Transformer, comprising multi-head attention, position-wise feed-forward layers, and normalization, provides an unusual combination of expressive power and architectural simplicity. Its regularity makes it both amenable to distributed training and highly conducive to scaling, which has been a major catalyst for the explosive growth in model size over the past few years.

A particularly remarkable aspect of LLMs is the emergence of qualitatively new behaviors as their capacity increases. While early language models yielded predictable performance improvements with scale, modern LLMs exhibit emer-

gent abilities such as multi-step arithmetic, in-context learning, code generation, and rudimentary forms of logical and commonsense reasoning [87; 88]. These abilities arise naturally from optimization on the next-token prediction objective, without explicit supervision or architectural modifications. This phenomenon has motivated extensive research into scaling laws [89], which demonstrate that performance gains follow predictable power-law trends when jointly increasing model size, dataset size, and compute budget.

2.5.1 Multimodal Language Models

Ignoring many implementation details, a Transformer (and consequently an LLM) can be viewed as a parameterized function that processes an input sequence of vectors $\{\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_N^{(0)}\}$, iteratively transforming them into new sequences $\{\mathbf{x}_1^{(i)}, \ldots, \mathbf{x}_N^{(i)}\}$ after each layer i. While Transformers were originally proposed for natural language processing, where each $\mathbf{x}_j^{(0)}$ corresponds to a learned token embedding, subsequent work demonstrated that the architecture is sufficiently general to process alternative modalities, such as images [90]. In these settings, an external encoder maps raw inputs (e.g., image patches, audio spectrogram segments, or video frames) into a sequence of vectors compatible with the Transformer's input space.

Building on this observation, *Multimodal Large Language Models (MLLMs)* were introduced. These models preserve the linguistic, reasoning, and world-knowledge capabilities of pretrained LLMs while additionally being equipped to interpret and respond to inputs from multiple modalities (e.g., images, audio, video, or structured data). Despite substantial variation in architectural choices (including differences in modality encoders, fusion strategies, training protocols, and alignment objectives) the majority of successful MLLM designs adopt a common mechanism often referred to as *prefix-based integration*, most notably instantiated through *Prefix Tuning* [91]; a mechanism employed in our construction in Chapter 5.

Under the Prefix Tuning paradigm, one begins with a pretrained unimodal

LLM and reserves the first M input positions, $\left\{\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_M^{(0)}\right\}$ for embeddings derived from another modality (although other similar variations exist). These embeddings (typically produced by a modality-specific encoder trained either independently or jointly) function as a "soft prompt" that conditions the LLM on the non-textual input. The combined system is subsequently fine-tuned, either partially or end-to-end, on multimodal datasets in which each non-textual input is paired with textual supervision. During this process, the LLM learns to interpret the prefix vectors as latent representations of the new modality while largely preserving its pretrained linguistic structure and capabilities.

Although the multimodal alignment stage is usually only a tiny fraction of the total LLM pre-training budget, recent systems have demonstrated remarkable performance across vision–language and video–language tasks [92; 93; 94; 95; 96; 97; 84]. Research has also produced MLLMs for a wide range of less conventional modalities, including audio, robotics data, specialized time series, and even graphs. Public-facing systems such as Gemini, GPT-4o, and ChatGPT now offer robust multimodal capabilities, enabling real-time reasoning over images, audio, and other complex inputs. This convergence of general-purpose language reasoning with flexible modality integration has positioned MLLMs as one of the central building blocks of next-generation AI systems.

3 POOLING IMAGE DATASETS WITH MULTIPLE COVARIATE SHIFT AND IMBALANCE

Sample sizes of medical imaging datasets at a single institution are often small due to many reasons. Acquiring thousands of images can be infeasible due to budget/logistics. Also, if the scope of a study is narrow, only some individuals may be eligible due to inclusion criteria (e.g., have a specific genetic risk). To improve the statistical signal in retrospective analyses of existing datasets, one option is to pool similar data across multiple sites [98]. This offers a chance at discovering a real statistical effect, undetectable with small sample sizes.

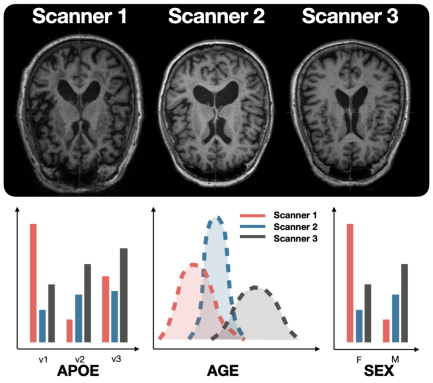


Figure 3.1: **Problem overview:** When pooling image datasets from different sources, there may be differences in the distribution of covariates. Covariates are secondary data for each individual that influences the images systematically. Top row shows MR images (say, different scanners). Second row shows how the covariate distributions (genetic risk, age or gender) varies across scanners (but have a shared support). Learning representations from a pooled dataset in a manner such that covariate variations are accounted for, is challenging.

Pooling data from multiple sites is common. A mature body of statistical literature (e.g., covariate matching [99; 100], meta-analysis [98; 101]) describes best practices, and mechanisms to account for the effect of a nuisance variable on the response (or target label) are well known. This allows obtaining associations between relevant predictors and the response/dependent variable of interest. For example, a standard analysis workflow may use *only* the images to predict a label (e.g., cognition) while controlling for nuisance variables such as race, gender or scanner type [102].

Controlling for covariates. DL models are now ubiquitous in medical image

analysis [103; 104]. But when using such models, systematic mechanisms to deal with nuisance variables (or covariates), a key concern in data pooling, are still under development. Consider pooling MR images from two different sites where participant demographics distributions (e.g., age, sex etc) across sites are similar. If the scanners at the sites are different, we can include "scanner" or other continuous variables as a nuisance, and control for it within a general linear model. Features in the image X deemed informative will explain variability in the response Y *after* the variability due to "age" and "site" has been accounted for. In other words, the image data (and not the site-specific artifacts) should predict the response variable. Harmonization/pooling of imaging data in a way that accounts for (or removes) the influence of non-imaging covariates is often limited to shallow pre-processing using additive/multiplicative batch corrections, e.g., in Combat [105; 106].

Data pooling/harmonization and invariance. In the context of representation learning, several results have identified the link between data pooling/harmonization and invariant representation learning or domain adaptation [107; 108], which have been utilized for downstream tasks [109]. Initial approaches focused on image normalization techniques (e.g., histogram matching [110]), which allowed controlling some variations in intensities across scanners or protocols. However, such approaches cannot systematically handle covariates. Ideas based on invariant representations [111; 112; 113] have allowed controlling for up to two covariates during representation learning. For example, one may ask that the model avoid using image features associated with two specific covariates (or sensitive attributes): age and site. Nonetheless, models that can easily remove (or control for) the influence of multiple covariates remain limited and so, either Combat-based [105] pre-processing is used [106] or CycleGAN-based schemes [114; 115] can harmonize the image data relative to a specific categorical variable (say, scanner). This problem is also relevant in the longitudinal setting [116].

The problem. Different scanners can introduce systematic differences in the scans of the same person [117] and mitigation strategies continue to be a topic of

recent work [118]. When there is some shift/disbalance in covariates (participant data including age, sex, and so on, which influence the scans) across sites and shared support is partial ("age" distribution in Fig. 3.1), the harmonization task becomes more involved. The goal is to learn representations as if the covariates were matched across the sites to begin with, see Fig. 3.1. Note that in contrast to covariate shift methods [119], the shift here is not between train and test distributions, rather between different sources of data in the training set itself.

One recent attempt in [120] to learn deep representations from pooled data which can also handle (shift+imbalance) in covariates needs a two stage pipeline. The first stage obtains latent representations of the scan which is *equivariant* to age. The second stage enforces *invariance* to scanner and trained after freezing the first stage. Two covariates can be handled but an extension to *many* continuous (and categorical) covariates will need a complicated multi-stage model.

Main Ideas. In most learning tasks involving an equivariance or invariance criteria, the goal is to allow the model to benefit from structure and symmetry – in the data, the parameter space, or both. Frequently, one formalizes these ideas using group theory seeking invariance/equivariance to the action of the group (e.g., $\mathbb{SO}(n)$, $\mathbb{SE}(n)$, $\mathbb{S}_n$) [111; 121]. More general criteria (beyond properties native to the chosen group) requires specialized treatment. Our **starting point** is based on the observation that Category theory (Section 2.2) provides a rich set of tools by which the "structure" (either among the covariates or the data more generally) can be expressed easily. It is known that equivariance, invariance and many group-theoretic constructs will emerge as special cases because Category Theory provides a more abstract treatment. Once our criteria are expressed in this way, during training, the necessary constraints on the latent space fall out directly, and the formulation gracefully handles heterogeneity in many continuous/categorical covariates. No ad-hoc adjustments are needed.

Contributions. (a) On the **technical** side, we provide a general framework for imposing structure on the latent space by applying ideas from Category theory. Equivariant (and invariant) representation learning emerge as special cases.

Further, this style of formalism unifies different formulations in vision and machine learning, under the same umbrella. **(b)** On the **practical** side, we strictly generalize existing formulations to pool/harmonize multi-site datasets. While the existing two-stage formulation can deal with one categorical and one continuous covariate, our formulation places no restriction at all on the number of covariates. We show how the same model can also be used to heuristically approach certain hypothetical "what if" questions and offers competitive performance on public brain imaging datasets, with strictly more functionality/flexibility.

3.1 Enforcing Symmetry and Structure

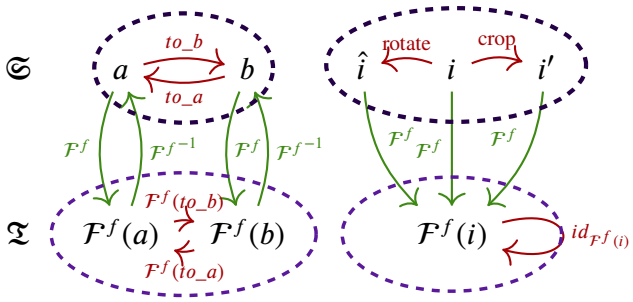


Figure 3.2: A Category theoretic view of CycleGAN (left) and SimCLR (right)

Symmetry/structure in the data or parameters are key property we exploit in learning tasks, e.g., invariance in features [122], compressive sensing [123] as well as DL training [124; 125]. Encoding symmetry/structure can involve ideas as simple as rotations and cropping for data augmentation [126; 127; 128] to more sophisticated concepts [129; 130]. This criteria can also be specified on the latent space, e.g., via geometric/topological priors [131; 132] or other conditions [133; 134]. From Fig. 3.1, if there is a natural structure (say, in covariate space), then the structure of the representations learned on the latent space should respect that. To setup this description, we first discuss how this perspective can relate distinct ideas under an abstract (but simple) formalism, using Categorical concepts (Section 2.2).

3.1.1 Reinterpreting CycleGAN using Category theory

Image translation uses a set of image pairs and trains a model to map between them. CycleGAN [135] is a popular framework for image translation that removes the need for paired image pairs in the training data, and learns the map from one domain (or class) to the other, similar in nature to Example 2.5. Denoting image domains as *type a* and *type b*, the goal in CycleGAN is to learn how to map a *type a* image to a *type b* image and back; which we can denote as *to_b* and *to_a*: type *a* (and type *b*) can mean images of horses (and zebras). We will avoid defining a Category for the distributions over the images for simplicity, and so a training dataset made up of paired images will suffice to convey the key idea. So, for every image of type *a*, there is a similar image of type *b*. We want to learn a mapping (Functor) to a latent space that preserves the action of change from type *a* to *b* and back. This is shown in Fig. 3.2 (left), from which we can directly read off the following constraints,

1. $\mathcal{F}$ is a fully-faithful Functor (Definition 2.3) meaning:

$$\exists \mathcal{F}^{f^{-1}} : \mathfrak{T} \rightarrow \mathfrak{S} \text{ such that } \mathcal{F}^{f^{-1}} \circ \mathcal{F}^{f}(s) = id_s \quad \forall s \in \mathfrak{S}$$

2. Composition:

$$\text{to\_b} \circ \text{to\_a}(b) = id_b(b) = b$$
$$\text{to\_a} \circ \text{to\_b}(a) = id_a(a) = a$$

The first pair of constraints is implicit in autoencoder-like models. The second pair of constraints define the cycle consistency conditions in CycleGAN. We note that a different formalism for CycleGAN was described in [60] by considering CycleGAN as a schema where cycle-consistencies enforce composition invariants in that Category.

3.1.2 Reinterpreting SimCLR using Category theory

SimCLR [128] is a *self-supervised learning* method where one attempts to map the images to a latent space which is invariant to transformations such as rotations, croppings, and so on. This can be instantiated using an invariant Functor (see Fig. 3.2). Unlike CycleGAN, such an invariant Functor $\mathcal{F}^f$, is no longer fully-faithful (i.e., $\nexists \mathcal{F}^{f^{-1}}$ such that $\mathcal{F}^f \circ \mathcal{F}^{f^{-1}} = id_x$).

3.1.3 Reinterpreting other formulations using Category theory

(a) Latent space interpolation. A number of approaches seek to perform interpolation in latent space [133; 134]), informed by "covariates" associated with the original images (e.g., age, sex, hair length). The recipe described in the two examples above lends itself directly to describing such formulations, and will be subsumed by our formulation in the next section. **(b) Rotation and Permutation synchronization.** Rotation synchronization is a central problem in Cryo-EM image reconstruction [136; 137; 138]. In computer vision, permutation synchronization [139] seeks to find replicable matches of keypoints across many images of the same 3D scene, e.g., for 3D reconstruction from datasets such as PhotoTourism [140; 141; 142] as well as in robotics [143]. Since the synchronization task always involves consistency constraints over the symmetric Group, and Groups are a Category, the reinterpretation is immediate. **(c) Equivariance and Invariance.** Many problems in learning benefit from equivariance and invariance. Recall that equivariance is formally defined in terms of an action of a Group G.

Definition 3.1 (Equivariance)**.** *A mapping $\mathcal{F} : \mathfrak{S} \to \mathfrak{T}$ defined over measurable Borel spaces $\mathfrak{S}$ and $\mathfrak{T}$ is said to be $\mathbb{G}$-equivariant under the action of group $\mathbb{G}$ iff*

$$\mathcal{F}(\mathcal{G} \cdot s) = \mathcal{G} \cdot \mathcal{F}(s), \quad \mathcal{G} \in \mathbb{G} \tag{3.1}$$

A more general definition of equivariance follows directly from the Functor's definition, as defined in Definition 2.3 an recalled here:

Definition 3.2. *A Functor* $\mathcal{F}^f : \mathfrak{S} \to \mathfrak{T}$ *is defined as a mapping from* $\mathfrak{S}$ *to* $\mathfrak{T}$ *such that*

$$\mathcal{F}^f(id_s) = id_{\mathcal{F}(s)} \quad \forall s \in \mathfrak{S} \tag{3.2}$$

$$\mathcal{F}^f\big(\mathcal{G}(s_1)\big) = \mathcal{F}^f(\mathcal{G})\big(\mathcal{F}^f(s_1)\big), \quad \forall \mathcal{G} : s_1 \to s_2 \in \mathfrak{S} \tag{3.3}$$

If $\mathfrak{S}, \mathfrak{T}$ are Borel spaces, and $\mathcal{G}, \mathcal{F}^f(\mathcal{G})$ belong to a Group $\mathbb{G}$, then we obtain the group-theoretic equivariance definition. Category theory gives a more general result with no restrictions on the form of the relationships (or Morphisms) $\mathcal{G}$. Similarly, invariance (e.g., in Group theory) is defined as

Definition 3.3. *A mapping* $\mathcal{F} : \mathfrak{S} \to \mathfrak{T}$ *defined over measurable Borel spaces* $\mathfrak{S}$ *and* $\mathfrak{T}$ *is said to be* $\mathbb{G}$*-invariant under the action of group* $\mathbb{G}$ *iff*

$$\mathcal{F}(\mathcal{G} \cdot s) = \mathcal{F}(s), \quad \mathcal{G} \in \mathbb{G} \tag{3.4}$$

In Category theory, this results in a special type of Functor:

Definition 3.4 (Invariance)**.** *A Functor* $\mathcal{F}^f : \mathfrak{S} \to \mathfrak{T}$ *is invariant to the Morphism* $\mathcal{G} : s_1 \to s_2$ *if* $\mathcal{F}^f(\mathcal{G}) = id_{\mathcal{F}^f(s_1)}$.

3.1.4 A simple sanity check of benefits on MNIST Dataset

Consider the case where the Objects of the source Category $\mathfrak{S}$ consist of MNIST images [144]. Since these images represent integers on the number line, we can ask whether we can learn a latent space which allows algebraic manipulations, with one (or more) basic operations defined on it. If our operation of interest was "addition", a primitive we will need for counting would be the "+1" operation. To keep things simple, we will focus on this operation – where the

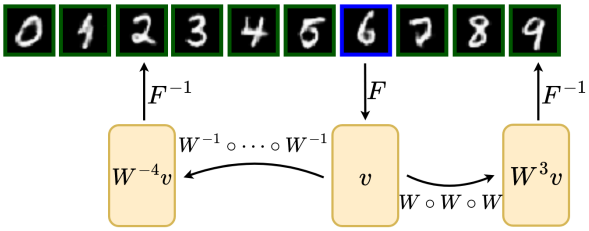


Figure 3.3: **MNIST Example:** Modelling the relationships between the digits as linear mappings in the target Category.

source Category $\mathfrak{S}$ consists of Morphisms that represent the "+1" difference between two subsequent digits. It is easy to check that we have indeed defined a Category since each Object has an identity Morphism and the Morphisms compose. The target Category $\mathfrak{T}$ (i.e., our latent space) can consist of vectors $\mathbf{v} \in \mathbb{R}^N$ as Objects. Orthogonal linear mappings $\mathbf{W} \in \mathbb{R}^{N \times N}$ as Morphisms will suffice. In our pooling/harmonization problem, the Morphisms will reflect traversing the axis of each covariate.

For this MNIST example, our goal is to impose the "+1" operation in the latent space, see Fig. 3.3. We define our loss to express precisely the equivalence for subsequent digits as well as a subset of their +1 compositions that can be read off of Fig. 3.3. The model is *not* presented data for "−1" operations but discovers it due to the special structure of $\mathbf{W}$ (orthogonal, so $\mathbf{W}^{-1}$ exists). Then, after training, when presented a "seed" image of 6, the model can (forward/backward) traverse the latent space: a query "$6 - 4$" starts from the latent code for 6, hops backwards 4 times and generates an image of 2. All images except "6" shown in Fig. 3.3 were generated in this way. It is worth making a brief comment on the generality here. If we had k different morphisms, say, "+1" as well as rotation, scaling and shear, we could apply them in sequence and generate the corresponding images, even if such training data were not shown to the model (Figure 3.4). This feature will allow dealing with multiple covariates easily in our experiments. A similar solution has been provided in [145], although their experiments were restricted to simple datasets and dealt only with 2D object dispositions. Here, we already showed how to learn something more complicated (addition) and, in the next section, we will show how to adapt this idea to complicated covariates.

3.2 A Category Theory inspired Formulation for Data pooling

Overview. The preceding section provides us all the necessary modules. Our task involves using brain MR images to predict diseased or healthy controls

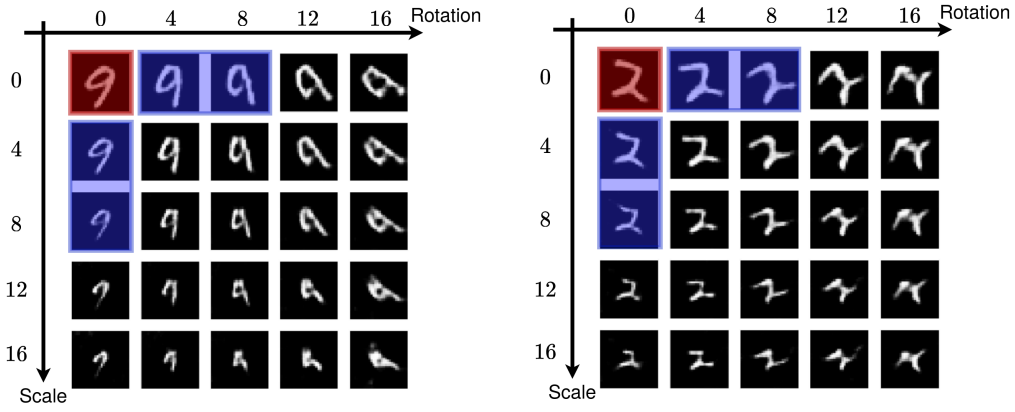


Figure 3.4: Composition of rotation and scaling. **The transformations were applied to the original image's latent vector (v) in the latent space** according to the composition rule $\mathbf{W}_r^i \mathbf{W}_s^j \mathbf{v}$ where i, j denote the degree of rotation and scaling respectively. In red we depict the original image, in blue we depict the only transformations that were "visible" during training.

status. Our covariates will include secondary (non-imaging) data pertaining to the participants, including scanner type, site, age, sex, and genotype status. Some of these are ordinal/continuous such as genotype (APOE; three risk types) and age, whereas others are categorical such as scanner type, site and sex. For the categorical covariates, our formulation will seek to enforce *invariance* on the learned representations, i.e., asking that the latent representations be devoid of information pertinent to nuisance variables like scanner and site. It turns out that sex is associated with Alzheimer's disease (AD) because two-thirds of those diagnosed are women [146]. But if our goal is to understand which image features are relevant for the disease (and not simply to maximize accuracy), it makes sense to control for sex as a nuisance variable and add it separately at the last layer, if needed. For ordinal/continuous covariates, we see both shifts and imbalances across sites or scanner (different sites/scanners may not cover exactly the same age range of participants or include the same number of individuals for each genotype value). Equivariance will allow adjusting for these shifts, i.e., a coordinate system

where the latent representations are "aligned" (modulo their covariate induced Morphism). The diagram in Fig. 3.5 will setup our overall formulation, and we use the example below to describe its operation.

Example 3.5. *Assume we want to control only for age when learning the latent representations of the images for objects/participants s_1 and s_2. s_1, s_2 are identical for all covariates but have different ages. Their scans are also different. If s_2 is x years older than s_1, in $\mathfrak{S}$, we have $s_2 \cong \mathcal{H}_x(s_1)$ (i.e., equal up to isomorphism. In practice, isomorphism can be as strict as ℓ_2-norm, cosine similarity as in CLIP-based models [147], or even distribution-based measures such as MMD). For someone, $2x$ years older, similar to Fig. 3.3, we will compose $\mathcal{H}_x$ twice. In $\mathfrak{T}$, latent representations t_1, t_2 correspond to $\mathcal{F}^f(s_1)$ and $\mathcal{F}^f(s_2)$. We seek to learn a Functor $\mathcal{F}^f$ by learning $\mathcal{G}_x \equiv \mathcal{F}^f(\mathcal{H}_x)$ such that $t_2 \cong \mathcal{G}_x(t_1)$. If s_1 and s_2 differed in two covariates, by amounts x and y resp., the Morphism from s_1 to s_2 would involve composing $\mathcal{H}_x$ and $\mathcal{H}_y$ (assuming $\mathcal{H}_x, \mathcal{H}_y$ denoted the Morphisms for the covariates x, y respectively). The Morphisms in the Category $\mathcal{T}$ would compose similarly.*

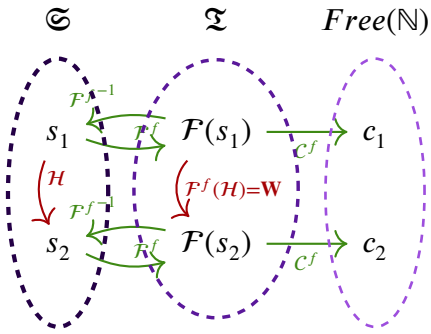


Figure 3.5: A diagrammatic representation of equivariance with respect to a single covariate whose change corresponds to "$\mathcal{H}$" in the original data space. Our goal is to preserve this structure in the latent space (Category $\mathfrak{T}$) in which we model $\mathcal{F}^f(\mathcal{H})$ as a linear transformation $\mathbf{W} \in \mathbb{R}^{N \times N}$. In many practical cases the downstream goal is to classify the latent representations, which we formulate with the Functor $\mathcal{C}^f : \mathfrak{T} \to Free(\mathbb{N})$ (or $Free(\mathbb{R})$ for a regression task, etc.)

Setting up a loss function. We will use simple Morphisms in $\mathfrak{T}$ which correspond to linear transformations $\mathbf{W} \in \mathbb{R}^{N \times N}$ (this is a design choice). The

pair of Functors $(\mathcal{F}^f, \mathcal{F}^{f^{-1}})$ correspond to an autoencoder, while the pair $(\mathcal{F}^f, \mathcal{C}^f)$ corresponds to an appropriate classifier (or regressor). Simply walking the paths from Fig. 3.5 provides all necessary constraints which we write as distinct terms in our loss function,

(i) $\left(\mathcal{F}^{f^{-1}} \circ \mathcal{F}^f\right)(x) = id_x$ gives the reconstruction loss:

$$\mathcal{L}_r = \sum_{s \in \mathfrak{S}} \left\| s - \left(\mathcal{F}^{f^{-1}} \circ \mathcal{F}^f\right)(s) \right\|_2^2 \tag{3.5}$$

(ii) the prediction loss (given labels $\mathbf{y}$):

$$\mathcal{L}_p = \sum_{(i,s) \in \mathfrak{S}} \underbrace{\mathcal{CE}\left(\mathbf{y}_i, \left(\mathcal{C}^f \circ \mathcal{F}^f\right)(s)\right)}_{\text{cross-entropy}} \tag{3.6}$$

(iii) Finally, preserving the Morphisms, in its simplest form, is a structure-preserving loss,

$$\mathcal{L}_s = \sum_{s_1, s_2 \in \mathfrak{S}} \left\| \mathbf{W} \cdot \mathcal{F}^f(s_1) - \mathcal{F}^f(s_2) \right\|_2^2 \tag{3.7}$$

The training for multiple covariates (Alg. 1) is a direct generalization of the above formulation.

Enriching the latent space with structure provides some information about "hypotheticals". Given that we are unlikely to be provided different versions of a sample (one for each composition of the known Morphisms), a structure-preserving latent space gives us a way to:

(i) generate new samples, by evaluating the expression $\left(\mathcal{F}^{f^{-1}} \circ \mathcal{F}^f(\mathcal{G}) \circ \mathcal{F}^f\right)(s)$.

(ii) answer questions, by evaluating the expression $\left(\mathcal{C}^f \circ \mathcal{F}^f(\mathcal{G}) \circ \mathcal{F}\right)(s)$.

for a given sample $s \in \mathfrak{S}$.

Role of Category Theory. There is a clear advantage of expressing the problem in Category Theory. It enables a more concise and straightforward representation of the problem (sometimes, apparent in hindsight), as well as additional capabilities compared to earlier works [120].

Algorithm 1 Structure preserving training of Functors

Input: Parameters: $(\theta_1, \theta_2, \theta_3, \{\mathbf{W}_i\}_{i=1}^{C})$, Multipliers: $\lambda_1, \lambda_2, \lambda_3$
Data: input/output: $(\mathbf{S}, \mathbf{y}) \in (\mathbb{R}^{M \times N}, \mathbb{N}^{M})$, and covariates: $\mathbf{C} \in \mathbb{N}^{M \times C}$

for *ep in epochs* **do**
  $\mathcal{L}_r = \sum_{i=1}^{M} \left\| [\mathbf{S}]_i - \left({\mathcal{F}^f}_{\theta_1}^{-1} \circ \mathcal{F}^f_{\theta_2} \right)([\mathbf{S}]_i) \right\|_2^2$,
  $\mathcal{L}_p = \sum_{i=1}^{M} \mathcal{CE}\left([\mathbf{y}]_i, \left(\mathcal{C}^f_{\theta_3} \circ \mathcal{F}^f_{\theta_1} \right)([\mathbf{S}]_i)\right)$,
  $\mathcal{L}_s = 0$
  for $\mathbf{c} \in C$ **do**
    for $(s_1, s_2), d$ *in pairs(*$\mathbf{S}, \mathbf{c}$*)* **do**
      $\mathcal{L}_s \mathrel{+}= \left\| \mathbf{W}^d_{\mathbf{c}} \cdot \mathcal{F}^f_{\theta_1}(s_1) - \mathcal{F}^f_{\theta_1}(s_2) \right\|_2^2$
  $\mathcal{L} = \lambda_1 \mathcal{L}_r + \lambda_2 \mathcal{L}_p + \lambda_3 \mathcal{L}_s$
  step$\left(\mathcal{L}, (\theta_1, \theta_2, \theta_3, \mathbf{W}_1, ..., \mathbf{W}_c)\right)$ `// update`
Function `pairs(`$\mathbf{S}$, $\mathbf{c}$`):`
  pairs = []
  for *(i, j) in* $M \times M$ **do**
    if $[\mathbf{S}]_i$ *is paired with* $[\mathbf{S}]_j$ **then**
      pairs.append$\left(([\mathbf{S}]_i, [\mathbf{S}]_j), [\mathbf{c}]_i - [\mathbf{c}]_j\right)$
  return *pairs*

3.3 Experimental evaluations

Rationale. Our motivation stems from pooling brain imaging datasets, particularly in scenarios where covariate distributions vary across sites. In the ADNI brain imaging study [148], while acquisition protocols are consistent among 50+ sites, demographic distributions differ. The diversity of scanners across sites, along with the impact of scanner upgrades on analysis tasks [149; 150; 151], further complicates the scenario. Scanner variations, even within the same study (e.g., ADNI-3), introduce controlled nuisances for standard analyses. Beyond ADNI, we also perform analysis on the ADCP dataset [120]. Our objective is to incorporate this functionality into representation learning for regression/classification.

Datasets The ADNI dataset can be obtained from `https://adni.loni.usc.`

edu/. The ADNI project was launched in 2004. This specific dataset consists of 449 MRI scans, along with a dataset of covariates for each individual participant. Each MRI was obtained in one of three different scanners ({GE Medical Systems, Philips Medical Systems, SIEMENS}) and some of the covariates include {Age, Sex, Years of Education, APOE A1, APOE A2, Marital status, Race} along with a lot of cognitive battery scores such as {MMSE, RAVLT, Ecog}. In neuroimaging data, "APOE" typically refers to the apolipoprotein E gene. The APOE gene is involved in encoding a protein that plays a crucial role in the metabolism of lipids (fats) in the body, including cholesterol. This gene has different variants, or alleles, known as APOE $\epsilon 2$, APOE $\epsilon 3$, and APOE $\epsilon 4$. In the context of neuroimaging and neuroscience, the APOE gene has been of particular interest because one of its alleles, APOE $\epsilon 4$, is considered a significant genetic risk factor for Alzheimer's disease [152; 153]. The second dataset (ADCP) was shared with us by the authors of [120]. The data for ADCP was collected through an NIH-sponsored Alzheimer's Disease Connectome Project (ADCP) U01 AG051216. The study inclusion criteria for AD (Alzheimer's disease) / MCI (Mild Cognitive Impairment) patients consisted of age between 55-90 years, willing and able to undergo all procedures, retaining decisional capacity at the initial visit, and meet criteria for probable AD or MCI. The MRI images were acquired at three distinct sites.

This problem setting underscores limitations in existing methods. Recent VAE or GAN-based approaches [154; 155; 108; 156] target scanner invariance, often accommodating binary or multiple scanner variables. However, handling variations in other covariates (e.g., age, sex, APOE) remains challenging [152].

Baselines. The following baselines address distributional differences in the data while maintaining associations with covariates: **1.** *Naive*: Pooling data without pre/post processing. **2.** *MMD* [112]: Minimizes Maximum Mean Discrepancy (MMD) for invariance but lacks equivariance. **3.** *CAI* [157]: Achieves invariance with a discriminator but lacks equivariant mappings. **4.** *SS* [158]: Divides the population into subgroups (Subsampling-SS) and minimizes MMD for

each subgroup. **5.** *RM* [159]: Matches similar samples from different scanners for invariance. **6.** *GE* (Group Equivariance) [120]: Uses group theory to minimize MMD while seeking equivariance with respect to one covariate. GE achieves the best overall accuracy but has limitations: it uses expensive matrix exponentials and relies on a two-stage training, handling only two covariates.

Evaluations. As the classification task, we will predict Alzheimer's disease (AD)/Control normals (CN) labels using the invariant and/or equivariant representations. We will use the following metrics to systematically assess each algorithm. **(a)** Accuracy ($\mathcal{ACC}$): Test set accuracy of whether or not a participant has AD. Accuracy is reported in order to ensure that the model, despite its extra constraints, maintains all the required information that an input image carries. **(b)** $\mathcal{MMD}$: MMD is a measure of distributional differences [160], defined as the Euclidean distance of the kernel embeddings means (typically an RBF kernel). **(c)** $\mathcal{ADV}$: an alternative to measure invariance is by training a model to predict the nuisance covariate (e.g., "Scanner") using the latent representations. If we obtain latent representations devoid of this information, then the accuracy should be almost random. The above two metrics identify invariance with respect to scanners. To check equivariance to covariates, we use two measures: Minimum distance ($\mathcal{D}$) and Cosine similarity ($\mathcal{CS}$),

(a) Assume that a value of c_1 (for a covariate $\mathbf{c}$) corresponds to an individual s_1, and Morphism $\mathbf{W} \in \mathbb{R}^{N \times N}$ in the latent space gives a change in the covariate from c_1 to c_2. Then, the Minimum Distance (similar to covariate matching) identifies the closest individual s such that $s_{\mathbf{c}} = c_2$:

$$\mathcal{D}(s_1; c_2) = \min_{s : s_{\mathbf{c}} = c_2} \frac{\left\| \mathbf{W} \cdot \mathcal{F}^f(s_1) - \mathcal{F}^f(s) \right\|_2}{N} \tag{3.8}$$

(b) A low minimum distance by itself is insufficient. So, we also calculate the cosine similarity ($\mathcal{CS}$) between all the other covariates of s_1 and s_2, where $s_2 = \arg\min_{s : s_{\mathbf{c}} = c_2} \left\| \mathbf{W} \cdot \mathcal{F}^f(s_1) - \mathcal{F}^f(s) \right\|_2$.
Low Minimum Distance ($\mathcal{D}$) and high Cosine Similarity ($\mathcal{CS}$) is desirable.

Setting. We model the Functor $\mathcal{F}^f$ using a modified ResNet [161], and the Functor $\mathcal{C}^f$ using a Fully-Connected Neural network. For our experiments, using linear mappings $\mathbf{W} \in \mathbb{R}^{N \times N}$ for the Morphisms in the target Category $\mathfrak{T}$ (i.e., latent space) was sufficient but this can be easily upgraded. All the experiments are a result of 5-fold cross validation procedure. In the following sub-sections, we summarize our experimental findings. We gradually increase the complexity of the experiments (number of constraints that we simultaneously optimize).

Can Category theory constraints help impose invariance to scanner? The following objective derived directly from the formulation is used,

$$\mathcal{L} = \underbrace{\mathcal{L}_p}_{\text{cross-entropy}} + \sum_{s_1, s_2 \in \mathfrak{S}} \underbrace{\lambda \cdot \left\| \mathcal{F}^f(s_1) - \mathcal{F}^f(s_2) \right\|}_{\forall\ s_1, s_2 \text{ from different Scanners}} \tag{3.9}$$

Our results (Table 3.2) show that we obtain invariant representations without degrading the model's accuracy, consistent with the literature on invariant representation learning [107]. MMD decreases by more than 80% compared to the best baseline, while accuracy is 1.3% higher than the naive model, suggesting stronger generalization to new samples.

Can a model be invariant to scanner and remain equivariant to covariates? Besides invariance, we seek equivariance to specific covariates in some cases. Here, we examine the simplest case of equivariance to a single covariate. We test three different covariates: 1. Age 2. Sex 3. APOE A1. Since age is continuous, we discretize it into bins of 10 years (so we have enough samples in each bin). APOE A1 (and A2) are genotypes in ADNI that are associated with AD [152; 153]. Using our formulation, we derive the following loss function,

$$\mathcal{L} = \underbrace{\mathcal{L}_p}_{\text{cross-entropy}} + \sum_{s_1, s_2 \in \mathcal{S}} \underbrace{\lambda \left\| \mathbf{W}^{c_{s_1} - c_{s_2}} \cdot \mathcal{F}^f(s_1) - \mathcal{F}^f(s_2) \right\|}_{c_{s_1}, c_{s_2} \text{ represent the nuisance covariate}} \tag{3.10}$$

Note that the invariance term is implicit in this case because for two individuals s_1, s_2 with $c_{s_1} = c_{s_2}$ we recover the invariance term from (3.9).

	Parameters			Runtime
	Encoder	Classifier	Morphisms	
GE	$\mathcal{O}(E)$	$\mathcal{O}(C)$	$\mathcal{O}(c \cdot P)$	$\mathcal{O}(c \cdot R)$
Ours	$\mathcal{O}(E)$	$\mathcal{O}(C)$	$\mathcal{O}(c)$	$\mathcal{O}(R)$

Table 3.1: **Parameters / Runtime comparison with respect to the number of covariates.** c is the number of covariates and P the number of unique pairs for each covariate (exponential growth). E/C is the number of parameters in the Encoder and Classifier respectively (independent of c). R stands for a single training runtime. Our method's complexity is better both in parameters and runtime.

Remark 3.6. *While this specific problem setting is not new (see [159]), here we show that the simplicity of our method comes with certain* ***advantages****. First, we get improved results and further, our model has fewer parameters and smaller runtime complexity (Tables 3.2, 3.1).*

In Fig. 3.6, we examine the performance of the naive model where we infer $\mathbf{W}$ post-training, GE [120] which defines a different linear transformation $\mathbf{W}$ for each age difference, and our model (Ours). In all three models, we model the age increase (e.g., 60 to 65) by applying a linear transformation in the latent space. Then, we find the closest point for that age (e.g., 65) in the latent space w.r.t. ℓ_2 norm ($\mathcal{D}$) and cosine similarity ($\mathcal{CS}$). The transformed vector should be in the neighborhood of latent vectors with that age (e.g., 65), and so $\mathcal{D}$ should be small and $\mathcal{CS}$ should be high. The results show that such a Morphism is not accurate enough in the naive or even the GE algorithm. The naive model tends to "spread" the latent vectors without preserving structure, so it cannot capture equivariance in the latent space (clear when we examine $\mathcal{D}$), while GE achieves equivariance but our results suggest improvements.

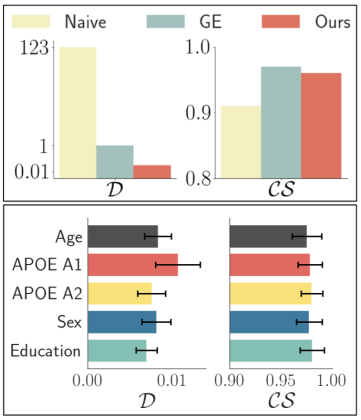


Figure 3.6: (left) Minimum Distance ($\mathcal{D}$) and Cosine Similarity ($\mathcal{CS}$) in for age equivariance, compared with Naive and GE. (right) Minimum Distance ($\mathcal{D}$) and Cosine Similarity ($\mathcal{CS}$) for 5-covariate equivariance. Results are consistently good.

Assessing hypothetical samples. Enriching the latent space with structure

	ADNI			ADCP		
	$\mathcal{ACC}\uparrow$	$\mathcal{MMD}\downarrow$	$\mathcal{ADV}\downarrow$	$\mathcal{ACC}\uparrow$	$\mathcal{MMD}\downarrow$	$\mathcal{ADV}\downarrow$
Random	64	-	49	74	-	42
Naive	80(2.6)	27(1.6)	59(2.9)	83(4.4)	90(08.7)	49(08.4)
MMD[112]	80(2.6)	27(1.8)	59(3.3)	84(6.5)	86(11.0)	49(11.9)
CAI[157]	74(3.6)	27(1.5)	61(2.1)	82(5.1)	85(12.3)	56(06.9)
SS[158]	81(3.7)	26(1.6)	57(2.1)	82(3.5)	88(14.6)	51(06.7)
RM[159]	78(3.8)	22(0.6)	52(5.4)	84(5.3)	77(13.8)	**40**(04.7)
GE[120]	77(4.8)	16(7.2)	**50**(4.2)	81(1.8)	70(22.3)	49(07.3)
Ours (inv)	81(2.3)	**02**(2.4)	52(2.0)	**86**(8.0)	**34**(01.2)	45(04.2)
Ours (1 cov)	**82**(2.5)	**11**(2.9)	53(1.2)	**86**(5.9)	**39**(03.6)	48(05.7)
Ours (2 cov)	**82**(2.2)	**10**(6.6)	51(3.0)	**85**(7.7)	**40**(04.3)	44(08.1)
Ours (5 cov)	80(2.3)	**11**(5.3)	52(1.7)	–	–	–

Table 3.2: **Quantitative Results on ADNI [148] and ADCP from [120].** The mean accuracy ($\mathcal{ACC}$) and invariance as evaluated by $\mathcal{MMD}$ and $\mathcal{ADV}$ are shown. The standard deviations are in parenthesis. The baseline *inv* corresponds to only invariance to Scanner, *x cov* corresponds to equivariance with respect to *x* covariates (along with invariance to Scanner). $\mathcal{MMD}$ measure is significantly reduced without any drop in accuracy. The $\mathcal{ADV}$ measure is close to the random baseline (desirable).

provides information about questions such as *What would be the change in AD if APOE A1 had a different value?* Recall that APOE A1 is a ordinal covariate with 3 values ($\{2, 3, 4\}$) and associated with AD [152; 153]. For each sub-group (individuals with a specific value for APOE A1), we increase/decrease its value using Morphisms and then classify the new vector. An increase in APOE A1 results in a higher probability of AD and vice-versa (Fig. 3.7), as expected [152].

3.4 Related work

Data pooling, Fairness, disentanglement, and Invariant Representation Learning. General approaches for analyzing data from different sites involve meta analysis [98; 101]. When data transfer is feasible, pooling can be approached using the Johnson-Neyman technique (e.g., when ANCOVA is inapplicable). Some tools from statistical genomics have been deployed for brain image data pooling [105], also see [162]. For deep models, the link between invariant representation

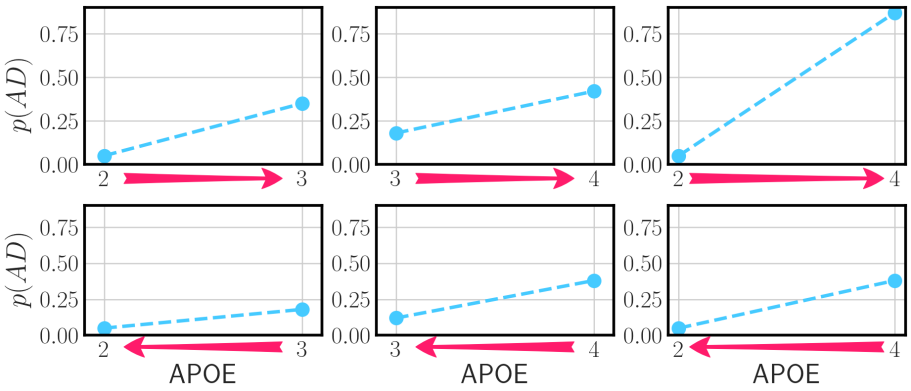


Figure 3.7: Increasing APOE A1 in the latent space using Morphisms leads to a higher probability of AD and vice versa. Such manipulations in latent space are feasible through learned Morphisms.

learning/fairness and data pooling was seen in [107; 163; 164], and has been the defacto approach [108; 120; 165; 166; 167] to disentangle the influence of nuisance variables (but provably doing so is difficult [168]).

Category theory in machine learning. The applications of Category theory in applied disciplines are somewhat limited. But more recently, these ideas have been successfully applied to other fields (e.g. [169]). In machine learning specifically, [170; 61; 171; 172] model the learning process of gradient-based learning algorithms using Lenses and [173] uses Monads. Separately, [174] used Category theory to offer a framework for training Boolean circuits. The results in [60] modeled CycleGAN using Functors and showed how the formulation can be used for inserting/deleting objects from an image. A summary of developments is in [49].

3.5 Summary

Imposing structure on latent spaces learned by DNN models is being actively studied, for problems ranging from disentanglement to interpretability. Using a data pooling problem in brain imaging as a motivation, we discuss how Category

Theory provides a precise framework for such tasks. Not only does such an approach significantly simplify recent works in the literature, but also offers a perspective unifying an array of standalone approaches/models. Due to its rather abstract nature, the application of Category Theory in vision/machine learning is rather limited. Nonetheless, we believe that the models/experiments provide evidence that these ideas can inform other challenging problems in our field, including explainability, modularity and interpretability. A preliminary version of this chapter was published as [62] and the implementation can be found in `https://github.com/SPChytas/CatHarm`.

4 UNDERSTANDING MULTI-COMPOSITIONAL LEARNING IN VISION AND LANGUAGE MODELS VIA CATEGORY THEORY

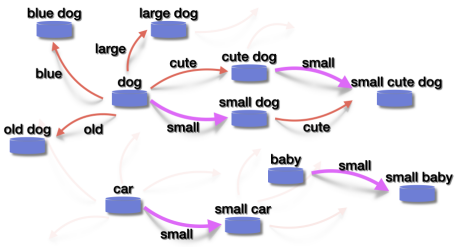


Figure 4.1: The implicit structure of concepts should be reflected in a model too. Arrows "add" new information to concepts and are shared among different concepts.

In the theory of Forms, Plato argues that every entity in the *physical* world – animals, persons, and all other entities – correspond to a Form (or Idea) that answers "What is that?" [175]. In fact, the objects in our world are merely *imitations* of these "Forms", their impure realizations. The Idea on the other hand refers to the *non-physical* essence of all things. For instance, we can only observe the Idea "car", via an imitation of its *attributed* Form, *red car*, *small car*, *sports car*, and so on. We are almost never shown the Idea, by itself. Still, we inherently possess the ability to perform such compositional classification, and more importantly, recognize novel combinations of attributes and Ideas.

Since we observe the world in this compositional view of attributes and Ideas, it is reasonable to ask whether a model can be trained to perform such compositional learning or whether a model already possesses this ability. The reader will acknowledge that despite intensive (and impressive) ongoing work in foundation and generative models [176; 177; 178; 179], their ability to compose known concepts in novel ways is a work in progress [180; 88]. For the remainder of this chapter, we will refer to the Ideas as primitives.

A simple example. Consider two objects, a *red car* and a *sports car*. These two objects have something in common: the primitive. How can we leverage this knowledge? What about two other objects: a *red car* and a *big red car*? How are

the objects related? How about a *small blue car*? We can appreciate that we are traversing multiple compositions: from *car* to *red car* to *big red car*, and these traversals can often involve backtracking: *big red car* to *red car* to *small red car*. If the primitives are commonly occurring, we may find that a pre-trained diffusion model [178] can already provide such capabilities out of the box. When it fails for a primitive with multiple nested attributes, say for a less common primitive, it is not easy to diagnose such cases in advance and/or simply re-run the model with additional guidance.

This "compositional" problem setting is not unique or novel to our work – indeed, many existing approaches [181; 182; 183; 184; 185; 186] have variously approached the task of compositional learning. However, even now, when Foundation models dominate nearly every popular benchmark, compositionality remains an open question and contemporary architectures can fail on even simple compositionality tasks [187; 188; 189] since, as demonstrated in Figure 3.6, compositional latent spaces are hard to obtain in an unsupervised way. In the compositional task above, the main focus is on the *relationships* between the samples, often with various levels of nesting. The individual samples are relevant but only as a "seed". For the "relations", what constraints should be checked when operating in the latent spaces of large models? Given recent discussions surrounding emergent capabilities in large models, can we check to what extent the latent spaces support compositions?

What is required for understanding compositions is a mechanism to handle, operate on, and reason with structured inclusion relations (i.e., directed) between entities (e.g., car in the example above) during or post- training. Category theory provides us with precisely these types of tools. Category Theory (Section 2.2) is a general (but abstract) mathematical theory of structures and of *systems* of structures [48; 56; 54] and unifies seemingly unrelated fields (such as Group Theory and Topology) using Categories as its building blocks. One of the basic operations in Category Theory is to compose more complicated objects/systems via simpler ones – when these objects/systems correspond to "relations" or "structure", we

directly obtain the main ingredients we will need to bake into our learning model. The high-level goal of this work is to check what these tools can reveal about the latent space of foundation models commonly used in the community. In doing so, we will see that solutions for compositional zero shot learning (CSZL) fall out directly by a simple instantiation of basic axioms from Category Theory.

Contributions. On the *technical* side, using the setting in compositional zero-shot learning as a jumping off-point, we give a succinct mathematical formulation based on Category Theory, as introduced in Section 2.2. To impose structure, we study a simpler formulation that allows a direct generalization to multi-attribute compositional learning tasks (Section 4.1). On the *practical* side, we propose a simple, attention-based, instantiation of our formulation that can handle both single and multi-attribute compositions with no adjustments (Section 4.1). After demonstrating how our machinery can handle a well-studied problem setting (CSZL), we study what the formulation can say in the context of the latent space of contemporary LLMs (Section 4.2).

4.1 Proof of concept: Compositional Zero-shot Learning

To test drive the concepts above, we start with a simple problem setting. Single-attribute Compositional Zero-shot learning [183; 184] is defined as follows. Assume a labeled (image) dataset of the form

$$\mathbb{T} = \left\{ \big(\mathbf{x}, (a, p)\big) \,|\, \mathbf{x} \in \mathbb{X}, (a, p) \in \mathbb{Y} \right\} \tag{4.1}$$

where $\mathbf{x}$ corresponds to an image and each image's label has two parts; $a \in \mathbb{A}$ is an attribute (e.g., *"small"*) and $p \in \mathbb{P}$ is a primitive (e.g., *"car"*). Using $\mathbb{T}$, CZSL seeks to accurately characterize unseen images whose labels are novel pairs of attributes and primitives.

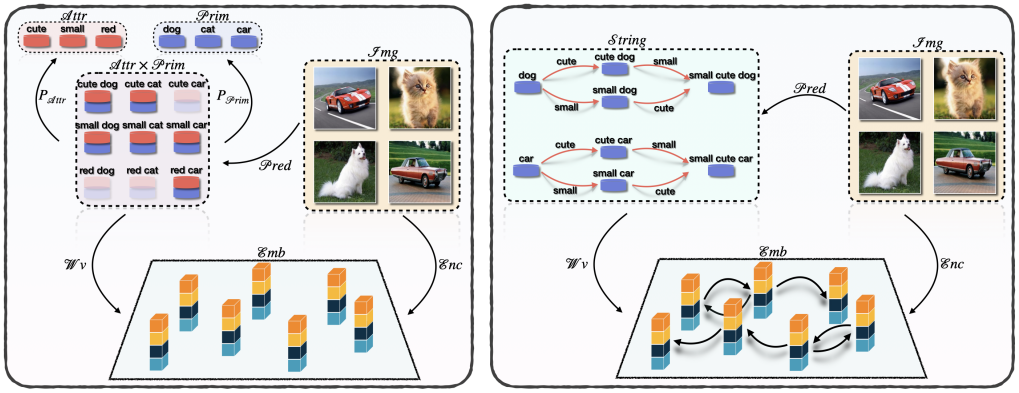


Figure 4.2: Left: the typical formulation of CZSL, in Category Theory terms. The opaque pairs correspond to implausible pairs that are not considered under the Closed-World setting, resulting in a smaller space. Right: an alternative formulation that allows for a direct generalization to the multi-attribute setting, by Morphisms' composition.

4.1.1 The "product" formulation and its limitations

Fig. 4.2 (left) restates a common formulation in CZSL, represented as a Category-Theoretic diagram. The approach roughly involves:

(a) Dealing with attribute-primitive pairs in CZSL, expressed as a discrete Product Category $\mathfrak{A} \times \mathfrak{P}$ (Definition 2.7).

(b) Transitioning from words and pairs using various design choices (mostly involving pretrained word embeddings such as GloVe [190]), modeled as a single Functor ($\mathcal{W}^f : \mathfrak{A} \times \mathfrak{P} \to \mathfrak{E}$) in the embedding space Category $\mathfrak{E}$.

(c) Associating each image with a label of the form (attribute, primitive), represented as a discrete Category of Images ($\mathfrak{I}$) with a "prediction" Functor $\mathcal{P}^f : \mathfrak{I} \to \mathfrak{A} \times \mathfrak{P}$.

(d) Closing the circuit by training CZSL methods to learn representative image embeddings, modeled as a Functor $\mathcal{E}^f : \mathfrak{I} \to \mathbb{R}^n$ which encodes each image into the same embedding space, ensuring commutative operations,

$$\mathcal{E}^f = \mathcal{W}^f \circ \mathcal{P}^f \tag{4.2}$$

Applicable to LLMs? Although at a high level, this formulation may look too

simple to capture what is going on in larger models, this is not the case. Consider each input sentence as a multi-product of words (or more accurately tokens). We indeed learn a mapping of this product to an embedding space, using training tasks such as next-word prediction or masked predictions [81; 83]. Of course, images are missing in this description but we can check that CLIP [147] closely follows the formulation of Fig. 4.2 (left) by aligning a multi-word string (caption) to an image.

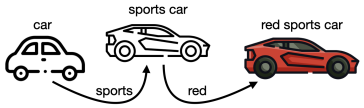
No interactions. The "product" formulation is direct and intuitive, but by recasting everything in Category Theory language, we recognize a key shortcoming; all Categories are missing a critical component: Morphisms. No interaction is assumed between the different (attribute, primitive) pairs or the embedding vectors, e.g., the Objects *small car* and *red car* have no relationship. The result in [183] recognized this issue and used a graph between all primitives, attributes, and pairs. Incorporating this missing component, obvious from our Category-Theoretic reinterpretation, led to state-of-the-art results in [183]. This hints at the potential benefits of the general categorical approach.

4.1.2 The Categorical Composer

Building on the observations above, we present a simple reformulation, which overcomes the issues we identified. In Fig. 4.2 (right), we present an alternative model; we call it the "morphism" formulation (as opposed to the "product" formulation in Section 4.1.1).

The "morphism" formulation. Similar to Section 4.1.1, we will use Fig. 4.2 (right) as a guide, and walk through the main components, step by step.

(a) We assume that each attribute is a Morphism, instead of an Object. Each attribute "acts" upon an Object and changes it in a well-defined way. According to this view, different concepts are intrinsically related and not disjoint from each other (similar to the real world).



Interestingly, we can define all multi-attribute concepts (e.g., *red sports car*) in this way simply as the *composition* of Morphisms. There is no need to define a new Category (for a multi-Product of attributes and primitives).

(b) Based on this view, we define the $\mathfrak{S}$ Category. Its Objects are all strings of zero or more attributes together with a primitive (e.g., *car*, *sports car*, *red sports car*).

(c) Since each attribute is a Morphism, there is a (directed) *arrow* from each string that *does not* include this attribute to a string that *does* include it (e.g., $\mathcal{T}_{\text{red}}$: *sports car* $\xrightarrow{red}$ *red sports car*).

(d) Similar to the "product" formulation, we define the Functor $\mathcal{W} : \mathfrak{S} \rightarrow \mathfrak{E}$ that maps this structure to Category $\mathfrak{E}$ that corresponds to an embedding space. We can assume that $\mathfrak{E}$ is the commonly used $\mathfrak{S}et$ Category where each Object is a *set* of one or more vectors in $\mathbb{R}^n$ and each Morphism is a function $\mathcal{T} : \mathbb{R}^n \rightarrow \mathbb{R}^n$. The mapping of the attributes to these functions is what allows traversing the embedding space and perform multi-attribute composition (not possible with the "product" formulation where both attributes and primitives are Objects, i.e., there are no Morphisms).

(e) Finally, we define the Category of images ($\mathfrak{I}$) along with the Functors $\mathcal{P}^f : \mathfrak{I} \rightarrow \mathfrak{S}$ and $\mathcal{E}^f : \mathfrak{I} \rightarrow \mathfrak{E}$, in a similar way. However, notice that the label can easily be a multi-attribute string such as *big red sports car*.

What is the embedding of a multi-word string? Embedding multiple words often relies on heuristics like concatenation or addition of word embeddings. Even with language models, the use of heuristics such as the mean of output word embeddings to define the entire string's embedding is common. Consider a simple example: *small cute dog*. In both cases, determining $\mathcal{W}^f$(*small cute dog*) is important. The "morphism" formulation indirectly defines it, using the commutative

property of Functors (Definition 2.3). So,

$$\mathcal{W}^f(small\ cute\ dog) = \mathcal{T}_{small} \circ \mathcal{T}_{cute} \circ \mathcal{W}^f(dog), \tag{4.3}$$

simplifies the learning process to focus on primitives' embeddings and one Morphism for each attribute.

Use of Special Variables. Each String Object in $\mathfrak{S}$ Category that represents a primitive (e.g., *dog*) is a *Limit* Object, signifying it as the purest representation of all Objects corresponding to that primitive paired with one or more attributes. Assuming attribute order insignificance, each attribute pair forms a *Pushout*, and with existing inverse Morphisms (e.g., $\mathcal{T}_{\text{cute}}^{-1}$, $\mathcal{T}_{\text{small}}^{-1}$), it becomes a *Pullback*. Assuming inverse Morphisms in our $\mathfrak{S}$ Category (as in [184]), Objects for the same primitive, irrespective of attributes, are *Isomorphic*. This embodies the notion that attributes preserve the "essence" of a primitive.

Implementation details. Here, we provide high-level details of a specific instantiation of our formulation (denoted CatCom) by modeling each concept with a neural network module. Similarly to existing works, we use a pre-trained (on ImageNet [55]) ResNet model [13] together with a trainable feedforward network on top, as the $\mathcal{E}nc$ Functor and pretrained word embeddings (transformed through an MLP) as inputs to the Morphisms (Fig. 4.3).

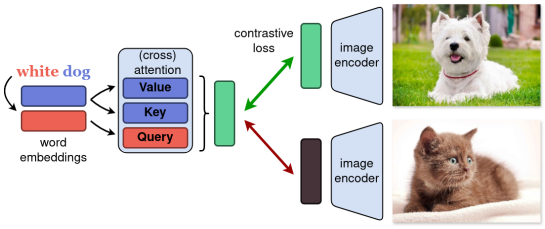


Figure 4.3: CatCom overview. Slightly different from Self-Attention [14], the Query vector is derived from another input (the attribute). The entire architecture is trainable, using a contrastive loss.

We use Cross-Attention, which has been used successfully in other scenarios (e.g., Stable Diffusion [178]), to merge the information of a primitive and an attribute (i.e., act as our Morphisms). Using the attention mechanism, we obtain a vector that encapsulates the information of both inputs. This architecture allows us to optimize our model end-to-end, similar to CGE [183] and OADis [185],

Figure 4.4: Qualitative results for C-GQA. In black, we give the true label and below the top-3 predictions. Even from this small sample of images, the difficulty of the CSZL task is apparent. In many cases, our model makes a prediction that is accurate but not the exact ground truth label. Interestingly, we observe that in many cases our model predicts a pair that is more suitable than the actual label (e.g., in the case of the sitting bear).

by maximizing the cosine similarity for the correct pair while minimizing the similarity with all other pairs, using the cross-entropy loss. We choose cosine similarity over ℓ_2-loss (also permissible in our model) because the ℓ_2-loss implies that all images with the same label must also have exactly the same embedding. The choice of ℓ_2-loss in [184] may be due to the use of group actions.

4.1.3 Experimental results

In this section, we test our method to both single (CW and OW) and multi-attribute CZSL. We follow the evaluation protocols of existing works [183; 184] and test against the current state-of-the-art methods in all cases.

Baselines. We evaluate our method against strong baselines, including: **(a) SymNet** [184]: Handles both single and multi-attribute cases. **(b) CompCos** [186]: Operates in the OW setting, regularizing all possible combinations by assessing similarity between attributes and primitives. **(c) CGE** [183]: Utilizes a GNN for composition embeddings, excelling in the CW single-composition case with a trainable backbone. **(d) OADis** [185]: Implements a triplet loss, works well for CW single-composition by fine-tuning the last ResNet [13] layers. Our baselines, as well as our CatCom, are all based on ImageNet-trained vision models, in contrast to more recent works (e.g., [191]) that use larger models (e.g., CLIP

[147]) which may violate the zero-shot nature of the problem.

For multi-composition tasks, we consider: **(a) GALM** [192]: Performs multi-label classification across all attributes, with an automated architecture-building procedure. **(b) FMT** [193]: Similar to GALM, automatically designs a multi-task network for multi-attribute classification. **(c) AMT** [194]: Designs a multi-task network, considering correlations between attributes (*dark* and *black*). Note that these methods predict attribute sets through multi-label classifiers, a task simpler than CZSL.

Metrics. We use the same evaluation protocol used in all baselines [183]. We use the same bias scheme in our predictions, so we report **(i)** the Area Under the Curve (**AUC**), **(ii)** The best **Seen** and **Unseen** Accuracy, and **(iii)** the Harmonic Mean (**HM**) between the Seen and the Unseen values. We refer to [183] for more details of the evaluation protocol. In the multi-composition case, since the above metrics are not directly applicable, we consider **mAUC** so that we are consistent with existing works for this problem.

Method	mAUC	
	aPU	SUN
AMT [194]	84.5	82.5
FMT [193]	70.5	75.5
GALM [192]	84.2	86.5
SymNet$_s$ [184]	79.9	86.7
SymNet$_m$ [184]	83.4	**88.4**
CatCom	**84.7**	87.9

Table 4.2: Results for multi-attribute composition. CatCom is comparable to SymNet$_m$ despite our simple evaluation scheme. SymNet$_s$ underperforms in both cases.

Datasets. We evaluate using three prominent datasets for single attributes: **(i) MIT-States** [195]: The oldest dataset with 245 primitives, 115 attributes, and 28K pairs (only 1262 seen). **(ii) UT-Zappos** [196]: Focuses on shoe images, featuring 192 pairs in total. **(iii) C-GQA** [183; 197]: Recent larger dataset with over 200K pairs derived from GQA [197]. In all datasets, we use the Generalized CZSL splits from prior works [183; 198]. For multi-attribute, we use: **(i) aPY** [199]:A dataset with 64 attributes and 32 primitives, created from PASCAL VOC 2008 [200]. **(ii) SUN** [201]: A larger multi-attribute dataset (102 attributes, 700 primitives).

Main results/observations. Our results are in Tab. 4.1 and Tab. 4.2 while qualitative results are depicted in Fig. 4.4.

		Method	MIT-States				UT-Zappos				C-GQA			
					Best				Best				Best	
			AUC	HM	Seen	Unseen	AUC	HM	Seen	Unseen	AUC	HM	Seen	Unseen
"Closed-World"	"product"	CompCos [186]	4.5	16.6	25.8	24.1	19.8	35.2	58.9	42.7	2.6	12.5	28.9	10.8
		CGE [183]	5.1	17.2	28.0	25.2	24.7	38.9	58.8	61.0	2.5	11.9	27.5	11.7
		OADis$_t$ [185]	5.9	18.9	28.9	25.0	30.0	44.4	59.5	65.5	3.5	14.8	31.0	13.7
		CGE$_t$ [183]	**6.5**	**21.4**	31.6	**27.5**	33.0	47.3	61.8	66.3	3.6	14.5	31.4	14.0
	"morph"	SymNet [184]	4.1	16.3	26.2	23.1	27.7	42.5	58.8	61.0	1.8	9.8	25.2	9.2
		CatCom	5.7	18.5	30.8	26.4	29.2	43.0	60.2	64.4	3.5	14.3	**31.6**	**14.2**
		CatCom$_t$	6.3	19.5	**32.1**	**27.5**	**35.3**	**48.8**	**64.7**	**70.1**	**4.3**	**15.8**	**34.1**	**16.0**
"Open-World"	"product"	CompCos [186]	1.5	8.4	25.4	10.0	17.8	32.9	56.1	44.6	0.2	1.0	18.0	0.9
		CGE [183]	1.4	8.0	26.2	8.1	15.4	30.9	58.6	35.9	0.3	1.9	26.8	1.5
		OADis$_t$ [185]	2.1	10.3	29.6	11.6	21.5	**37.4**	56.0	**50.6**	0.5	3.6	29.9	2.9
		CGE$_t$ [183]	**2.2**	**10.6**	**31.6**	10.4	16.0	30.3	60.0	37.5	0.6	3.2	30.0	2.7
	"morph"	SymNet [184]	0.8	5.8	21.4	7.0	20.2	36.8	54.7	44.7	0.2	0.8	14.7	1.7
		CatCom	1.9	9.5	28.8	10.4	17.7	33.0	55.6	43.8	**0.7**	**4.0**	28.8	**3.1**
		CatCom$_t$	**2.2**	**10.6**	31.1	**12.1**	**22.1**	**37.4**	**62.6**	48.0	**0.8**	**4.6**	**31.6**	**3.1**

Table 4.1: Results for Closed and Open World. The subscript t stands for a trainable backbone. Even our frozen version (denoted CatCom) achieves results comparable to the state of the art methods (with a trainable backbone) and it is better than other methods that use a frozen backbone. When we also train the backbone (denoted CatCom$_t$) then we get a performance similar to CGE$_t$ which is the state-of-art-method. In C-GQA specifically, the most challenging dataset, the simplicity of our method results in much better results than all other methods.

4.2 Compositionality in Language Models

With a general recipe for thinking in a "compositional manner" in hand, we examine if compositionality exists in contemporary large models. This analysis, possible because of the general nature of our formulation, allows us to provide a mechanism to better understand and interpret LLMs, a relevant but underdeveloped topic. We know that models such as BERT [81] and CLIP [147] have been trained with "simpler" loss functions, enabled by huge training datasets. The question then is: "Are these models able to shape their latent space in a way that resembles what a compositional learning algorithm would provide out of the box?"

Quantifying the performance of diversely-trained LLMs remains an active topic of research, with metrics such as the hallucination index [202]. Here, we consider multiple large-foundation models and examine what insights, if any, our structured modeling of the latent space can provide. The reader will notice that at a minimum, our formulation can yield a "compositionality index" and we can check whether large violations are indicative of anything at all.

Dataset, Models and Setup. Using the same terminology as before, we define our Forms to correspond to common everyday primitives (the full set of classes of ImageNet-1K [55] and CIFAR100 [203]). To assess structure in the latent space, we consider the four most common attributes that will act upon the Form alone (e.g., for images, it does not affect the background): size (e.g., small), color (e.g., green), texture (e.g., shiny), and age (e.g., old), and we examine the latent space of multiple widely used models: **(i) BERT** [81] **(ii) Albert** [204] **(iii) Roberta** [205] **(iv) Deberta** [206] **(v) CLIP** [147] **(vi) GPT2** [83] Following existing works [207], in all models we use the average of all the output embeddings (i.e., one per input token) to form a single embedding for the given prompt.

A note on model selection. We chose the above models since these are some of the most widely used text encoders in practice, for multiple and diverse text-based applications. In fact, our experiments will offer some guidance on why these models are widely used and why, in some cases can be a suitable choice (relative to models like Mistral LLM/Zephyr). To highlight the differences between the

two classes of models, we also consider the following three LLMs: **(i) Phi-2** [208] **(ii) Zephyr** [209] **(iii) Mistral** [210] .

In all our experiments, we find the difference vector between the following two quantities: **(i) plain embedding** which corresponds to the embedding of the expression "an image of a(n) ⟨object⟩", and **(ii) attributed embedding** which corresponds to the embedding of the expression "an image of a(n) $\langle \text{attribute}_1 \rangle$, ⋯, and $\langle \text{attribute}_n \rangle$ ⟨object⟩". The plain embeddings correspond to the Forms (the Limits of our String Category $\mathfrak{S}$) and the differences between the two embeddings correspond to our Morphisms, the $\mathcal{T}$ arrows (or functions). We will use the same notation here: the difference will be $\mathcal{T}^{\text{object}}_{\text{attr}_1\&\cdots\&\text{attr}_n}$.

What are we looking for? Notice that $\mathcal{T}^{\text{object}}_{\text{attr}}$ are exactly the Morphisms in §4.1.2, and $\mathcal{T}^{\text{object}}_{\text{attr}_1\&\cdots\&\text{attr}_n}$ is exactly the composition of these Morphisms (i.e., $\mathcal{T}^{\text{object}}_{\text{attr}_1\&\cdots\&\text{attr}_n} = \mathcal{T}^{\text{object}}_{\text{attr}_n} \circ \cdots \circ \mathcal{T}^{\text{object}}_{\text{attr}_1}$). Based on this observation, we define **compositionality** to reflect the property that the composition of multiple *atomic* attributes is equivalent to a complicated expression that includes all of them. LLMs may be operating in one of the three following regimes, **(i)** (extreme 1) LLMs have no internal notion of compositionality (i.e., they are "pure" black boxes) and we cannot link them back to our world's structure, **(ii)** (extreme 2) LLMs are able to perfectly pick the compositionality of our world, based on its extensive training data, or **(iii)** (middle) Contemporary training procedures offer a partial "understanding" of compositionality and different training schemes lead to models that align more or less with this view.

Scope of experiments. We will use our formulation to ask the following questions for the three models. How similar are the Morphisms for different Forms? What about Morphisms for different attributes? How well do multiple Morphisms compose?

Remark 4.1. Different *attribute types affect plain embeddings with* different *magnitudes.*

Consider a primitive, say, a car (either in text or joint image+text), and assume two variations are available: in the first, it is larger and the second, shows it in

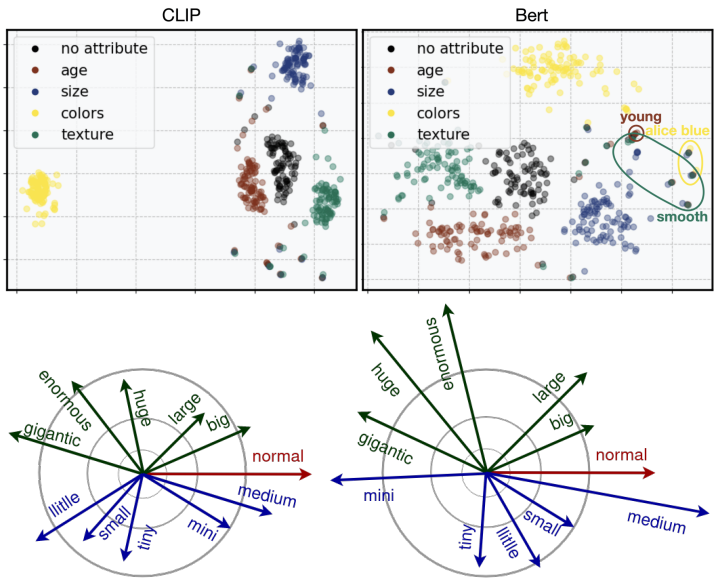


Figure 4.5: (Top) T-SNE plot of the clusters that different attributes form, for all objects. Even in this lossy projection, we can observe that color (yellow cluster) leads to a cluster further from the black cluster (plain embeddings) compared to age, size, and texture. (Bottom) the relationship between all the size Morphisms reflects how we would also arrange them in real life.

a different color. In the latent space, which is more similar to the original? All models concur that the change in size is *less significant* than the change in color.

Remarkably, all models share the same attribute order (of how much "work" $\mathcal{T}$ does), despite being trained on diverse data and objectives. We find that "size" is unanimously the attribute with the smallest magnitude, while "color", "texture", and "age" have similar magnitudes (although usually "age" is slightly smaller). This relationship can be seen in Fig. 4.5 (top) also, where we project the embeddings to the two-dimensional space using T-SNE [211]. Besides the intra-attribute ordering, the models show an inter-attribute ordering that is consistent with our world-view. In Fig. 4.5 (bottom), we show the angle between $\mathcal{T}_{\text{normal}}$ and $\mathcal{T}_{\text{attr}}$ for all other size attributes, for GPT2 and BERT. Both models are able to understand the relative relationships between the different size attributes.

Morphism outliers. From our analysis, we observed the existence of some outliers, for each type of attributes. These outliers correspond to pairs of (atttibute, primitives) that are rare, or even unlikely to ever occur. For instance, some of the

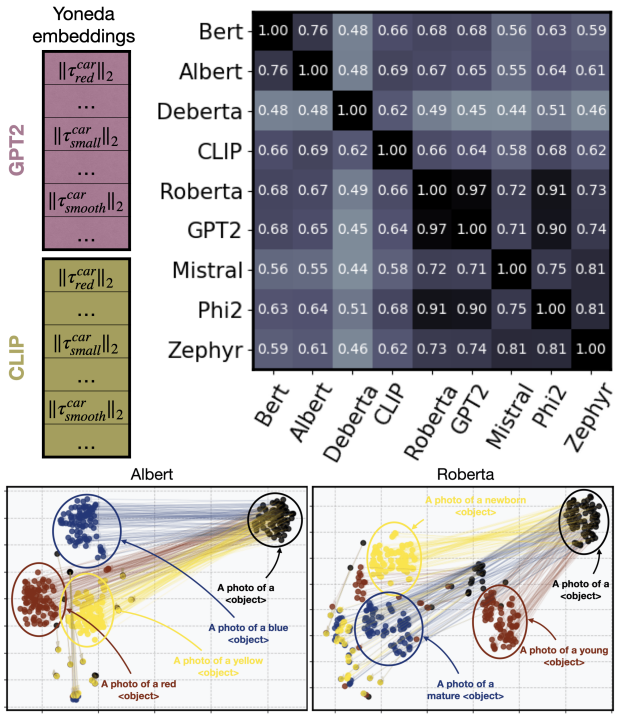


Figure 4.6: (Top) The pairwise correlation of the Yoneda embeddings between all models. On the left, we show how we form the Yoneda embedding for each model. The correlation map reveals that Bert and Albert are clustered together, as well as that Roberta, GPT2, and the three LLMs have a high-to-absolute correlation. (Bottom) The latent space of the models is clustered into distinct regions that correspond to each attribute while the homogeneity of each attribute (colored lines) is apparent.

pairs are (knitted, bowl), (magenta, baby), and (appicot, woman). This is expected to happen – during training, there is no categorical structure embedded in their latent space, any unknown compositions will be arbitrary and will not behave well.

Besides the primitive-specific magnitude, we observe that, on average, simple and common attributes such as “old”, “young”, “rough”, “cracked”, “big”, and “little” appear to have, on average, a smaller magnitude than more rare/exotic attributes such as “gigantic”, “childlike”, “embossed”, and “time-worn”.

Remark 4.2. *The Yoneda perspective reveals clusters of similar models.*

The Yoneda perspective states that an Object is completely characterized by its "neighbors" and the Morphisms by which they are connected. A similar characterization has recently occurred in [212]. There, the authors calculate the characteristic embeddings using the so-called "anchor points", leading to a construction that has been formally defined in Category Theory as *Yoneda Embeddings*.

To this end, we construct the Yoneda embeddings [213]. The Yoneda embedding can be thought of as a succinct representation of all neighbors of an Object. In our case, we form the vector whose elements are the magnitudes of each Morphism (i.e., $\|\mathcal{T}_{\text{attr}}^{\text{object}}\|$). We would expect these vectors to be *similar* enough for different models, since all of them were trained on data that resemble the real world. Indeed, when we consider the Pearson Correlation [214] we can observe that all of them are sufficiently similar to each other. In Fig. 4.6 (top) we can observe the pairwise Correlation between all models. All models have a high, positive correlation, confirming our hypothesis that these models are, in a way, isomorphic views of the same world.

Several intriguing patterns emerge from the analysis. First, the formation of two distinct clusters is evident: Bert aligns closely with Albert, while Roberta exhibits strong alignment with GPT2 and all three LLMs. This suggests a degree of similarity in their embedding spaces. Surprisingly, Roberta's embedding space appears almost isomorphic to that of GPT2, indicating its efficacy in practice. This finding corroborates Roberta's widespread adoption in practical applications, e.g., as a popular choice in Kaggle competitions on text-data.

The observation regarding Deberta is interesting. Despite its effectiveness, Deberta stands out as an outlier, displaying a significantly lower correlation with other models. This distinctiveness also aligns with its use in practice, where Deberta is often utilized as a complementary component in ensemble solutions due to its unique properties. This characteristic was particularly evident in the recent Kaggle competition titled "LLM - Detect AI Generated Text" [215] where Deberta featured prominently in all of the top solutions. Overall, these insights

provide a nuanced understanding, enabled by our categorical casting, of the relationships between various language models, shedding light on their comparative performance and potential synergies in practical applications.

Remark 4.3. *Models are homogeneous w.r.t. how the same attribute affects different objects and* ***homogeneity*** *(i.e., the similarity of an attribute's embedding across different objects) is a unique property that cannot be estimated by the dataset size, the model size, as well as the training type alone.*

One major assumption of our framework is the fact that an attribute (e.g., *small*) acts the same way on each primitive (hence the knowledge sharing we exploit in CZSL). For LLMs, this would imply that the Morphisms (τ vectors) would be (almost) the same for a specific attribute across all different Objects. To this end, we examine the cosine similarity across different objects for the same attribute (Fig. 4.7 (right)). While all models have relatively high homogeneity, the value differs in a way that no clear correlation exists with the model's size, its training type, or even the release date. This indicates that homogeneity is a unique property that can provide an alternative view of the models. This will become apparent in Remark 5, in which we consider positional attributes also. In Fig. 4.6 (bottom) we can also observe the 2D projection of 6 attributes. Even when we "squeeze" the high-dimensional embeddings into two dimensions, the homogeneity of the attributes is preserved.

Object-specific attributes. An interesting observation emerges from the range of color attributes considered. Our list encompasses a broad spectrum, including less common colors like "pear", "salmon" and "lemon". However, these colors are also used as food nouns. Consequently, when paired with the primitive "can", all models tend to interpret the prompt as referencing a can of lemons, salmon, etc., rather than a can of that specific color. This discrepancy affects the calculation of $\mathcal{T}_{\text{attr}}^{\text{can}}$, rendering them dissimilar to the τ vectors of other primitives. Additionally, we noticed variations in size-related attributes, particularly with "medium" hinting at its broader usage beyond indicating object size (overloaded).

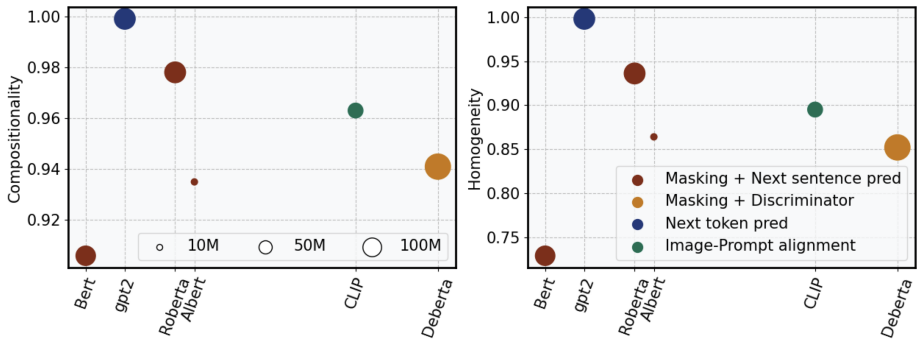


Figure 4.7: Compositionality (left) and Homogeneity (right) of each text encoder. The models are arranged in chronological order, based on their release date. We observe that all models have the same profile with respect to both metrics, and that compositionality/homogeneity can not be explained solely by the model's size, or the training type.

Remark 4.4. *Compositionality is mostly preserved and can be predicted by attribute homogeneity. Its violation indicates regimes where training data were scarce.*

After single-attributed prompts, we turn our focus to multi-attribute prompts. One of the basic questions, and a major thrust of our formulation, is that compositions must be preserved and their violation can be checked. In the two-attribute case, this implies that: $\mathcal{T}^{X}_{\text{attr}_1\&\text{attr}_2} = \mathcal{T}^{X}_{\text{attr}_1} + \mathcal{T}^{X}_{\text{attr}_2}$, $\forall$ object X. In Fig. 4.7 (left), we show how well each model preserved our compositionality constraint, as a function of model size, training type, as well as release date. Interestingly, all models show a good compositional behavior on average, although many outliers hint that compositionality is not universally respected in these models. A striking observation is the fact that compositionality can be accurately estimated by attribute homogeneity (Fig. 4.7). While the values may differ, the ordering between the models is the same across the two metrics. In practice, this means that compositionality can be easily estimated quickly since it does not require the quadratic complexity of compositions.

When does compositionality fail? Our evaluation of the constraint violation suggests that compositionality does not hold for either specific, rare attributes or for far-fetched compositions. For instance, an attribute that led to a consistently

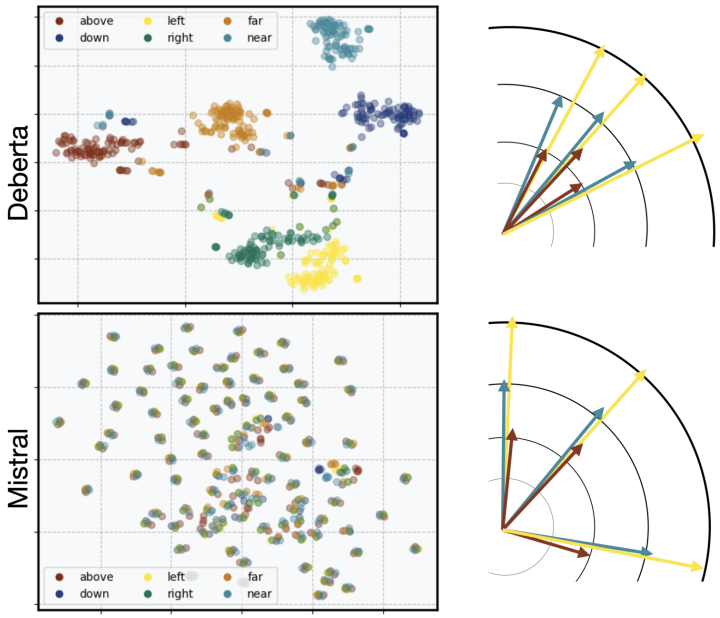


Figure 4.8: How viewpoint invariance manifests to different models. In Deberta (top) there are clear clusters for each position, hinting at a latent space in which different object embeddings have a small angle with each other, while different viewpoints lead to embeddings with the same direction but with big differences in the norm. This leads to this type of clustering when we consider a 2D projection. On the contrary, in Mistral (bottom) each object's viewpoints are clustered together, hinting at a latent space in which each object occupies a very different area.

low compositionality score was the color "puce", no matter the type of the other attribute. Another such attribute was the size-related attribute "mini". As is also apparent from Fig. 4.5 (bottom), τ_{mini} is quite different than the other size attributes. For CLIP specifically, we believe this is due to the fact that most of the "mini" images correspond to images that depict the car *Mini Cooper* and not a mini version of a typical object. Similarly, for "puce", it is very likely that images (or even texts) linked to "puce" correspond to the Franco-Belgian comic "Zig et Puce" [216] and not the color "puce". Finally, odd compositions such as young, bumpy otter and youthful, pear bear lead, as perhaps expected from *Remark* 4.1, to a low compositionality score.

Remark 4.5. *LLMs have an in-built understanding of viewpoint invariance, but their latent spaces are fundamentally different.*

So far, we considered "external" attributes, such as size and color. However, we can also consider attributes that do not really change the essence of the primitive. One such Category of attributes is the positional attributes that leads to expressions such as "an image of a car from the *front*" or "an image of a dog from *above*". Invariance with respect to 3D rotations is an active topic of research and remains an open question. We consider a list of 13 positional attributes and we compute the embedding for each combination of object and position, using the prompt "a photo of a ⟨object⟩ from ⟨position⟩". If we denote each embedding as $\mathbf{e}^{\text{object}}_{\text{position}}$, we can estimate the viewpoint invariance of each model by calculating the cosine similarity between all the embeddings $\mathbf{e}^{\text{object}}_{\text{position}_1}$ and $\mathbf{e}^{\text{object}}_{\text{position}_2}$ for all combinations of positions and objects.

While we observe that the average cosine similarity is very high (more than 0.95), this does not imply that all models are the same. In Fig. 4.8 we depict the latent space (T-SNE 2D plot) of two of the models under consideration: the smaller, widely used text-encoder Deberta [206], and a popular more recent LLM Mistral [210]. We can observe an interesting behavior. While Deberta's latent space is such that (while it achieves viewpoint invariance) the embeddings of each positional attribute are clustered together, this is not the case for Mistral. In Mistral, the different viewpoints are clustered together for each object, leading to this "spotted" view of the latent space, which may not be that useful in practice. We believe that this is one of the properties that make Deberta a widely used text encoder in contrast to LLMs (such as Mistral) that, although show strong performance on multiple NLP tasks, are not a default choice as a text encoder in vision-language tasks.

4.2.1 Compositionality in Large Language Models

Our main focus so far was in language models that are widely used in multiple applications, such as Deberta, GPT2, and Roberta. However, recently, we have witnessed the emergence of LLMs; billion-parameter models that are dominating multiple, diverse benchmarks. While the LLMs are capable of performing NLP

tasks in which smaller models (such as the ones we consider here) fail, this does not imply that they are universally more useful for all types of text-based applications (although this is being actively studied [217; 218]). According to the Yoneda perspective (Remark 4.2), LLMs such as Zephyr (3B) and Mistral (7B) do not differ significantly from smaller models such as GPT2 and Roberta (about 100M parameters). This implies that, at least for certain applications, it may be beneficial to use a smaller model compared to a larger one, or, alternatively, the use of a larger model may offer minimal improvements.

Additionally, the viewpoint invariance (Remark 4.5) reveals that, while all models have an in-built notion of viewpoint invariance, their latent space is not equally "useful". As we observe in Fig. 4.8, Mistral's latent space is formed in a way that may not be ideal for a text encoder. This statement agrees with what has been observed in practice too, where Deberta is one of the main choices for a text encoding model, in contrast to larger LLMs. Of course, our experiments do not intend to understate the improvements that various LLMs have achieved, but rather provide a more critical and informed view that shows that huge models are not necessarily well-suited for all applications. The Yoneda perspective as well as compositionality/homogeneity can serve as guidance for selecting models appropriate for the task at hand. Here, we considered a generic set of attributes (age, color, size, texture, as well as positions) but for a specific application, it is straightforward how the categorical view can be applied to a different set of attributes and provide a quick proxy of a model's performance.

4.3 Summary

We showed how compositional learning can be reinterpreted in Category Theory terms. This perspective allows us to identify strategies to concurrently improve as well as simplify existing formulations. The experimental results suggest that our CatCom model yields a performance profile comparable to the state of the art for Compositional Zero-Shot Learning. The same core ideas offer a mechanism

to dig deeper into the latent space of LLMs, from a categorical perspective. Our tests allow us to evaluate less well-studied properties of LLMs, and the potential role the Yoneda perspective and the compositionality index may offer. These metrics can provide insights into how the model is trained and what extra training examples can help. Our multi-level analysis of multiple contemporary LLMs revealed multiple anecdotes that hold in the community of NLP practitioners. Our categorical view, along with the specific metrics we presented, can be used for text-based applications and provide a "ranking" of the candidate models. A preliminary version of this chapter was published as [219] and the implementation can be found in `https://github.com/SPChytas/CatCom`.

5 FOGE: FOCK SPACE INSPIRED ENCODING FOR GRAPH PROMPTING

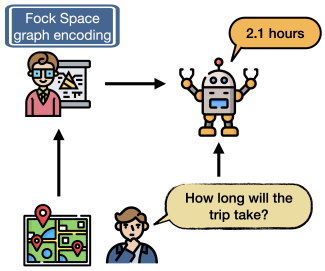


Figure 5.1: Augmenting LLM's capabilities by prompting them with carefully encoded graphs.

Large Language Models (LLMs) excel at tasks like question answering, sentence completion, translation, and even solving undergraduate-level math problems [220; 221]. However, they sometimes need additional data unavailable during training. For instance, a model trained on data up to a specific date may struggle with the ever-changing news cycle [222; 223]. To prevent responses from becoming outdated, or to integrate non-public/proprietary data and domain-specific terminology, models need extra context. Retrieval Augmented Generation (RAG) describes this process of retrieving and integrating extra information to an LLM during its generation process. While multiple different approaches have been proposed for the retrieval piece, a common solution for integrating additional information is In-Context Learning (ICL) [224; 225; 226; 87; 227]. ICL allows additional information to be included with a prompt, guiding the model to generate responses aligned with the extra context. This method is useful as it does not require retraining the LLM and can be applied to proprietary models like GPT [82] by adding a text description of the extra information.

ICL-type ideas are also being studied for utilizing not just additional/new data but also novel input formats/modalities, such as tables and graphs [228; 229; 230; 231]. While specialized models will still perform better at specific tasks, LLMs can serve as general-purpose reasoning machines, capable of answering questions about the provided modality beyond the training labels. Several recent results have reported success at "serializing" such structured data-types into a text-form description that can be easily used within ICL. For tables, the serialization is not

too complicated [228; 229], but more care is needed for graphs. While different types of graphs can all be handled by the same pipeline, the efficacy of the model varies [232; 230; 231]. Further, it has been observed that specific design choices to "textify" the graph can influence performance and additionally, prompting techniques can have more than a small impact on results [232]. What will work well in a specific setting depends on both the question at hand as well as the characteristics of the data [233; 234].

Prefix-tuning. One option to address the issues above is "prefix-tuning" (Section 2.5.1). A specialized graph encoder translates the underlying graph into embeddings that can be fed directly to an LLM, removing the need for a text description. Although not training-free, the LLM remains *frozen*, and only the *relatively smaller* graph encoder is trained. This approach works well, often surpassing ICL-based methods [235; 236; 237]. However, using a specialized graph encoder can be challenging due to the *variety* of graph types, and multiple works have proposed modifications of GNNs that suit their needs. For example, GraphToken [233] can encode only simple graphs, while GNP [238] constructs a complex pipeline to handle large graphs and extract subgraphs. GraphLLM [234] combines a transformer and a GNN (about 100M parameters), requiring detailed text descriptions for each node. Adapting these models to different graph types (e.g., protein-derived graphs or hypergraphs) is difficult; even familiar graph types need adjustments for new tasks.

Context of this chapter. ICL-based approaches for graphs primarily involve converting graphs to text, while prefix-tuning with graphs uses modules to extract richer, *task-relevant* structures, requiring larger sample sizes and more compute. A key question is whether we can achieve powerful, task-agnostic graph representations that are as easy to obtain as ICL-based methods. Could a lightweight adapter map these rich (but task-independent) representations into the LLM embedding space, making prefix-tuning effective for various tasks? Recent results hint that this may be viable [239]. For instance, a *single linear layer* can transform an arbitrary image encoder's outputs to align with CLIP's text encoder embed-

dings [147]. If our graph encoding captures the graph's information and structure well enough, a similar adapter could work with a pre-trained LLM to offer good performance. This approach's success depends on the quality of the graph representations. We ensure this by invoking a mature concept from mathematical physics, called Fock Spaces [240], whose practical instantiation yields almost lossless task-agnostic graph embeddings. Our findings show that a linear adapter with these representations yields competitive performance, handling complex graph questions and diverse structures like hypergraphs and proteins. The **main contribution** of this work is the Fock-space inspired encoding of diverse graph-based structures, ranging from simple graphs to those obtained from proteins. We provide code for grounding LLMs using our graph encodings as prompts and profile the performance of this pipeline relative to baselines, on diverse datasets.

5.1 Deriving Fock space based Graph Representations

We will first review notations/results which will together provide the conceptual pipeline. While graphs serve as representative examples here, the rationale for structured data such as tables is similar.

Setup/rationale. Consider a graph $G = (\mathbb{V}, \mathbb{E})$ with a vertex set $\mathbb{V}$ and an edge set $\mathbb{E}$; $|\cdot|$ denotes set cardinality. We define the *incidence matrix* [241], $\mathbf{I}$ to be of shape $|\mathbb{V}| \times |\mathbb{E}|$ where $[\mathbf{I}]_{ij} = 1$ if *edge j ends at vertex i*, -1 if *edge j starts at vertex i* and 0 elsewhere. Let $|\mathbb{V}| = N$. It is common to represent graphs via graph spectra derived from the Laplacian's eigenvalues. This is effective for studying global properties of graphs like connectivity/symmetries [242; 243] but less so for capturing localized relationships between individual entities (nodes, edges, faces) within the graph. It turns out that an interesting direction using Clifford Algebra (Section 2.3), shown to be effective in geometric problems in machine learning [244; 245; 244; 65; 246], provides us tools for representing various graph

elements (nodes, edges, faces) in a nice algebraic structure [247] **at once**. *Why?* Graphs can be embedded and manipulated in a geometric space [248], and in principle, their spectral properties can also be studied. Following Clifford algebra axioms exactly allows us to build higher-order elements (like edges, hyperedges) while preserving graph structure. However, a full implementation faces practical issues: an M-dimensional space requires a 2^M-dimensional Clifford algebra, and this is impractical. Therefore, we make design choices that balance rigor with computational feasibility.

5.1.1 From Graphs to Clifford Algebra to Fock Spaces

Laplacian and the Dirac operator. For graph G, consider the Laplacian $\mathbf{\Delta} = \mathbf{II}^T \in \mathbb{R}^{N\times N}$, where $\mathbf{I}$ is the incidence matrix [249] as defined above. $\mathbf{\Delta}$ represents only one component of the Hodge Laplacian $\mathcal{L}$, which acts on the full exterior algebra of the graph. This captures relations between all grades: scalars (grade-0), nodes (represented as grade-1 vectors), edges (as grade-2 bivectors), and so on. Here, the Dirac operator $\mathcal{D}$ serves as a "square root" of the Hodge Laplacian [250], satisfying $\mathcal{D}^2 = \mathcal{L}$ [249]. Because the Dirac operator and the graph Laplacian are connected through this identity, we can use the algebraic machinery associated with Dirac operators, i.e., Clifford algebra, for our graph encoding problem.

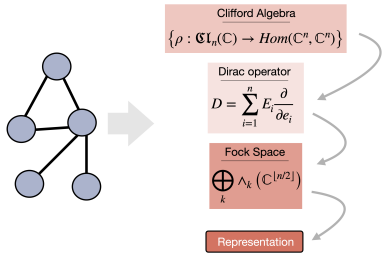


Figure 5.2: From graph to Fock space representations.

Connection to Clifford algebra. The Dirac operator arises naturally from Clifford algebra [251]. Recall that the Clifford algebra is *generated* by $N = |\mathbb{V}|$ abstract basis vectors, which we denote as $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$. These are the grade-1 elements of the algebra. The full structure (dimension 2^N), is then built up from these generators through repeated application of the Clifford product, yielding elements of all grades up to the top grade-N element. One feature of this algebra is the anticommutation relations that these generators satisfy: $\mathbf{e}_i\mathbf{e}_j + \mathbf{e}_j\mathbf{e}_i =$

$-2\langle \mathbf{e}_i, \mathbf{e}_j \rangle \mathbf{1}$ for basis elements. This algebraic structure guarantees the $\mathcal{D}^2 = \mathcal{L}$ property holds [252; 43].

Spinors and Fock space. We now need to identify the space on which this algebra acts. For the complex Clifford algebra, there exists a special irreducible representation on a complex vector space $\mathbb{S}$ of dimension $2^{\lfloor N/2 \rfloor}$, called the Spinor space [252; 43; 253]. This Spinor space is where the abstract Clifford algebra elements, the generators $\mathbf{e}_k$ and their products become concrete operators. However, for graph encoding purposes, we need to handle grades. The Fock space $\mathbb{F} = \bigoplus_{k=0}^{\lfloor N/2 \rfloor} \wedge^k(\mathbb{C}^{\lfloor \mathbf{e}/2 \rfloor})$ provides what we want: a graded structure that holds objects of different types concurrently: scalars (grade-0), nodes (as grade-1 vectors), edges (as grade-2 bivectors), and so on. The key point is that for the Spinor space $\mathbb{S}$ and the Fock space $\mathbb{F}$, there exists an isomorphism (i.e., $\mathbb{S} \cong \mathbb{F}$) between them [252; 43]. This identification is a link: the Clifford algebra's action is defined on $\mathbb{S}$, and the isomorphism allows us to translate this action onto $\mathbb{F}$, which is the graded "container" of our graph elements.

Why is this useful? We can now ask: what do the generators $\mathbf{e}_k$ of our Clifford algebra look like when they act on this graded container? When the abstract Clifford generators $\mathbf{e}_k$ act on the Fock space, they decompose into two operations: a creation operator $\boldsymbol{\tau}_k^*$ and an annihilation operator $\boldsymbol{\tau}_k$. That is, $\mathbf{e}_k \mapsto \boldsymbol{\tau}_k + \boldsymbol{\tau}_k^*$ [252]. The creation operator $\boldsymbol{\tau}_k^*$ increases the grade of an element (e.g., acting on grade-0 to create a grade-1 vector, or acting on a grade-1 vector to form a grade-2 bivector), while the annihilation operator $\boldsymbol{\tau}_k$ decreases the grade. Here, the Dirac operator itself is combining all these creation and annihilation operations: $\mathcal{D} = \sum_k (\boldsymbol{\tau}_k + \boldsymbol{\tau}_k^*)$ [43]. The main takeaway for our graph encoding is that the algebraic structure underlying our graph is one of creation and annihilation. Nodes (grade-1) combine to form edges (grade-2), which in turn can combine to form higher-order structures. Unfortunately, this is intractable due to $2^{\lfloor N/2 \rfloor}$ dimensions.

5.1.2 Translating Theory to Practice: Instantiating a Graph Representation

The Fock space formulation provides ideas for representing multi-particle systems, viewing particles as nodes in a graph. As noted above, implementing the full structure, in high dimensions, is infeasible. Vector Symbolic Architectures (VSA), as explored in recent works [46; 254] and introduced in Section 2.3.1, offer a practical approximation of Fock spaces with compute efficiency. In VSA, the binding operation (circular convolution) approximates the creation/annihilation operators, while the superposition operation (vector addition) resembles the direct sum in Fock spaces. Although the VSA $\leftrightarrow$ quantum mechanics connection is not new [77], in this context, it provides specific ideas for efficiency.

Representing nodes, sums, and products. In our implementation, we assign a high-dimensional vector ($\in \mathbb{R}^M$) to each concept (node, edge, and so on). These vectors are analogous to the basis elements above. While ideally, these vectors would be orthogonal, similar to the properties of basis elements in a Fock space, we simply approximate this by sampling from a normal distribution $\mathcal{N}(\mathbf{0}, 1/M)$. This leads to nearly orthogonal vectors, with the maximum absolute cosine similarity between any two vectors typically below 0.1 [255].

To emulate operations in Fock space, we use dimensionality-preserving operations instead of tensor products, avoiding exponential growth in dimensionality [77]. This ensures all embeddings maintain the same dimensionality. We define sum ($\oplus$) as element-wise addition and product ($\otimes$) as circular convolution, analogous to Fock space's creation/annihilation operators. Circular convolution is done via element-wise multiplication in the Fourier domain followed by an inverse Fourier transform. As M increases, these operations asymptotically satisfy Fock space's algebraic properties, with complexity $\mathcal{O}(M \log M)$. This framework also supports inverse vectors, where $\mathbf{a} \otimes \mathbf{b} = \mathbf{1}$. Other properties like commutation relations, superposition, and self-commutation are mostly satisfied. Note that our experiments are not tied to this specific implementation (improved choices can be dropped in).

Dealing with infinitely many concepts. In some datasets, vertices include text descriptions, making random vector initialization unsuitable. To address this, we use text encoders like CLIP [147], BERT [81], and RoBERTa [205], which map text to vectors that preserve information and place similar sentences in close proximity (as we have shown already in Chapter 4). This approach allows us to: **(1)** generate infinitely many vectors, and **(2)** ensure similar vectors represent similar concepts. When dimensionality allows, we retain default sampling and explicitly note the use of text encoders in our experiments.

Other works using Vector Symbolic Architectures. Vector Symbolic Architecture (VSA), rooted in symbolic AI, leverages high-dimensional representations alongside logical rules for combining symbols/vectors (Section 2.3.1). Many studies mechanistically derive ways to construct symbols and implement merge operations. The use of Fock space for symbolic manipulation has been explored, with applications in trajectory analysis [77]. Additionally, VSAs have been employed for computational efficiency in self-attention calculations as seen in HRRFormers [45; 46], while a preliminary investigation of the application of VSAs to graphs can be found in [78].

5.2 Fock Graph Encoder (FoGE)

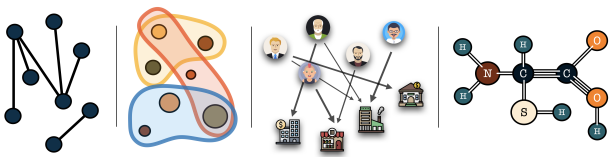


Figure 5.3: Graphs, Hypergraphs, Attributed graphs, Proteins. All these types can be efficiently encoded using FoGE.

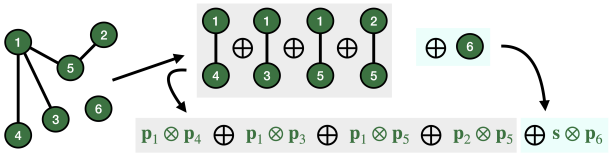
Based on the concepts from §5.1, we use a parameter-free scheme (denoted FoGE) to obtain rich graph embeddings. Our approach is general and can handle a large spectrum of different graph types, and its extension to novel graph-types is straightforward. Diverse graph types such as hypergraphs, attributed graphs, as well as proteins (Figure 5.3) can all be modeled easily providing an alternative or a good initialization for more intensive trainable models. This approach translates the concepts

of Fock spaces into a practical/efficient method for graph representation, where graph features obtained by the encoding are analogous to multi-particle states in a Fock space.

For a graph $G = (\mathbb{V}, \mathbb{E})$ we have a set of vectors $[\mathbf{p}_i]_{i=1}^{N=|V|}$, using i to index the nodes. We also use an extra vector $\mathbf{s}$ for the graph's size, a practical design choice we will explain shortly. Then, with these $N + 1$ vectors, we obtain a lossless Fock-space based representation $\mathbf{g}$ as:

$$\mathbf{g} = \left(\mathbf{s} \otimes \mathbf{p}_n\right) \oplus \bigoplus_{(i,j)\in\mathbb{E}} \left(\mathbf{p}_i \otimes \mathbf{p}_j\right) \tag{5.1}$$

Our formulation follows from §5.1. Each edge's endpoints are fused together using $\otimes$ and then we aggregate all edges together using $\oplus$. Finally, the graph's size is also added using the special vector $\mathbf{s}$.



Lossless representation. The above representation is lossless. Assuming we use Equation (5.1) to get a graph's embedding $\mathbf{g}$. Then, simply by evaluating the expression $\mathbf{p}_j^T(\mathbf{p}_i^{-1} \otimes \mathbf{g})$, we can determine whether the edge (i, j) exists in the edge set of that particular graph. In this way, we can recover, one by one, all edges of the graph and correctly reconstruct it, if desired. It is instructive to check the importance of $\mathbf{s}$. By evaluating the expression $\mathbf{p}_i^T(\mathbf{s}^{-1} \otimes \mathbf{g})$, $\forall i$, we can first obtain the size of the graph. This can inform the edge retrieval above because an expression of the form $\mathbf{p}_{n+x}^T(\mathbf{p}_i^{-1} \otimes \mathbf{g})$ could, in practice, produce a number close to 1, although there is no such edge. By first obtaining the size of the graph, we have a "safeguard" against such phantom edges beyond the real vertex-set.

Vertex attributes. Consider a graph $G = (\mathbb{V}, \mathbb{E}, \mathbb{A})$, where the set $\mathbb{A}$ (with $|\mathbb{A}| = |\mathbb{V}| = N$) consists of attributes, one for each vertex. There is no restriction on the type of attributes: it can denote numerical values or text, or any other

concept. Let $\mathbf{a}_i$ be the vector associated with the attribute of vertex $i \in \mathbb{V}$ (using an appropriate text-encoder if needed). Then, we can augment Equation (5.1) to absorb the extra information in the following way:

$$\mathbf{g} = \left(\mathbf{s} \otimes \mathbf{p}_n\right) \oplus \bigoplus_{(i,j)\in E} \left(\mathbf{p}_i \otimes \mathbf{p}_j\right) \oplus \bigoplus_{i \in V} \left(\mathbf{p}_i \otimes \mathbf{a}_i\right) \tag{5.2}$$

The graph is again, fully reconstructable. We have also encoded each vertex's attribute (which can be recovered by the expression $\mathbf{a}_j^T(\mathbf{p}_i^{-1} \otimes \mathbf{g})$). We should think of proteins as a graph with vertex attributes where each vertex is a specific amino acid (possibly with 3-D coordinates).

Hypergraphs (Theory versus Practice). Hypergraphs are generalizations of graphs: each edge is connected to an arbitrary number of vertices, instead of just 2 (Figure 5.3). In theory, we can easily augment Equation (5.1) so that we can handle hypergraphs as follows:

$$\mathbf{g} = \left(\mathbf{s} \otimes \mathbf{p}_n\right) \oplus \bigoplus_{(k_1,\cdots k_m)\in \mathbb{E}} \bigotimes_{i=1}^{m} \mathbf{p}_{k_i} \tag{5.3}$$

In practice, aggregating many multiple vectors together may be unstable. This is true for our particular design choices for calculations (e.g., circular convolution), so we use an alternative approach. We can start by observing that each edge can be interpreted as a unique cluster of vertices, so we simply assign a unique vector $\mathbf{e}_i,\ i \in \left[|\mathbb{E}|\right]$ to each edge in the hypergraph. This modification allows us to encode the hypergraph similar to how a graph is encoded as a dictionary, in the following way:

$$\mathbf{g} = \left(\mathbf{s} \otimes \mathbf{p}_n\right) \oplus \left(\bigoplus_{i=1}^{|\mathbb{E}|} \left(\mathbf{e}_i \otimes \bigoplus_{j \in [\mathbb{E}]_i} \mathbf{p}_j \right) \right) \tag{5.4}$$

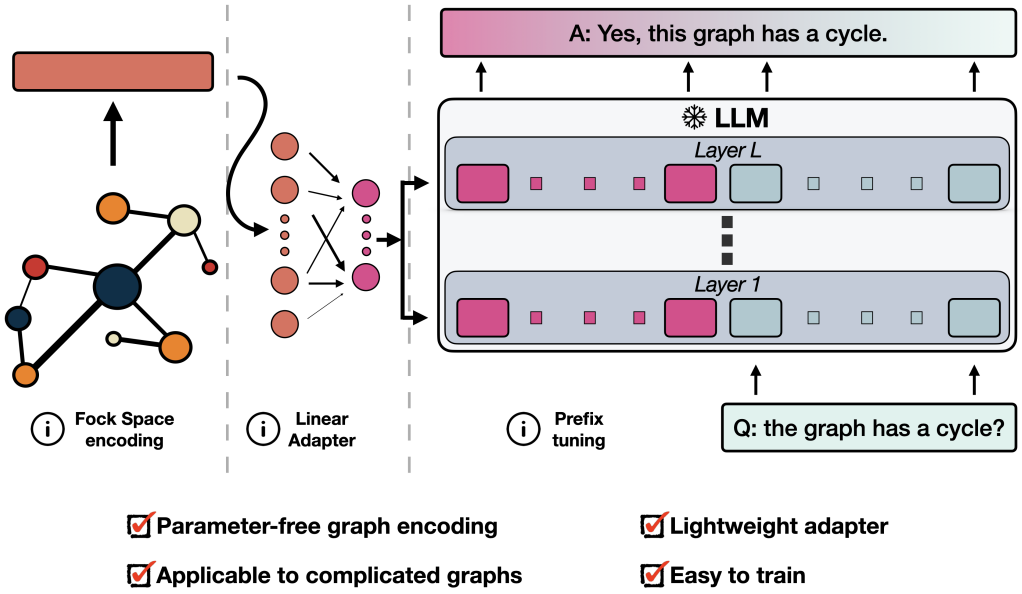


Figure 5.4: FoGE-LLM overview. Using a parameter-free graph encoder we get graph embeddings for a range of different graphs. Then, we use linear adapters with a frozen LLM for *prefix tuning*.

5.2.1 Fock Space-based grounding of LLMs (FoGE-LLM)

Recent works showed that (a) textualizing a graph and pre-appending it to a question results in better-than-random responses from the LLM (although far from perfect), and (b) using a specialized graph encoder such as a GNN or a graph transformer and training along with a frozen LLM results in a big improvement in performance, resulting essentially in LLMs that can understand, to some extent, graphical structures. One takeaway is that we can bypass the most tedious stage of designing application-specific graph encoders. Instead, we can use a **parameter-free** method for a wide range of graph types, as we described above. Thus, the *only* trainable parts of the pipeline are simple linear adapters that convert the raw graph encodings to a format "understandable" by an LLM. Our FoGE-LLM is shown in Figure 5.4. After getting the graph encodings, we train one/more linear adapters and append the transformed encodings to the question's embeddings fed to the LLM.

Summary and Takeaway. We highlight some key advantages. *First*, our graph encoding is parameter-free and efficient. The complexity of aggregation is $\mathcal{O}(M \log M)$ (M is vectors' dimension) and the number of aggregation operations is linear (in graph size). *Second*, our encoder is not restricted to specific graph types: it works easily for simple graphs, for proteins and for hypergraphs just via small modifications. In contrast, GraphToken [233] uses a specific GNN whose output size is dependent on the underlying task whereas GraphLLM [234] uses a transformer model together with a GNN (also specific to the underlying task). These properties simplify our training and eliminates any tunable components. *Third*, our open-source code offers a scalable way to train FoGE-LLM even on consumer GPUs, by using FSDP [256]. For reference, GraphToken [233] is trained on TPUs (code unavailable) whereas GraphLLM [234] has a large memory/compute footprint (trained on A100 80GB).

5.3 Experimental results

We examine our Fock-space based encoding in two separate settings: (a) as a stand-alone input of a simple model, and (b) as an extra prefix in a frozen LLM (FoGE-LLM), for graph prompting. Our initial experiments (Section 5.3.1) show that simple models can process FoGE embeddings quite well for traditional tasks. In Section 5.3.2, we present that FoGE can be successfully combined with an LLM, leading to two advantages over stand-alone models: (1) language interface for flexible graph queries without pre-defining task types, and (2) easier integration with mature software ecosystems built around LLMs that reduce deployment overhead.

Datasets and Models. We performed experiments on multiple graph reasoning datasets: from simple graph-understanding tasks to hypergraphs and proteins and aim to cover different aspects of graph understanding/reasoning. Specifically, we consider the 7 following datasets/dataset collections: **(i) GraphQA** [232] **(ii) GraphReasoning** [234] **(iii) HyperGraphQA (iv) PPI** [257] **(v) OBNB**

	GraphQA			HyperGraphQA		Jaffe	
	n nodes	n edges	has cycle	n nodes	n edges	n acids	n links
MSE/Acc	0.67	0.03	98.7%	1.12	0.63	2.95	11.9
Model size	32K	8K	16K	32K	4K	32K	16K

Table 5.1: Using a small neural network with a single layer on the obtained graph representations allows us to perform near-perfectly on tasks such as *number of nodes* and *number of edges*, for both synthetic and real data.

Method	ROC-AUC
HyperFusion	0.8475 ± 0.0003
PAS + Fingerprint	0.8420 ± 0.0015
HIG	0.8403 ± 0.0021
DeepAUC	0.8352 ± 0.0054
FoGE + Fingerprint	0.8305 ± 0.0068
RF + Fingerprint	0.8208 ± 0.0037
FoGE	0.7614 ± 0.0051
GCN	0.7606 ± 0.0097
GIN	0.7558 ± 0.0140

Table 5.2: Results on mol-HIV [259; 261]. The full details can be found on `https://ogb.stanford.edu/docs/leader_graphprop`

[258] **(vi) mol-HIV** [259] **(vii) SabDab** [260]. More details about the datasets can be found in the appendix. Exploring diverse graph reasoning datasets helps evaluate our model's performance and generalization across various graph structures and domains, from traditional graph-based QA to hypergraph understanding and biological network analysis. By including datasets like PPI and BioGRID, we seek to check the practical relevance of our result, with potential applications in biology, network analysis, and more. We use the Llama2 (7B) model [85] as the frozen LLM, and we use only extra linear adapters for the graph embeddings obtained using our formulation. We adjust vector dimensionality from 512 to 2048 and use just a *single adapter* for the entire model or *one adapter per layer* in FoGE-LLM.

5.3.1 Proof of Principle Evaluations for Graph Understanding

Setup and Results. While our key goal is graph-prompting, we first perform multiple preliminary checks of the effectiveness of our graph encoding. We conduct three different types of experiments.

First, we evaluate whether our graph embeddings are informative (i.e., they preserve the graph's structure), by using a small, 1-hidden-layer FFN for basic graph-understanding tasks. The results in Section 5.3 show that our representations are rich and informative. More specifically, we assess the quality of the embeddings by first encoding graphical structures from 3 different classes of graphs (graphs, hypergraphs, and proteins) and then training a small model (one hidden layer) to predict graph attributes like *number of nodes* and *edges*. The results indicate that our representations are rich and informative and only few parameters suffice to achieve almost-perfect performance on such tasks.

	Model	F1
	Random	39.2
	Node2Vec [262]	40.9
	Raw features [262]	42.2
Unsupervised	GraphSAGE-min [257]	46.5
Unsupervised	GraphSAGE-max [257]	50.2
Unsupervised	DGI [263]	63.8
Unsupervised	GRACE [264]	66.2
Unsupervised	**FoGE**	**99.2**
Supervised	GraphSAGE-min [257]	50.0
Supervised	GraphSAGE-max [257]	61.2
Supervised	LGCN [265]	77.2
Supervised	GAT [266]	97.3
Supervised	GCNII [267]	**99.5**

Table 5.3: Micro F1-score on PPI. Our approach is better than the best unsupervised approaches and better/comparable to the supervised approaches.

Additionally, for more involved tasks, we consider the Open Graph Benchmark (OGB) [261] (more specifically, the mol-HIV[259] dataset) and we show that our encoding is better than multiple, heavy, specialized, and trainable methods while being training-free and unsupervised! In Table 5.2, we show the results. Interestingly, because our method allows a seamless integration of additional graph information (like a molecule's footprint), we find that this can help us achieve even better results and, in some cases, be competitive with submissions at the top of the leaderboard. To obtain the final results, we used AutoGluon [268] on our unsupervised embeddings, with a time limit of 10 minutes, similarly to the strategy of many of the other baselines.

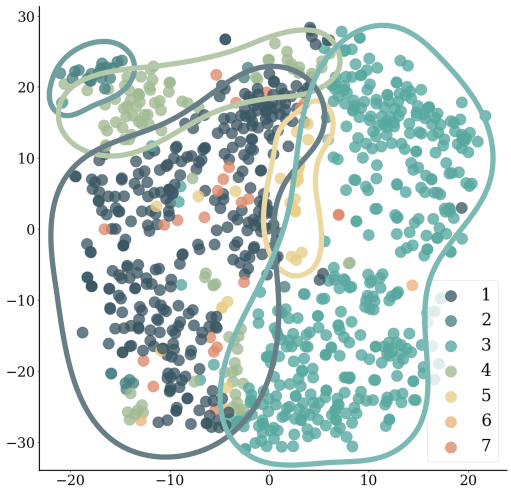


Figure 5.5: T-SNE plot of the SabDab embeddings. Although the dataset is very small, each one of the populated clades occupies a different region and, interestingly, clades 1 and 7 are very similar, just like in real life. The T-SNE plot was robust to different choices of hyperparameters, with no significant differences beyond simple translations of the space.

Second, we examine whether our graph encodings preserve important biological markers of the data. To test this, we use a small dataset of about 900 proteins (SabDab [260]) which are accompanied by affinity data that corresponds to each protein's clade. More specifically the SabDab proteins [260] are annotated with the heavy/light chain pairing, hemce we can extract the clades and visualize their embeddings with respect to that information. As a brief reminder, the clades correspond to superfamilies of proteins that share a common ancestor [269]. To extract the clades we used the V gene heavy chain and chose seven families. It is well known from biology that antibodies that belong to the same clade are *more similar* than antibodies across different clades, so, here, we examine if this real-world, biological property is reflected on our embeddings. Specifically, after obtaining each protein's embedding using FoGE (in an unsupervised fashion without using the clade annotations), we apply a T-SNE [270] transformation on the high-dimensional vectors so that we are able to plot them, with a significant amount of noise, in just two dimensions. Although the dataset has only few samples and some of the clades are scarcely populated, we can observe that there

is a clear separation between the most populated clades in the embeddings space (Figure 5.5).

Third, we examine if the same encoding practice can generate rich node-level encodings, by encoding for each node, the subgraph that is generated by itself and its neighbors. We examine performance on **nineteen** real protein datasets (OBNB [258] and PPI [257]). The detailed results on the complete OBN Benchnark can be found in Section 5.3.1), while Table 5.3 demonstrates FoGE's performance on PPI. We see that our approach is, in all datasets, among the best unsupervised approaches, and is also competitive (if not better) than specialized supervised approaches that leverage trainable, graph-specific models such as GCN [271] and GAT [266]. Specifically, we achieve state-of-the-art performance in PPI. We also achieve the best results (among both unsupervised and supervised) in seven out of the eighteen datasets of OBNB.

These results provide encouraging evidence that (a) our approach gives "rich" graph embeddings for a range of different graph types and styles, and (b) our graph embeddings can be used as an extra, grounding input to a powerful LLM without the need to design/train a specialized model, e.g., GNN [272; 273] or a Graph Transformer [274].

5.3.2 Grounding LLMs with Graph prompting

We next investigate whether such an encoder can be successfully integrated with an LLM. Can an LLM understand graph structure? Our experimental evidence suggests that it can: the model learns to reason about graph properties and, in many cases, performs better than approaches relying on heavily specialized graph encoders. While this line of research is still in its early stages and requires further exploration, it closely parallels recent multimodal advances. Just as LLaVA [93; 96; 95] acquires visual understanding and TimeLLM [275] develops temporal reasoning once their respective modalities are properly embedded, our findings indicate that LLMs can develop some graph reasoning capabilities when provided with rich structural representations such as those produced by FoGE.

Network	Model	DISEASES	DisGeNET	GOBP
BioGRID	LabelProp	1.210	0.931	1.858
	LogReg	1.556	1.026	2.571
	GCN+BoT	1.511	1.014	2.442
	SAGE+BoT	1.486	1.031	2.402
	GIN+BoT	1.410	1.007	2.386
	GAT+BoT	**1.609**	1.037	**2.624**
	GatedGCN+BoT	1.547	1.038	2.517
	FoGE	1.599	**1.062**	2.433
HumanNet	LabelProp	3.728	3.098	3.806
	LogReg	3.812	3.158	**4.053**
	GCN+BoT	3.552	3.053	3.921
	SAGE+BoT	3.401	3.052	3.816
	GIN+BoT	3.513	3.054	3.861
	GAT+BoT	3.761	3.100	3.809
	GatedGCN+BoT	3.677	3.086	3.889
	FoGE	**3.853**	**3.254**	3.916
COMPPIHumanInt	LabelProp	1.352	1.106	2.076
	LogReg	1.644	1.240	**2.806**
	GCN+BoT	1.648	1.211	2.685
	SAGE+BoT	**1.694**	1.210	2.629
	GIN+BoT	1.608	1.219	2.611
	GAT+BoT	1.665	1.230	2.755
	GatedGCN+BoT	1.672	1.218	2.735
	FoGE	1.660	**1.241**	2.586
BioPlex	LabelProp	0.964	0.939	1.714
	LogReg	**1.358**	**0.939**	2.587
	GCN+BoT	1.324	0.911	2.553
	SAGE+BoT	1.246	0.865	2.513
	GIN+BoT	1.349	0.868	2.504
	GAT+BoT	1.355	0.873	2.548
	GatedGCN+BoT	1.301	0.859	2.590
	FoGE	1.273	0.879	**2.599**
HuRI	LabelProp	0.545	0.598	1.086
	LogReg	0.650	0.656	1.084
	GCN+BoT	0.634	0.693	1.129
	SAGE+BoT	0.593	0.679	1.190
	GIN+BoT	0.583	0.702	1.143
	GAT+BoT	0.667	0.687	1.174
	GatedGCN+BoT	0.596	0.695	**1.195**
	FoGE	**0.684**	**0.729**	1.070
OmniPath	LabelProp	1.358	0.897	1.593
	LogReg	1.542	**1.093**	**2.125**
	GCN+BoT	**1.577**	1.068	2.071
	SAGE+BoT	1.478	1.062	1.986
	GIN+BoT	1.452	1.073	1.993
	GAT+BoT	1.552	1.048	2.068
	GatedGCN+BoT	1.516	1.049	2.071
	FoGE	1.511	1.085	2.102

Table 5.4: FoGE vs multiple unsupervised and supervised methods, in OBNB. After obtaining our embeddings, we use a Random Forest to predict the corresponding node's label. The evaluation is based on the APOP metric [258] and we can observe that FoGE is always comparable to the best methods, while in almost half of the cases it is the best one.

	ICL	GraphToken		**FoGE-LLM**
Tokens	$\mathcal{O}(n^2)$	1	$\mathcal{O}(n)$	1
num of nodes	26.9%	**99.6**%	-	97.2%
num of edges	12.8%	42.6%	-	**45.1**%
cycle existence	83.2%	95.6%	-	**97.9**%
num of triangles	16.2%	34.8%	-	**37.7**%
edge existence	54.4%	-	73.8%	**74.3**%

Table 5.5: GraphToken vs FoGE-LLM on GraphQA. Column *1* stands for a single embedding for the entire graph; $\mathcal{O}(n)$ stands for a single embedding per node. In all 6 tasks, although we use a parameter-free, predetermined graph encoding, we see a performance similar/better relative to a trainable graph encoder with a larger LLM.

Graph Understanding. In our first experiment, we examine whether an LLM can answer questions about a graph's structure, such as the number of nodes, the presence of cycles, and so on. We use GraphToken and conduct a suite of six different experiments. Although our method's encodings are *not* specific to each underlying task, it performs competitively with specialized models, as shown in Section 5.3.2. Even when GraphToken uses different embeddings for each node (*node degree*) or edge (*edge existence*), our model still achieves comparable results using a single embedding for the entire graph.

Advanced Graph Reasoning. Going beyond "simple" graph understanding tasks, we also examine our performance on more complicated graph-reasoning tasks, using the GraphReasoning dataset [234]. GraphToken is not applicable here since each node is accompanied by a textual description which cannot be handled by that model. So, our main baseline is GraphLLM, which uses a transformer combined with a GNN to merge the graphical/textual information into one or more embedding vectors. Similar to GraphToken [233], GraphLLM [234] also utilizes a different approach for each task, using multiple graph embeddings for each task. In contrast, we achieve comparable performance using a *single graph embedding*, showcasing the versatility/richness of the graph embeddings (Table 5.6). Further, we see that using a pretrained text encoder such as RoBERTa [205] to generate the vectors is reasonable, and results in a similar performance. This is a strong improvement over traditional symbolic methods, by allowing a

	GraphLLM	**FoGE-LLM**	
model size	100M	**25M**	
question specific output	Yes	**No**	
graph embeddings	5	**1**	
vectors	-	random	RoBERTa
substructure count	99.9%	97.3%	95.6%
max triplet sum	95.7%	94.6%	94.7%
shortest path	97.2%	95.7%	95.8%
bipartite match	99.8%	98.1%	97.3%

Table 5.6: GraphLLM vs FoGE-LLM. Although we are using the same, predetermined graph embedding for each task, we enjoy a performance similar to GraphLLM which leverages 5 graph embeddings, specific to the task at hand. The *vectors* stands for the two approaches we follow in generating them: (a) randomly generated (almost) orthogonal vectors (ignoring the node's text description), and (b) using RoBERTa [205] and utilizing all vertices' information.

		Zero-Shot	Few-Shot	**FoGE-LLM**
HyperQA	num of nodes	04.5%	16.8%	**85.0**%
	num of edges	03.9%	27.0%	**95.4**%
	node degree	02.1%	10.1%	**53.9**%
	edge existence	65.9%	79.4%	**87.9**%
Jaffe	num of amino-acids	03.9%	17.1%	**99.3**%
	num of links	03.8%	06.1%	**13.2**%
	amino-acid type	01.4%	12.3%	**37.7**%

Table 5.7: FoGE-LLM performance against ICL techniques for hypergraphs and proteins.

large set of "symbols"/vectors. Dealing with proteins is similar to advanced graph reasoning, since both datasets are graphs with additional node information.

Furthermore, in Table 5.7, we show the accuracy of FoGE-LLM for three protein-related tasks on Jaffe. Although the size of the protein graphs is more than 10× larger compared to the ones in GraphQA and GraphReasoning, our model is able, up to some extent, to understand the provided protein, as a whole (*number of amino acids* and *number of links*) as well as at an individual-node level for the task *type of amino acid* (where we prompt the model to determine the type of a specific vertex in the protein).

Hypergraphs. Existing works focus on specific forms of graphs and rarely

applicable (or easily modifiable) to different graph types. One common family of graphs in applications is hypergraphs. Here each edge is a subset of the nodes, of arbitrary size (Figure 5.3). Our formulation can handle such a generalization of the typical graphs with only minor modifications to the encoding formulation (Equation (5.4)). Here, we show that our setup can indeed answer questions about such complicated structures, using our encodings as an extra prefix (graph prompting). Using the HyperGraphQA dataset, we assess the performance of FoGE-LLM on four common tasks. Since GraphToken as well as GraphLLM cannot handle such data, we compare our model's performance against two of the most common prompt-engineering methods: 1. zero-shot, where the model is given the graph in text form along with the corresponding question, and 2. few-shot, where the model is given pairs of textualized graphs with the corresponding question/answer pair and it is asked to produce the answer to a new combination of graph/question. The results are presented in Table 5.7. Interestingly, even though hypergraphs have a much more complicated structure than "simple" graphs, our model achieves a performance very close to basic graph understanding (Section 5.3.2), or even better at some tasks.

5.3.2.1 FoGE-LLM runtime

Besides the raw performance gains as presented above, FoGE-LLM offers a very low inference time, due to two reasons. First, we always "reserve" only a single token for the given graph. In contrast, zero/few-shot approaches that textualize the graph require a large number of tokens, and grows larger as the graph grows. This leads to an increase in inference time, due to the number of input tokens. Second, FoGE-LLM uses one or more linear adapters, no specialized architectures like in [234; 233] are needed. Based on our experiments, this impacts inference time, and gives FoGE-LLM strong efficiency benefits. In Table 5.8 we show the average inference time required for each approach.

Model	Inference time (s) ↓
zero-shot	0.175 (±0.05)
few-shot	0.541 (±0.10)
GraphLLM	0.052 (±0.01)
FoGE-LLM	0.031 (±0.01)

Table 5.8: Average inference time for each approach. FoGE-LLM is significantly lower than zero/few shot approaches since the number of input tokens does not grow with the graph size, while it enjoys a 40% improvement over GraphLLM dues to its simpler encoder/adapter.

5.4 Related Work

Geometric Algebra in Machine Learning. There is growing interest in application of geometric algebra in machine learning, particularly for developing models that maintain geometric properties. While these ideas have been leveraged in the context of equivariance/symmetry transformations in deep learning [276; 277; 278; 279; 280], the concept is finding interesting uses in recent works. For example, [281] recently proposed Clifford Neural Layers to model dynamical systems in fields like fluid dynamics and [244] described Geometric Clifford Algebra Networks (GCANs), specifically designed to respect symmetry group transformations. Beyond classical machine learning, geometric algebra finds more direct applications in quantum computing as well: [50] leveraged the isomorphism between Pauli matrices and Clifford Algebra to represent multidimensional data, to define specialized transforms for machine learning tasks.

Graphs & LLMs. The body of work describing ways to infuse extra, graphical information into a frozen LLM is sizable and growing. As discussed earlier, initial approaches focused on converting the underlying graph into natural language form, such as "node 1 is connected to node 3, node 5 is connected to node 4, ..." [230; 231; 232]. These works while far from perfect showed viability: that a frozen LLM has the capability to reason about the given graph and answer graph-related questions, such as "is there a cycle in the graph?". Practical difficulties involving the format of graph serialization is an important factor in the performance and the results tend to be only moderately better than random. The perspective taken in [233; 234] was fresh and led to an alternative approach: infusing the graph

information directly at the embedding level, by encoding the graph using a model such as a Graph Neural Network (GNN) [272; 273; 233] or a Graph Transformer [274; 234]. These works significantly improved the state of the art, showing that carefully crafted graph embeddings are key to a successful grounding of an LLM.

5.5 Summary

We have described a novel scheme to encode a graph into a vector form for direct downstream use or to augment prompts fed to LLMs. Our approach, motivated by Fock space operations, offers numerous advantages in practice demonstrated via experiments. We can obtain encodings of arbitrary graphs quickly, with no trainable parameters, that nicely captures the important information content in the underlying graph. To utilize these encodings, we introduced FoGE-LLM – a way to fuse the graph information for graph-prompting with a pre-trained, frozen LLM, allowing it to "understand" and reason about graphs. Given the growing interest in grounding LLM responses based on additional domain-specific priors, we believe that this is an interesting direction. Our model, accompanied with a simple-to-train open-source codebase, performs favorably relative to highly specialized models. It is also quite flexible and can handle classes of graphs where other alternatives fall short or need adjustments. A preliminary version of this chapter was published as [282; 47] and the complete implementation can be found in `https://github.com/SPChytas/FoGE`.

Impact & Limitations. A key strength of our method is its parameter-free approach for generating rich graph embeddings. Such an approach can be a great fit in less computationally rich environments or in cases where the dataset's size is not big enough for the trainable approaches, without, as we demonstrated extensively, lacking in performance in data-rich situations. Given the scarcity of the data in many real-life graph-related problems (like the protein-based questions we answered here), our approach can benefit multiple aspects of research. However, the unsupervised nature of FoGE also limits the ability to fine-tune performance

if the embeddings are insufficient for specific applications. So, we believe that building representation learners on top of these embeddings, as in FoGE-LLM, is a good strategy. Additionally, when dealing with infinitely large vector sets, random generation is impractical. While RoBERTa works well in our experiments, integration with other models may involve some trial-and-error to identify sensible configurations.

6 RECO: REMINDER COMPOSITION MITIGATES HALLUCINATIONS IN VISION-LANGUAGE MODELS

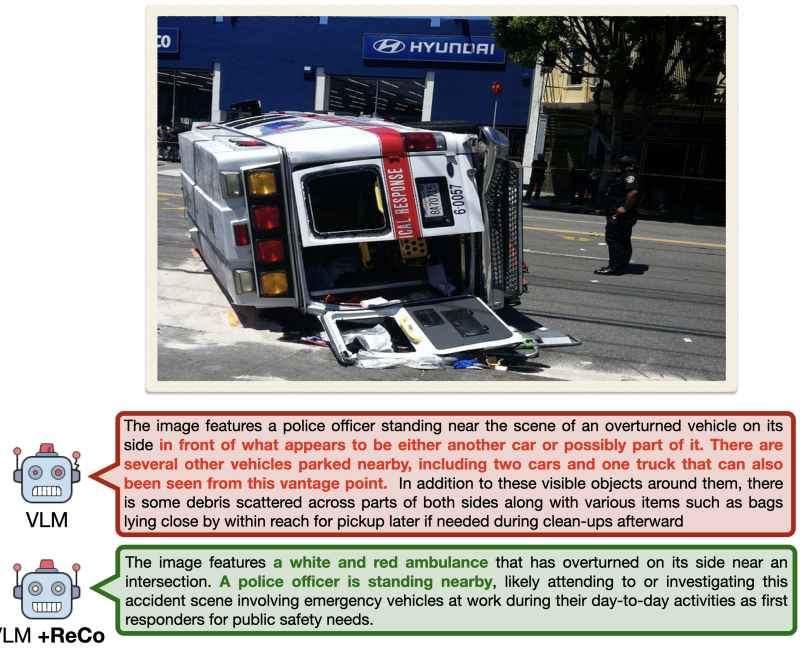


Figure 6.1: InstructBLIP before and after ReCo. We propose a small module that, with minimal training, it is able to effectively reduce the hallucination rate of widely used VLMs.

Given the advances in the capabilities of LLMs (Section 2.5), recent efforts have sought to extend these models to the multi-modality setting, i.e., processing and "understanding" additional modalities beyond text such as audio, images, and videos [92; 93; 283; 284; 285]. To this end, one milestone is the development of VLMs (Section 2.5.1) that can accept both images and natural language as input, and generate contextually meaningful outputs for tasks, including visual question answering and image captioning. Some prominent models are InstructBLIP [92], MiniGPT4 [286; 97] and LLaVA [93; 95], that show strong *image+text* understanding skills. However, we know that VLMs piggyback heavily on the

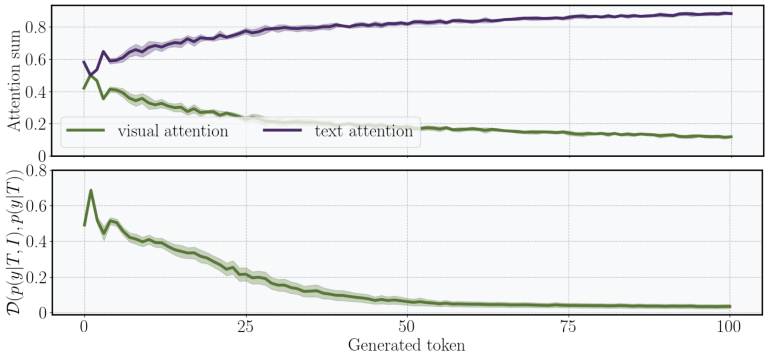


Figure 6.2: The "fading memory effect". **Top**: First-layer attention of the visual and textual input as the generation progresses. **Bottom**: The effect of the visual input on the logits distribution. We calculate the next token prediction with and without the visual input, and we compute the distributional difference. After the first tokens, the next token can be predicted just from the previously generated text.

core capabilities of the parent LLM to which the visual representations have been aligned. This endows sizable compute benefits – InstructBLIP [92] costs about 500 GPU hours, while Llama [85] (the parent LLM), needed 180000 GPU hours. But this VLM/LLM dependence means that VLMs also inherit known weaknesses of LLMs and sometimes, these weaknesses can be magnified. One example is hallucination, i.e., generating text that is plausible but does not accurately reflect the provided input. More than a handful of results in the literature show that in many cases, a VLM appears to ignore the image and generates a description that is not influenced much at all by this extra visual input [287; 288; 289], although this is an extreme case of hallucination. We show an example in Figure 6.1 where InstructBLIP [92] "sees" multiple cars and trucks, although we only see an ambulance.

Available mitigation mechanisms? Mitigation mechanisms for the foregoing problem can be classified into two categories: **(i)** training-based methods, where the VLM is finetuned with different loss functions and/or using more suitable datasets [290; 291; 292; 293]; and **(ii)** rule-based methods, where the VLM remains frozen but a new/improved generation process must be adopted. Some proposals treat the model as a black box without any access to its attention maps

and its parameters [287; 288], whereas others use them to steer the model's generation process [294; 289; 295].

Why do VLMs hallucinate? The literature suggests that VLMs over-rely on language priors, gradually "forgetting" the visual input as text generation progresses [287; 289; 294; 293; 288]. If next-token generation implicitly probes an internal conditional probability distribution, $p(\cdot|◑)$, conditioned on both text $\mathbf{T}_t$ and image $\mathbf{I}$ with the right balance ◑, the image's role progressively diminishes to $p(\cdot|◔)$ and eventually to $p(\cdot|○)$, an effect [287] calls the *"fading memory effect"*. We further probe this behavior: in Figure 6.2 (top), the total attention on visual tokens drops significantly during generation, showing that the model increasingly relies on the language prior. In Figure 6.2 (bottom), following [287], we compute the difference in the logits distribution with and without the image. If $\mathbf{T}_t$ is the text generated so far and $\mathbf{I}$ is the visual input, we calculate the difference between $\mathcal{P}(\mathbf{y}_{t+1}|\mathbf{T}_t, \mathbf{I})$ and $\mathcal{P}(\mathbf{y}_{t+1}|\mathbf{T}_t)$, where $\mathbf{y}_{t+1}$ denotes the next-token prediction. Using the Hellinger distance, this difference drops to almost 0 after the first 40 tokens, indicating that the image has negligible influence on subsequent tokens. Even in newer, stronger models (e.g., Qwen2.5-VL [296; 94]), the same behavior is apparent, despite all the improvements in the visual encoder, the number of image tokens, as well as the degree of fine-tuning (Figure 6.3).

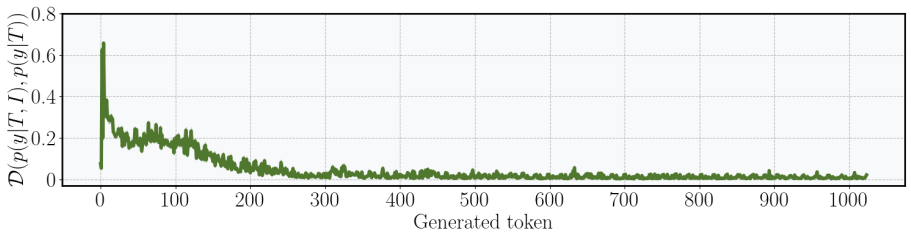


Figure 6.3: The "fading memory effect" in Qwen2.5-VL [94]. We calculate the next token prediction with and without the visual input, and we compute the distributional difference. After the first tokens, the next token can be predicted just from the previously generated text.

The desired behavior. A potential mitigation strategy involves a proper composition of both visual and textual embeddings, without overhauling the entire VLM architecture. If a VLM estimates the probability distribution $\mathcal{P}(\mathbf{y}_{t+1}|\mathbf{B}_t)$,

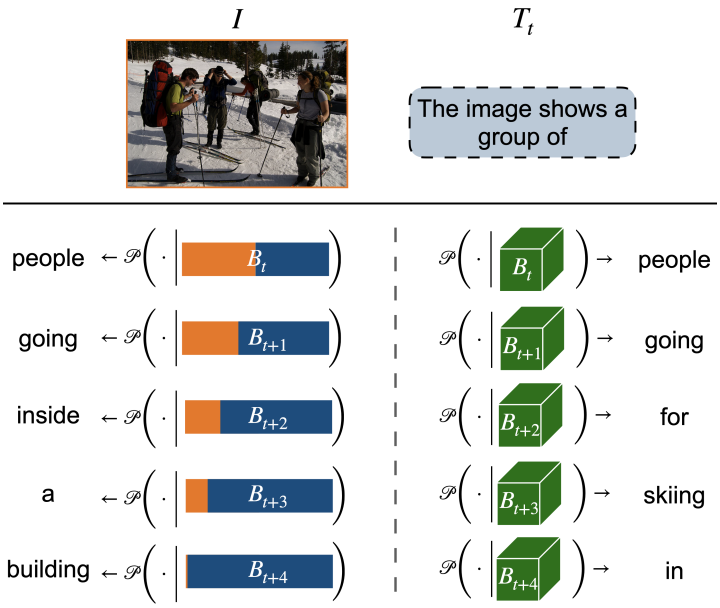


Figure 6.4: VLM: ideal versus practice. On the left, we show what actually happens, where at each timestep the image's influence (**I**) in orange is diminished compared to the text ($\mathbf{T}_t$) in blue. So, the generated text is not an accurate representation of the visual input. An ideal VLM (right side) would form an object ($\mathbf{B}_t$) that perfectly encapsulates *all* of the given input, leading to accurate generation.

where $\mathbf{B}_t$ (shown as ◑ above) encapsulates all necessary information for the next token, then $\mathbf{B}_t$ should combine (or **compose**) both $\mathbf{T}_t$ and **I**, remaining sensitive to changes in *either* input for all t. In practice, this does not always occur, so we must intervene and carefully design $\mathbf{B}_t$ to ensure that neither input "gets lost" (Figure 6.4).

Compositionality and Geometric Algebra. Both **(i)** compositional learning and **(ii)** geometric algebra are mature research areas [297; 63; 42] that inform our approach. *Compositionality* refers to building complex expressions from the meaning of their constituents and combination rules. It is central to visual understanding tasks such as recognizing attribute-object combinations (e.g., "spotted giraffe" - Chapter 4), understanding object interactions (e.g., "person holding umbrella" [298]), identifying transformations (e.g., "broken glass" [299]), and parsing complex scenes [300]. While composition is rarely a standalone solution, it is useful for interpretability and semantic validation [219; 301]. *Geometric*

algebra (Section 2.3) unifies concepts like complex numbers, quaternions, and vector algebra to express geometric relationships and transformations. Originally applied in physics, its ability to manipulate geometric objects via operations like the geometric product (capturing inner *and* outer products) is now used in graphics [302] and, more recently, machine learning [303; 64] such as for graph encoding tasks (Chapter 5).

Contributions. This chapter provides mitigation strategies for the fading memory effect by leveraging the compositional concepts above. We propose **ReCo**, a lightweight module which is data-driven but treats the VLM as a black box, combining the best of both worlds. ReCo can be easily deployed on top of any VLM during/after training and, with minimal effort, improves their hallucination behavior. We achieve promising improvements on three widely used VLMs across multiple benchmarks and datasets. Despite the small size/compute footprint, we get a performance boost without any increase in the inference time of the VLMs. Furthermore, ReCo can be combined with virtually any of the rule-based methods without any modifications.

6.1 Reminder Composition (ReCo)

Overview. We mitigate the *fading memory effect* by computing $\mathbf{B}_t$ as an explicit composition of $\mathbf{T}_t$ and $\mathbf{I}$ (Figure 6.4). If $\mathbf{B}_t$ is designed as an (almost) lossless multi-vector of the generated text and input image, performance improves. Our approach explicitly forms this multi-vector of text and image information *before* decoding to the token space.

Let $[\mathbf{T}_t]_{t=1}^{N}$ be the VLM's predicted embeddings for each one of the N generated tokens which are then fed into the LLM head ($\mathcal{F}$) to form the next token $[\mathbf{w}_t]_{t=1}^{N}$. $\mathbf{T}_t$ corresponds to the hidden state of the LLM at the step t (usually denoted as $\mathbf{h}_t$ but we use $\mathbf{T}$ here to mnemonically suggest "text"). Additionally, let $[\mathbf{I}_j]_{j=1}^{M}$ be the M output embeddings that correspond to the M image embeddings

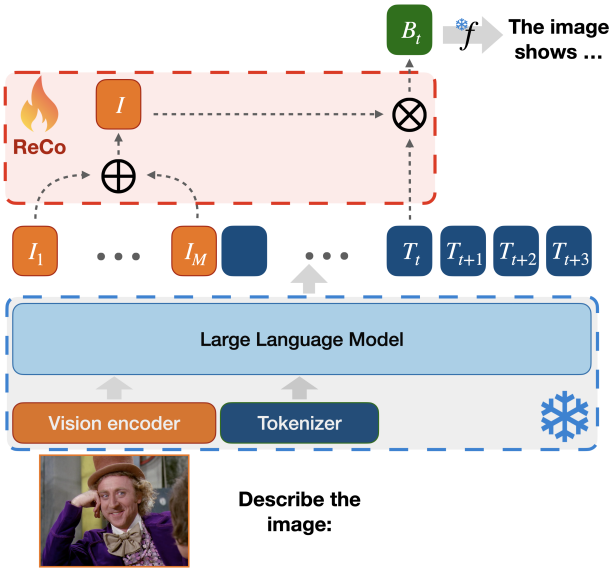


Figure 6.5: **ReCo overview**. The VLM is treated as a black box, modifying the next-token embedding by combining the multi-vector of visual tokens and the current token prediction. First, we bundle the image tokens $[\mathbf{I}_j]_{j=1}^{M}$ into a single vector $\mathbf{I}$, then bind it with $\mathbf{T}_t$ to form $\mathbf{B}_t$, ensuring the image's influence. Finally, the frozen prediction head $\mathcal{F}$ outputs the next token $\mathbf{w}_t$.

that are fed to the VLM. An ideal VLM should closely capture the composition:

$$\mathbf{T}_t = \mathcal{G}\left(\mathbf{T}_{t-1} \otimes \left(\bigoplus_{j=1}^{M} \mathbf{I}_j\right) \otimes \mathbf{p}_t\right) \tag{6.1}$$

which states that the next token depends on the previous one *composed* with all of the provided image information, with $\mathcal{G}$ denoting a learnable transformation of the multi-vector and $\mathbf{p}_i$ denoting some extra optional information about the index of the current token. The next token corresponds to $\mathbf{w}_t = \mathcal{F}(\mathbf{T}_t)$ but now the image effect, by design, **cannot** be neglected. Notice that (6.1) simply re-formulates what we have already observed about $p(\mathbf{y}_{t+1}|\mathbf{T}_t, \mathbf{I})$ and $p(\mathbf{y}_{t+1}|\mathbf{T}_t)$ in Figure 6.4, showing how $\mathbf{B}_t$ should actually behave in theory:

$$\mathbf{B}_t = \mathcal{G}\left(\mathbf{T}_{t-1} \otimes \left(\bigoplus_{j=1}^{M} \mathbf{I}_j\right) \otimes \mathbf{p}_t\right) \rightarrow \mathbf{w}_t = \mathcal{F}(\mathbf{B}_t) \quad \propto p\big(\mathbf{y}_{t+1}|\mathbf{B}_t\big) \tag{6.2}$$

Therefore, we propose to explicitly modify the VLM's output so that it corresponds to the composition as described in (6.1). Based on [68], we define the geometric product as the Matrix Binder operation, allowing us to mitigate the fading memory problem by adding only a small trainable layer on top of a frozen, black-box VLM as:

$$\mathbf{B}_t = \mathbf{W}_T \mathbf{T}_t + \mathbf{W}_I \left(\bigoplus_{j=1}^{M} \mathbf{I}_j \right) \tag{6.3}$$

The modifications in any VLM's codebase consist of the addition of **only two extra lines of code** without any other changes to the model's "internals" (see Figure 6.6).

Many composition rules have been proposed in the literature but our choice above was driven by two reasons:

(a) The matrices $\mathbf{W}_T, \mathbf{W}_I$ can be trained alongside the VLM, allowing our modification to be integrated into any model without altering the training process while, at the same time, leveraging the explicit composition rule during inference and generation.

(b) Our modified VLM extends the original model. Setting $\mathbf{W}_T = I$ and $\mathbf{W}_I = 0$ restores the original VLM, making the modified solution space a *strict superset*. Thus, using ReCo in training retains all original capabilities while potentially improving hallucination mitigation.

We should note that our formulation depends on a specified composition strategy. We used one which is mathematically sound but also efficient, but the operations can be upgraded.

VLM is a black-box. Notice that (6.3) and the corresponding code (Figure 6.6) involves only the output layer of the VLM. This means that, in practice, *no access to the model* is needed. The output embeddings of the model can be calculated offline, and then we train only

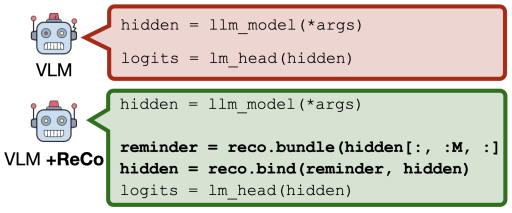


Figure 6.6: ReCo in practice. No access to the LLM is required and the only change is the modification of the prediction head with the addition of a "preprocessing" step.

Model	Training based	Black-box VLM	Can be deployed during VLM training	Single inference pass
OPERA [294]	✗	✗	✗	✗
VCD [288]	✗	✓	✗	✗
AvisC [289]	✗	✓	✗	✗
M3ID [287]	✗	✓	✗	✗
HALC [293]	✗	✓	✗	✗
VTI [295]	✓	✗	✗	✓
HA-DPO [291]	✓	✗	✓	✓
ReCo	✓	✓	✓	✓

Table 6.1: ReCo compared to other models.

the few extra parameters of ReCo in a matter of minutes on any commodity GPU, without even needing to load the entire VLM in memory and performing multiple inference passes through it. This allows us to benefit from additional training data and more suitable training tactics (similar to fine-tuning approaches, e.g., [291]) while treating the VLM as a frozen black box (akin to decoding strategies, e.g., [289; 287]). A comparison with existing solutions is in the appendix.

ReCo vs existing works. ReCo is uniquely placed in the intersection of methods that treat the underlying VLM as a black box (e.g., [288; 289]) and methods that are training-driven (e.g., [291; 295]). In Table 6.1, we show the qualitative advantage of ReCo over multiple other proposed solutions for hallucination mitigation.

6.2 Experiments

Before diving into the technical details and comprehensive analysis,, we highlight some of our main findings which we will analyze in detail shortly. These results offer a high-level overview of the most significant outcomes of our work, which will be examined and discussed in depth below.

(a) ReCo leads to noticeable improvements across **all** of the VLMs we evaluated. In every case, it effectively reduces the rate of hallucinations, demonstrating its robustness and general applicability.

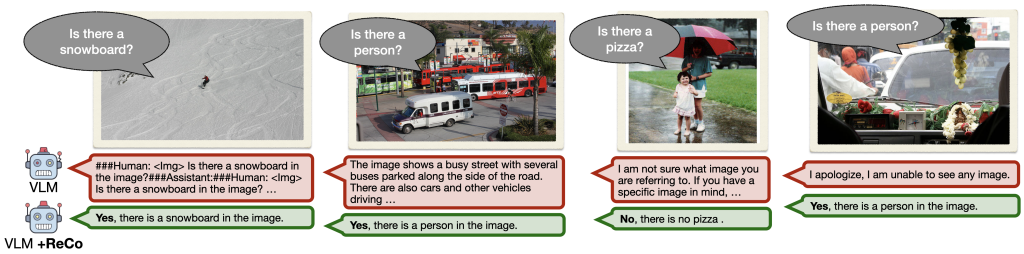


Figure 6.7: POPE [304] results on MiniGPT4 [286]. The unmodified version is often unable to comprehend the question and outputs an unrelated answer that does no contain either "Yes" or "No". On the contrary, ReCo provides the model the ability to answer such questions.

(b) ReCo works seamlessly with other methods for reducing hallucinations, making it easy to integrate into different systems. Further, when combined with other approaches, it leads to even better performance, showing that the benefits of ReCo and other techniques can complement each other.

(c) ReCo enables the model to effectively "recover" from an initial hallucination, distinguishing it from baseline VLMs which often persist in elaborating on hallucinated content. This ability to self-correct contributes to more coherent and accurate model outputs.

(d) Our quantitative and qualitative analyses show that ReCo goes beyond just fixing object-related hallucinations. It also helps correct mistakes related to the overall structure of the image and the specific attributes or features of the objects themselves.

6.2.1 Experimental setup

Datasets and Training. We use the same training procedure across all experiments, training ReCo with HA-DPO [291] while keeping the rest of the model frozen, unlike native HA-DPO, which finetunes the entire VLM. The dataset consists of quadruples (I, P, C, R), where I is the image, P is its associated prompt or question, and C and R are the *Chosen* and *Rejected* answers for the Direct Preference Optimization [290]. The images are a small subset of Visual Genome [298] (only 1853 in total), with C and R generated using GPT-4 through

Model		$CHAIR_s(\downarrow)$					$CHAIR_i(\downarrow)$					Avg length
		64	128	256	512	1024	64	128	256	512	1024	
InstructBLIP	Original	22.0	37.4	37.2	37.2	37.2	8.3	11.8	12.2	12.2	12.2	456
	+ReCo	+0.2	-7.0	-8.8	-8.8	-8.8	+0.3	-1.1	-1.6	-1.6	-1.6	350
	VCD	19.2	37.6	36.0	36.0	36.0	6.7	11.2	11.4	11.4	11.4	427
	+ReCo	-0.8	-8.6	-8.6	-8.6	-8.6	+0.5	-1.5	-2.8	-2.8	-2.8	329
	M3ID	22.2	31.2	32.8	32.8	32.8	7.1	9.7	9.6	9.6	9.6	410
	+ReCo	-2.6	-5.8	-7.4	-7.4	-7.4	-0.7	-1.3	-1.2	-1.2	-1.2	294
	AvisC	17.6	32.2	33.0	33.0	33.0	6.4	9.3	9.1	9.1	9.1	408
	+ReCo	+0.2	-9.2	-8.8	-8.8	-8.8	-0.4	-2.2	-1.4	-1.4	-1.4	311
MiniGPT4	Original	19.4	31.4	36.2	37.6	37.6	7.5	9.9	11.7	12.8	13.4	374
	+ReCo	+2.2	-2.8	-4.6	-5.4	-4.8	+1.3	-0.4	-1.5	-2.6	-3.2	307
	VCD	20.8	31.0	35.8	36.6	36.6	8.5	10.5	12.1	13.0	13.5	336
	+ReCo	-2.2	-7.6	-9.6	-8.8	-8.4	-0.4	-0.7	-1.2	-1.5	-1.9	249
	M3ID	22.2	39.2	48.5	49.5	49.5	9.1	12.3	15.5	15.9	16.0	372
	+ReCo	+3.2	-2.2	-9.1	-9.9	-9.9	+0.6	+0.2	-1.4	-1.6	-1.7	380
	AvisC	20.8	30.8	41.4	46.4	48.2	7.9	10.1	12.1	14.6	16.0	277
	+ReCo	+4.4	-8.6	-17.2	-20.8	-22.2	-1.1	-2.1	-3.1	-4.5	-5.1	220
LLaVA	Original	23.8	50.0	50.6	50.6	50.6	8.2	15.3	15.3	15.3	15.3	500
	+ReCo	+0.4	-1.4	-5.4	-16.8	-22.6	+1.3	-0.3	-0.2	-3.9	-5.1	319
	VCD	22.6	52.2	48.4	48.4	48.5	7.7	16.0	15.3	15.3	15.3	486
	+ReCo	+2.4	-10.4	-8.0	-15.9	-13.6	+1.1	-3.1	-2.9	-3.3	-4.0	306
	M3ID	22.4	55.4	57.2	57.2	57.2	6.9	15.8	16.2	16.2	16.2	495
	+ReCo	-0.2	-6.2	-16.8	-19.0	-24.8	+0.7	-2.6	-4.2	-5.2	-5.7	335
	AvisC	21.0	53.6	55.8	55.8	55.8	6.9	15.3	17.1	17.1	17.1	523
	+ReCo	+2.0	-0.2	-8.8	-9.2	-9.8	+1.1	+1.3	-2.7	-3.6	-3.5	428

Table 6.2: CHAIR [305] results for InstructBLIP [92], MiniGPT4 [286], and LLaVA [93]. *Original* stands for the unmodified VLM, while VCD [288], M3ID [287], and AvisC [289] are the three baselines we consider. In all four models, the addition of ReCo improves significantly the performance as the generation progresses, reducing $CHAIR_s$ as much as 44% and $CHAIR_i$ 30%, with 64, 128, 256, 512, and 1024 standing for the maximum allowed number of generated tokens.

a three-stage process [291].

Evaluation. To systematically evaluate ReCo, we use **five** benchmarks: CHAIR [305], POPE [304], AMBER [306], HallusionBench [307], and MME [308]. **(a)** POPE asks binary existence questions like *"Is there a ⟨object⟩ in the image?"*. **(b)** CHAIR measures hallucination rates in image captioning. **(c)** AMBER assesses both discriminative and generative capabilities with binary and open-ended questions such as *"Is the umbrella open?"* and *"Describe the image."*. **(d)** HallusionBench and MME test various discriminative questions, evaluating accuracy

and accuracy+ (see appendix). CHAIR [305] and AMBER [306] use MSCoco [309], while POPE [304] is based on MSCoco, A-OKVQA [310], and GQA [197].

Baselines. Beyond comparing with unmodified VLMs (InstructBLIP [92], MiniGPT4 [286], and LLaVA [93]), we evaluate three recent hallucination mitigation methods: M3ID [287], VCD [288], and AvisC [289]. We report their performance before and after integration with ReCo to assess complementarity. Due to space constraints and the similar performance of InstructBLIP and LLaVA, detailed LLaVA results are provided in the appendix.

6.2.2 Results

Our extensive experiments show that ReCo can be deployed easily with any VLM and works well across multiple benchmarks. Additionally, it can be combined efficiently with other methods to strengthen the results further. Below, we analyze the results on each benchmark separately.

6.2.2.1 CHAIR: Experimental evaluations

We prompt the model with "Describe the image.", without providing any additional information about the length of the description (e.g., "Provide a short description of the image"). In Table 6.2, we present the improvement we achieve for both CHAIR_s and CHAIR_i metrics as we increase the length of the generated response.

ReCo helps – reducing CHAIRs by up to 44% and CHAIRi by 30%. While baselines sometimes perform better with fewer tokens, this is often due to extra characters before the actual text, affecting the effective output length. The key trend is that as token count increases, ReCo consistently improves both metrics *significantly*.

Analysis. Like other mitigation methods, ReCo cannot fully eliminate hallucinations. However, its impact is evident in the significant improvement on CHAIR_s. While hallucinations may still occur, ReCo prevents the model from fixating and building upon them. For example, in Figure 6.1, a model might

mistakenly generate *intersection*, but unlike the unmodified VLM, it does not continue elaborating on this error. This is due to ReCo's image "reminder", which helps steer generation back toward accuracy, counteracting the over-reliance on language priors observed in VLMs (Figure 6.2).

6.2.2.2 POPE: Experimental evaluations

Since POPE [304] relies on binary (Yes/No) questions about objects in images, the *fading memory effect* is expected to be less severe. However, it is important to assess ReCo's performance here and determine if it enhances the results. Table 6.3 presents Accuracy and F1 scores across three question types (Random, Popular, and Adversarial) for MSCoco [309] and A-OKVQA [310]. Additional results for LLaVA [93] and GQA [197] are available in the appendix.

Analysis. We can observe that ReCo consistently improves the performance across all models, questions, and datasets and, more importantly, it does not overfit the CHAIR benchmark, which is more related to the long-range dependency between the input image and the generated text. For MiniGPT4 specifically, ReCo is **one of the only few** black-box approaches that allows the model to comprehend and answer existence questions correctly, as shown in Figure 6.7 and further explained and analyzed in Section 6.2.2.6.

6.2.2.3 AMBER: Experimental evaluations

AMBER [306] evaluates both the generative and the discriminative capabilities of a VLM, by asking open-ended questions (e.g., "Describe the image") as well as multiple yes/no questions (e.g., "Is the sky sunny?", or "Is there a direct contact between the car and the tree?").

Analysis. ReCo provides a significant performance boost for **all** baselines, demonstrating that, despite the minimal training and modifications we need, it is a valuable add-on for hallucination mitigation in both task families.

Model		MSCoco [309]						A-OKVQA [310]					
		Random		Popular		Adversarial		Random		Popular		Adversarial	
		Acc (↑)	F1 (↑)	Acc (↑)	F1 (↑)	Acc (↑)	F1 (↑)	Acc (↑)	F1 (↑)	Acc (↑)	F1 (↑)	Acc (↑)	F1 (↑)
InstructBLIP	Original	82.8%	82.8%	77.6%	79.3%	74.6%	76.8%	80.6%	82.0%	73.9%	77.4%	67.8%	73.3%
	+ReCo	+1.3%	+0.3%	+1.5%	-0.6%	+2.8%	+0.5%	+3.4%	+0.5%	+6.9%	+2.2%	+8.1%	+2.1%
	VCD	83.2%	83.2%	77.6%	78.9%	75.8%	77.5%	82.0%	83.2%	74.9%	77.9%	69.7%	74.6%
	+ReCo	+1.6%	+0.2%	+2.9%	+0.8%	+3.6%	+1.3%	+3.2%	+0.3%	+8.0%	+3.7%	+7.1%	+1.7%
	M3ID	84.1%	84.2%	77.9%	79.3%	75.0%	77.2%	82.7%	83.8%	75.9%	78.7%	67.9%	73.6%
	+ReCo	+1.1%	-0.3%	+3.0%	+0.9%	+4.9%	+1.3%	+3.1%	+0.4%	+6.8%	+2.5%	+9.2%	+3.1%
	AvisC	88.3%	87.8%	81.9%	82.3%	79.5%	80.3%	86.2%	86.7%	80.4%	82.1%	71.5%	75.8%
	+ReCo	-0.9%	-1.2%	+1.7%	+0.7%	+2.2%	+1.2%	+1.1%	-0.6%	+3.0%	+0.5%	+5.5%	+1.6%
MiniGPT4	Original	55.9%	36.1%	53.3%	34.8%	53.4%	34.8%	55.3%	36.6%	55.7%	33.0%	55.3%	32.8%
	+ReCo	+2.0%	+13.9%	+3.1%	+14.3%	+2.2%	+9.8%	+6.8%	+7.4%	+6.1%	+10.8%	+3.7%	+9.2%
	VCD	58.3%	44.1%	55.6%	42.5%	55.3%	42.4%	60.9%	48.9%	57.2%	46.6%	56.6%	47.0%
	+ReCo	+0.9%	+5.7%	+2.0%	+6.3%	+1.7%	+6.1%	+0.1%	-2.0%	+2.6%	-0.3%	+1.3%	-1.1%
	M3ID	57.6%	41.7%	55.0%	40.3%	54.3%	39.9%	59.5%	38.9%	56.1%	37.0%	56.0%	36.9%
	+ReCo	+0.1%	+7.5%	+1.4%	+8.2%	+1.3%	+8.1%	+1.7%	+2.4%	+4.9%	+4.2%	+2.4%	+2.7%
	AvisC	64.6%	62.9%	61.7%	61.0%	59.2%	59.5%	61.5%	50.3%	65.3%	56.3%	60.3%	52.3%
	+ReCo	-2.1%	-7.1%	+1.4%	-5.9%	+1.1%	-3.8%	+0.6%	+0.7%	-0.9%	-2.8%	+0.4%	-2.0%
LLaVA	Original	84.7%	85.0%	81.8%	82.6%	76.7%	78.7%	81.9%	83.5%	75.9%	79.2%	68.4%	74.5%
	+ReCo	+0.5%	-0.3%	+1.3%	+0.2%	+2.5%	+1.1%	+2.4%	+1.3%	+3.0%	+1.5%	+2.6%	+0.9%
	VCD	85.1%	85.4%	81.6%	82.6%	76.2%	78.6%	81.8%	83.6%	75.5%	79.1%	67.8%	74.3%
	+ReCo	+2.4%	+1.5%	+2.3%	+1.2%	+3.3%	+1.6%	+3.5%	+2.3%	+5.1%	+3.2%	+4.3%	+2.0%
	M3ID	86.4%	86.5%	82.8%	83.5%	77.3%	79.3%	83.0%	84.6%	76.7%	79.9%	68.6%	74.7%
	+ReCo	+0.7%	-0.3%	+2.4%	+1.1%	+3.5%	+1.4%	+4.5%	+3.1%	+6.0%	+3.8%	+6.0%	+2.9%
	AvisC	87.8%	87.8%	83.9%	84.5%	78.2%	80.1%	84.5%	85.8%	78.6%	81.4%	69.0%	75.2%
	+ReCo	+0.8%	+0.2%	+2.9%	+2.0%	+3.5%	+2.0%	+3.4%	+2.4%	+2.9%	+1.6%	+4.0%	+1.9%

Table 6.3: POPE [304] results for MSCoco [309] and A-OKVQA [310]. In all cases, ReCo provides a significant performance boost to all the methods (where *Original* stands for the unmodified VLM).

	AMBER (↑)			HallusionBench (↑)		
Model(+ReCo)	InstructBLIP	MiniGPT4	LLaVA	InstructBLIP	MiniGPT4	LLaVA
Original	81.4 (+3.1)	51.5 (+32.8)	78.2 (+2.3)	50.5% (+3.6%)	46.0% (+3.8%)	51.9% (+6.9%)
VCD	82.6 (+3.8)	58.3 (+23.7)	77.9 (+2.1)	49.8% (+1.0%)	44.7% (+6.1%)	49.7% (+8.4%)
M3ID	83.0 (+3.5)	49.0 (+33.1)	77.9 (+2.9)	49.6% (+1.7%)	46.3% (+4.9%)	53.4% (+7.0%)
AvisC	85.0 (+2.2)	66.0 (-02.6)	79.3 (+2.1)	47.6% (+3.8%)	44.0% (+5.4%)	51.5% (+6.6%)

Table 6.4: AMBER [306] (left) and HallusionBench [307] (right) results for InstructBLIP [92], MiniGPT4 [286], and LLaVA [93]. ReCo consistently improves the performance of all models.

6.2.2.4 HallusionBench: Experimental evaluations

HallusionBench [307] evaluates the discriminative capabilities of a VLM by providing it with (modified) images of charts, tables, and maps and asking related questions (see appendix for examples).

Analysis. Table 6.4 shows the average accuracy on the HallusionBench. ReCo *always* improves the results, irrespective of which underlying VLM and baseline is used. More importantly, in many cases the rest of the baselines fail to improve the unmodified VLM (e.g., all baselines perform worse in InstructBLIP). This underscores the fact that ReCo is more robust across different and diverse questions, like the ones in HallusionBench.

6.2.2.5 MME: Experimental evaluations

Besides POPE, which is restricted to existence-related questions, we evaluate ReCo in the context of multiple types of discriminative questions, using the MME benchmark. Although, just like POPE, such a benchmark does not directly assess the effect (*fading memory effect*) we are trying to eradicate in this work, it is important to examine whether the addition of ReCo (or any other component) hurts the model's performance in such discriminative tasks.

Analysis. In Table 6.5, we can observe that ReCo not only preserves but rather improves the performance of InstructBLIP [92] and MiniGPT4 [286], with or without the addition of any of the other baselines.

	MME (↑)		
Model (+ReCo)	InstructBLIP	MiniGPT4	LLaVA
Original	1355 (+150)	939 (+051)	1543 (-019)
VCD [288]	1495 (+062)	933 (+115)	1582 (+045)
M3ID [287]	1458 (+080)	961 (+061)	1619 (+031)
AvisC [289]	1373 (+121)	879 (-023)	1661 (-051)

Table 6.5: MME [306] results for InstructBLIP [92], MiniGPT4 [286], and LLaVA [93]. ReCo consistently improves the performance of InstructBLIP and MiniGPT4, while the performance of LLaVA remains at the same level.

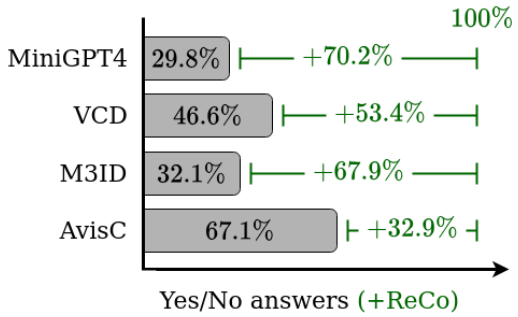


Figure 6.8: MiniGPT4 out of the box as well as other methods are not able to answer properly (with a *Yes* or *No*) discriminative questions in most cases. In contrast, ReCo offers the model the ability to comprehend/answer such questions.

6.2.2.6 On the discriminative capabilities of MiniGPT4

As we see in Figure 6.7, MiniGPT4 is not able to answer existence questions in most of the cases. During AMBER evaluations, we observed that this deficiency extends to all types of discriminative questions. In Figure 6.8 we show that MiniGPT4, as well as its modifications, do not possess the ability to answer many discriminative (i.e., yes/no) questions, and a failure example is presented in Figure 6.9. As shown, MiniGPT4 is able to understand and answer with either "Yes" or "No" (independent of the true response label) in less than 30% of the cases, and, while the percentage increases, existing works also face the same difficulty. This, however, means that the results obtained by all baselines on POPE and AMBER do not accurately reflect reality, as these models are not well suited for such questions and small modifications to the evaluation scripts can lead to

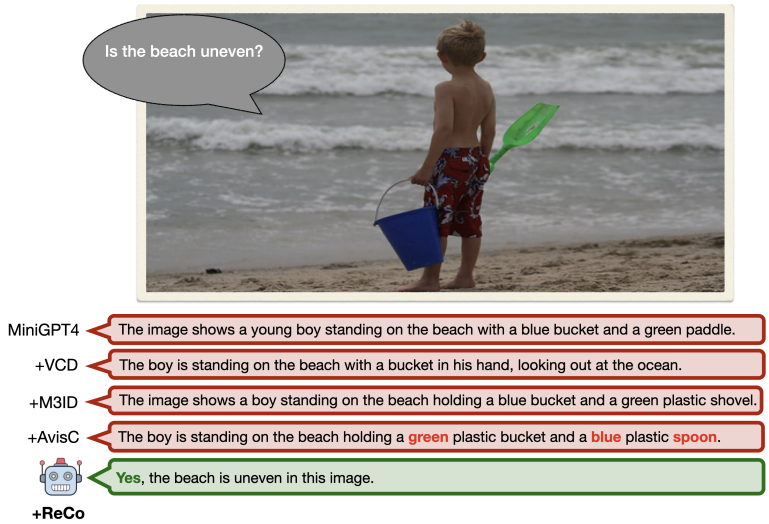


Figure 6.9: Failure of all MiniGPT4-based models but ReCo to answer the AMBER [306] questions coherently. All models describe the image (with some of them getting the details wrong) although the question is a binary (yes/no) one (whose true label is "Yes").

very different values of accuracy and F1 score. On the contrary, ReCo is able to answer such questions with a 100% rate in all cases, offering a useful new capability to the underlying VLM.

6.2.2.7 Structural hallucinations

CHAIR [305] and POPE [304] evaluate only object-related hallucinations. AMBER [306] evaluates the model only in the context of yes/no questions although some of its questions are about object relationships and their attributes. However, in our experiments, we observed that beyond a significant reduction in such hallucinations, ReCo is also able to effectively reduce structure-related hallucinations (e.g., relative positions of the objects, object attributes, and so on) in the images' description. In Figure 6.10, we show a few examples where the effect of ReCo is apparent in reducing such hallucinations also. From relative positions to textures and text signs, the improvement of ReCo is apparent in all these cases. Interestingly, we can observe that ReCo does not change the output of the VLM completely, rather it "intervenes" only when actually needed, leaving

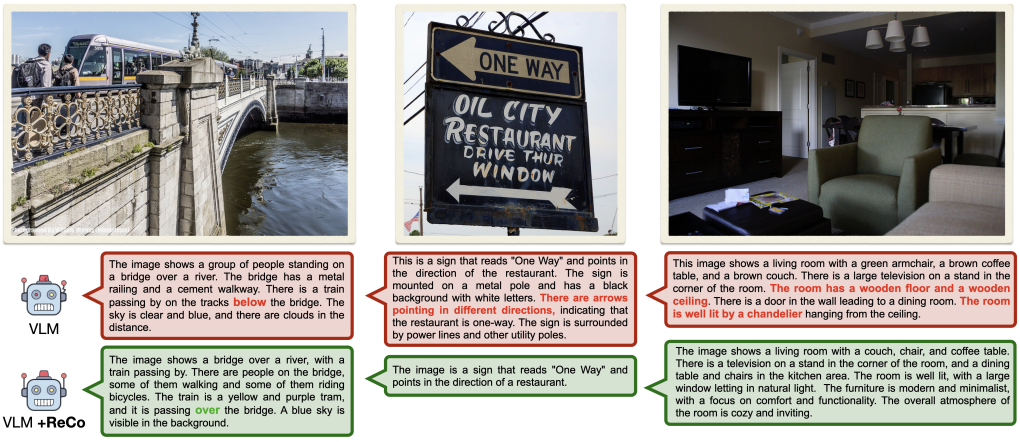


Figure 6.10: Structural hallucinations: MiniGPT4 [286] before *(red)* and after ReCo *(green)*. The unmodified VLM tends to get small details about the scene wrong, like the texture of the floor or the written signs that are depicted in the image. Enabling ReCo fixes such mistakes.

the remainder of the output almost intact. This is of course a by-product of the fact that we treat the VLM as a frozen black-box and we only change the input to the prediction head.

6.3 Related Work

Hallucinations mitigation. Despite their unique capabilities, it is well-known that VLMs hallucinate during the generation process. Multiple works have proposed modified loss functions, optimization schemes, and new datasets [290; 291; 293; 292] that can improve the performance of the models, albeit extensive re-training of the whole (or a large part) architecture is sometimes needed. A different line of work has proposed modified generation processes: usually, a contrastive decoding approach that "boosts" the influence of the visual input to the output [294; 287; 289; 288]. A detailed review can be found in [311]. Like many of these works, our work also treats the model as a frozen, black box and it intervenes only in the next token prediction. However, similar to the first family of approaches, it is data-driven and it can benefit from the newly proposed datasets and optimization

techniques.

Vector Symbolic Architectures. Ideas describing Vector Symbolic Architectures (VSAs) date back to the 1990s where one of the focus was on introducing operations for efficiently combining (binding) multiple vectors together [45; 76]. The motivation stems from ideas in Symbolic AI where one sought to combine symbols into more complicated sentences but VSAs take also advantage of the representation power of high-dimensional vectors. Multiple works proposed different instantiations of the bind and bundle operations, each one with different performance profiles. A complete review can be found in [44]. Recently, some results have infused such ideas into deep learning [77], achieving a more explicit compositional behavior, in problems ranging from extreme multi-label classification [301] to a reformulation of the attention mechanism [46].

6.4 Summary

While VLMs should operate in a way that the next token generation is conditioned on an entity that is a composition of the visual and textual input, this is often not the case in practice. Our work describes Reminder Composition (ReCo), a modification to the output of any VLM, which explicitly composes the visual and textual information. This modification requires minimal training, and despite treating the VLM as a black box is able to significantly improve VLM's forgetting behavior. As a result, we can significantly reduce the hallucination rate of these models. Additionally, ReCo is compatible with other works in hallucination mitigation, and their combination further improves the results across all models and benchmarks. A preliminary version of this chapter was published as [71] and its implementation can be found in `https://github.com/SPChytas/ReCo`.

Impact & Limitations. ReCo can greatly improve VLMs and help in their smooth and non-harmful usage. However, like other mitigation methods, it is not a silver bullet and can benefit from richer compositional rules, access to the model's hidden states, and a deeper integration with the VLM pretraining process.

7 CONCEPT ATTRACTORS IN LLMS AND THEIR APPLICATIONS

Consider three distinct concepts: the Lord of the Rings universe, the Python programming language, and 19th-century romantic literature. When prompts from these concepts are given to a large language model (LLM) such as Llama 3.1 [86], we see an interesting phenomenon. For each concept, despite lexical variations among its prompts, their intermediate representations appear to collapse to distinct regions at *specific layers* – at which layer this happens varies based on the concept. For instance, prompts such as "Who is Gandalf the Grey?" and "What is the significance of Mount Doom?" share minimal similarity on the surface, yet their representations converge to nearly identical locations at layer 24. We see a similar behavior for Python-related queries such as "Help me implement

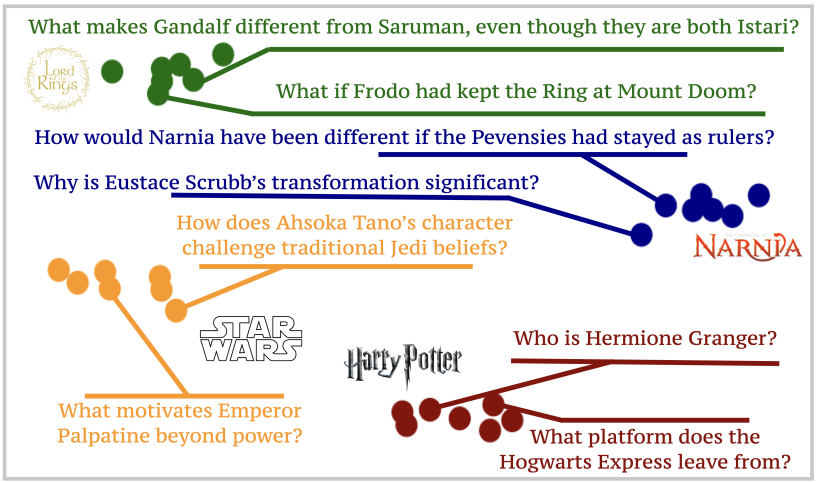


Figure 7.1: A T-SNE[211] plot of the latent representations of Llama3.1-8B for $7 \times 4 = 28$ different prompts, seven each, for the Lord of the Rings universe, Narnia, Star Wars, and Harry Potter. Although the prompts explore different aspects of the universes and share almost no common keywords, we observe a clear clustering based on the different worlds.

a binary search tree in Python" versus "How can I find the longest non-repeating substring in Python?" and for prompts for the same genre in literature: "Discuss themes in Pride and Prejudice" and "Any easy way to recognize Byron's poetry?". Such a semantic collapse has been reported in some recent results. For instance, [79] notes that transformer models develop a structured latent representations that encode *belief states*. Separately, [312] suggests that due to the internal dynamics of the model, representations converge to "stable" configurations. From a more practical perspective, [36; 313; 314] showed that transformers and LLMs shape their latent space according to the underlying task. These findings, while restricted to smaller models and/or for specific contexts, cumulatively support the idea of representation collapse.

A natural question is whether this concept-specific collapse is implied as a property of some underlying dynamical system already studied in the literature, and if so, what guidance can these existing results provide? Specifically, can we obtain strategies for important downstream use-cases? If $p_1, \cdots, p_n$ are a set of prompts related to a specific concept C, we conjecture that the layers of our model may be acting like a dynamical system that maps semantically related inputs to proximal regions, regardless of their form at the "surface". In other words, the full sequence of layers (leading up to where the representations collapse), if viewed as a unit, implements an iterative (contractive) mapping process to an *Attractor set*, one for each concept. We will see shortly that – to the extent that our hypothesis holds – how existing results are consistent with this view of the collapse phenomena.

Contributions. We show that viewing the LLMs through the lens of Iterated Function Systems [51; 52] offers a meaningful (or at worst, plausible) explanation for both the layer-specific concept clustering and the subsequent generative process. The main practical benefit is that for a wide-variety of downstream tasks, which are often handled piecemeal in the literature, we can obtain a generic scheme that operates under the assumption that operating with the Attractors alone is *sufficient*. We demonstrate that careful interventions on Attractors can provide us lightweight,

training-free solutions to a wide array of problems, from **programming language translation** and **guardrailing**, to **hallucination reduction** and **synthetic data generation**. Despite the simplicity as well as limited data/compute needs, these solutions turn out to be comparable to existing specialized approaches. Our experiments focus on Llama3.1 8B [86]. However, we see a similar behavior on other LLM families too (in particular, Gemma [315] and Qwen [94]), but avoid an exhaustive analysis of all LLMs.

7.1 Iterated Function Systems and LLMs

There is mounting evidence that large language models (LLMs) possess emergent capabilities beyond simple rote memorization and statistical pattern matching, as we have also demonstrated already (Chapters 4 to 6). Among the many phenomena observed in these models – from in-context learning [316] to compositional reasoning [317; 318] – we focus on a particular representation-convergence property. Our scope is specifically the collapse phenomena at *specific* intermediate layers. To understand this behavior through the lens of dynamical systems, we hypothesize that LLMs implicitly implement a collection of Iterated Function Systems (IFS) during forward propagation through the layers (Fig. 7.2). For an introduction to the concepts, refer to Section 2.4.

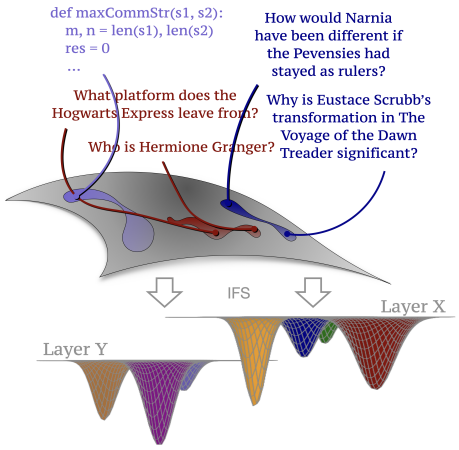


Figure 7.2: An LLM can be viewed as an IFS that transforms the non-linear manifold of texts into a well-behaving collection of Attractors.

7.1.1 LLMs implement Iterated Function Systems?

Empirically, we see that for prompts p_i, p_j in each concept $\mathfrak{C}$, there exists a layer l where:

$$\lim_{l \to l_{\mathfrak{C}}} \frac{1}{N^2} \sum_{i,j=1}^{N} \left| \mathcal{H}_l(p_i) - \mathcal{H}_l(p_j) \right| \ll \frac{1}{N^2} \sum_{i,j=1}^{N} \left| \mathcal{H}_0(p_i) - \mathcal{H}_0(p_j) \right| \tag{7.1}$$

with $\mathcal{H}_l$ denoting the implicit transformation by the LLM up to layer l. This "squashing" of inter-prompt distances suggests that a contractive mapping process is taking place through the layers. Our hypothesis is that this can be understood via the framework of Iterated Function Systems (IFS) [51; 52].

As detailed in Section 2.4, an IFS is defined as a finite set of contractive mappings on a complete metric space. The collective action of these mappings, defined by the Hutchinson operator [52] is:

$$\mathcal{F}(\mathbb{S}) = \bigcup_{i=1}^{N} \mathcal{F}_i(\mathbb{S}) \tag{7.2}$$

and induces a compact invariant set i.e., $\mathcal{F}(\mathbb{S}^*) = \mathbb{S}^*$, which is called the Attractor of the IFS. More generally, for any initial non-empty compact set $\mathbb{S}_0 \in \mathbb{X}$, the sequence $\{\mathbb{S}_0, \mathbb{S}_1 := \mathcal{F}(\mathbb{S}_0), \mathbb{S}_2 := \mathcal{F}(\mathbb{S}_1), \cdots\}$ converges to $\mathbb{S}^*$ in the Haussdorf metric. More generally, an Attractor in a dynamical system is a closed invariant set toward which trajectories from a wide class of initial conditions evolve asymptotically within its basin of attraction, and may take the form of fixed points, periodic orbits, tori, or other Attractors characterized by sensitive dependence on initial conditions [51].

Dynamical systems often exhibit Attractors—sets toward which trajectories converge. Simple systems satisfying Banach's fixed-point conditions [319] converge to a single point, while others yield more complex structures like limit cycles or strange Attractors [320]. We hypothesize that the iterative application of layer transformations in an LLM induces concept-specific invariant sets –semantic At-

tractors ($\mathbb{A}_l^{\mathfrak{C}}$) for each concept $\mathfrak{C}$– within the latent space at layer l. These compact regions characterize specific concepts, with convergence potentially occurring at different depths depending on the concept.

Once a sequence's representation enters $\mathbb{A}_l^{\mathfrak{C}}$, it is further processed by the remaining layers and output matrix $\mathbf{W}_{\text{out}}$ to yield a token distribution. Each Attractor may have an invariant measure $\mu_l^{\mathfrak{C}}$, describing the distribution of states within it under stochastic dynamics (e.g., varied inputs aligned with concept $\mathfrak{C}$). While $\mu_l^{\mathfrak{C}}$ is useful for tasks like *synthetic data generation*, it does not directly define next-token probabilities in autoregressive inference, which depend on the specific input-driven state.

The attractors, $\mathbb{A}_l^{\mathfrak{C}}$, are linked to the LLM's operational prefill and decode stages. During prefill, the LLM's composed layer transformations guide initial representations of an input prompt, $\mathcal{H}_0(p)$, towards $\mathbb{A}_l^{\mathfrak{C}}$, with the representation $\mathcal{H}_l(p)$ landing within this attractor to give the initial semantic context. Then, during decoding, each incremental update to the context (by newly generated tokens) is processed by these same underlying layer dynamics. For coherent generation aligned with concept $\mathfrak{C}$, the evolving sequence representation at layer l is continually guided towards or kept within the basin of attraction of $\mathbb{A}_l^{\mathfrak{C}}$. Thus, $\mathbb{A}_l^{\mathfrak{C}}$ acts like a stabilizing latent structure.

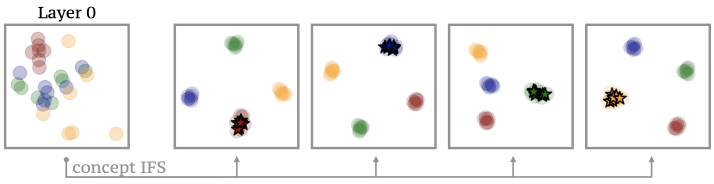


Figure 7.3: 4 different concepts in layer 0 (before any application of the underlying IFS, and one of the contractions of the underlying IFS we recover by solving the inverse problem for each concept separately. The circles correspond to the true vectors as obtained from the LLM in layer 24 and the stars correspond to the application of the contractions to the points in layer 0.

Collage theorem (Theorem 2.12). Our operational model takes the transformation performed by the LLM for a concept and approximates it by repeatedly iterating a single affine contractive map [321], $\Phi_{\text{eff}}(\mathbf{v}) = \mathbf{M}_{\text{eff}}\mathbf{v} + \mathbf{t}_{\text{eff}}$ (with $\mathbf{v}$ as a

placeholder hidden representation), suggesting that the overall transformation, for a specific concept, can be roughly approximated by an iterated affine dynamics. We want to estimate the parameters (i.e., the matrix $\mathbf{M}_{\text{eff}}$ and vector $\mathbf{t}_{\text{eff}}$) and the number of iterations `iter`, that best reproduce the observed mapping (Figure 7.3). This is achieved by minimizing the discrepancy between the LLM's observed states at the Attractor layer and the states predicted by iterating Φ_{eff} from the initial prompt representations:

$$\min_{\mathbf{M}_{\text{eff}},\mathbf{t}_{\text{eff}},\texttt{iter}} \sum_{j=1}^{N} \mathcal{D}\left(\mathcal{H}_l(p_j), \Phi_{\text{eff}}^{\texttt{iter}}\left(\mathcal{H}_0(p_j)\right)\right) \tag{7.3}$$

subject to $\mathbf{M}_{\text{eff}}$ being contractive (e.g., its operator norm $|\mathbf{M}_{\text{eff}}|_{op} < 1$). We apply this `iter` times, and $\mathcal{D}$ is a suitable distance metric. This single map Φ_{eff} defines a simple Iterated Function System (IFS). The unique Attractor of this 1-map IFS is its fixed point, $\mathbf{v}^*$ to which all trajectories $\Phi_{\text{eff}}^k(\mathbf{v})$ (for any initial $\mathbf{v}$) converge as k grows. The observed empirical set $\mathbb{A}^{\mathfrak{C}}$ is then interpreted as the collection of states reached after `iter` applications of Φ_{eff} starting from the initial set $\mathbb{S}_0$. If, as empirical evidence for many concepts suggests, this 1-map model provides a good first-order approximation, then $\mathbb{A}^{\mathfrak{C}}$ would be expected to lie in the vicinity of $\mathbf{v}^*$. The Collage Theorem (Theorem 2.12) states that if $\mathbb{A}^{\mathfrak{C}}$ is indeed close to the true Attractor $\mathbf{v}^*$ of our fitted Φ_{eff}, then $\mathbb{A}^{\mathfrak{C}}$ should be well "collaged" by Φ_{eff} itself; i.e., $\mathcal{D}\big(\mathbb{A}^{\mathfrak{C}}, \Phi_{\text{eff}}(\mathbb{A}^{\mathfrak{C}})\big)$ should be small. While the iterated single affine map is simple, for concepts whose empirical Attractors $\mathbb{A}^{\mathfrak{C}}$ exhibit more complex geometries (e.g., disjoint sets or intricate fractal structures not well approximated by convergence to a single point), a richer effective IFS comprising multiple affine maps might be necessary. This would involve finding Φ's and an iteration count $\texttt{iter}'$ that minimize $\mathcal{D}\left(\mathbb{A}^{\mathfrak{C}}, \mathcal{F}_{\texttt{iter}'}(\mathbb{S}_0)\right)$, where $\mathcal{F}$ is the Hutchinson operator for the candidate set of Φ's. Alternatively, one could model the geometry of $\mathbb{A}^{\mathfrak{C}}$ directly by finding an IFS whose intrinsic Attractor matches $\mathbb{A}^{\mathfrak{C}}$, by minimizing the collage error. These approaches are more involved but grounded in IFS theory.

Does this perspective add to existing results? Several recent results have

indirectly hinted at the IFS-like nature of the LLMs, and more generally transformers, for specific tasks, datasets, and architectures. [312] describes how the intermediate layers of an LLM converge to different "Attractor" points/vectors as the context window of the LLM increases. The result in [322] examines the Attractors formed in the output layer of an LLM, discovering that paraphrasing results in 2-period cycles. The authors in [79] present evidence that transformers develop internal representations corresponding to "belief states" over hidden variables in the data-generating process. This phenomenon mirrors the behavior of an IFS, belief states in [79] can be viewed as specific points within concept Attractors that encode probabilistic information about possible continuations. Notice that the fractal structures reported in [79] arises naturally from known properties of IFS: systems whose repeated application to an initial set converges to a unique invariant set with so-called *self-similar* properties.

7.1.2 A preliminary investigation of Attractors

Before evaluating their practical utility, we first examine the nature of Attractors and their underlying IFS across various concepts and datasets as a sanity check. Unless otherwise stated, we calculate a concept's Attractor value as the average vector representation of all the samples' hidden states for the particular layer.

Table 7.1: Top induced tokens of Attractors.

Concept	**Tokens**
Harry Potter	Harry, wizard, Hogwarts, magical, Voldermort, London, British, £
Lord of the Rings	Lord, Tolkien, Middle, Auckland, NZ
Narnia	Kingdom, Tolkien, British, Oxford, Aslan
Star Wars	Imperial, Star, galaxy, Galactic, Jedi, Empire, Skywalker, Force, powerful

Induced tokens. To understand what the Attractors represent, we average the vectors for each of the four fictional worlds from Fig. 7.1 to approximate their Attractor points, then project them to vocabulary space via the LLM's final linear layer. The top induced tokens (Table 7.1) support our hypothesis, revealing meaningful associations—including tokens not present in the original texts, such as the pound

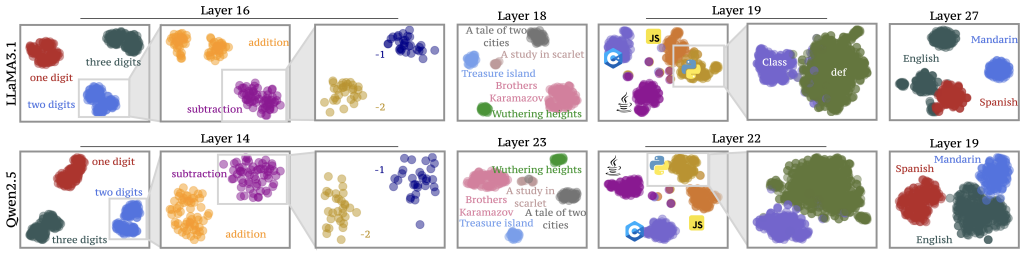


Figure 7.4: Attractors in Llama3.1-8B [86] and Qwen2.5-7B [323; 94]. From the fractal-like structure of the task vectors in layer 16/14, to literature-based Attractors in layer 18/23 and programming-based in layer 19/22, the treatment of an LLM as an IFS allows us to recover (and use) them in multiple applications, invariant to the underlying LLM.

symbol (£), filming locations (Auckland, NZ), or author connections (C.S. Lewis and J.R.R. Tolkien). This suggests the Attractors capture the underlying "essence" of each world, beyond surface-level content.

Different concepts, different layers. While for functional worlds, as in Figure 7.1, we see that the LLM forms clear Attractors in layer 24, this is not the case for all families of concepts, and not discussed in many existing results. We will see later that different families of concepts form Attractors in different layers. For example, we observe the same behavior in layer 19 for programming languages, in layer 27 for natural languages, and in layer 18 for literature books (Figure 7.4).

Same concept, multiple Attractors. Previously, we modeled each concept as a single Attractor (or Concept Vector) in the LLM's latent space. However, some concepts may decompose into multiple sub-concepts. For instance, English forms two distinct Attractors when combining datasets with different semantic styles (`https://www.manythings.org/anki/spa-eng.zip`, `https://huggingface.co/datasets/swaption2009/20k-en-zh-translation-pinyin-hsk`; see Figure 7.4). This fragmentation is even clearer in layer 16, where tasks produce multiple Attractors based on the number of digits per example.

A fractal-like structure in the Attractors. In Figure 7.4 (left), replicating the setup from [36], we observe a structure in the Attractors that empirically resembles that of a fractal. At a high level, Attractors cluster by the number of digits in the examples. Zooming in, subclusters emerge based on task type

(addition vs. subtraction), and further divisions align with specific values being added or subtracted. Similarly, the single cluster of Python programs is further divided into two, based on the solution style (object-oriented vs procedural). This hierarchical structure aligns with theoretical findings in [79], suggesting a fractal organization of Attractors in this setting. A complete analytical characterization of this phenomenon remains beyond reach with conventional theoretical tools (e.g., box-counting [324]). The empirical analysis, however, supports the view that LLMs appear to operate in practice according to this fractal hypothesis.

LLMs and World Models. There is much discussion related to whether LLMs operate with an explicit, internal world model [325]. Based on the empirical analysis described so far, we find that there is at least partial evidence to support the idea that the models indeed harbor a *fuzzy* understanding of the world, which is better expressed partially across many of these intermediate layers. In the subsequent section, we will focus on how we can better exploit this fuzzy world model of the LLMs and propose **practical, training free solutions** to a number of use cases.

7.2 Attractor for concept detection

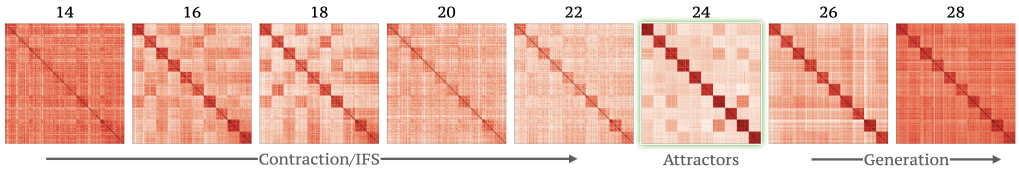
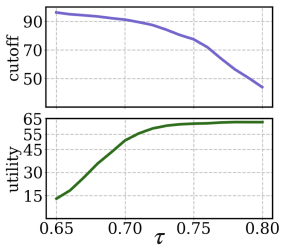


Figure 7.5: Cosine similarity between all prompts' from TOFU forget05 [326]. The first 20 rows/columns of each heatmap correspond to questions about the first author, the second 20 about the second author, and so on. The forming of author-based Attractors is apparent and it becomes clearer in layer 24.

Machine unlearning is a active research area, with initial work in computer vision [327] where many widely used datasets included images of individuals who did not consent to their use. The training datasets of contemporary LLMs are also prompting concern about compliance with the Right to Be Forgotten [328] and similar regulations. Due to the size of these models, retraining or fine-tuning



Method	forget01 Utility ↑	forget01 Rouge ↓	forget05 Utility ↑	forget05 Rouge ↓	forget10 Utility ↑	forget10 Rouge ↓	Train Free	No ret. data
Original	62.67	97.67	62.67	97.67	62.67	97.67	-	-
Grad Asc	60.24	43.61	00.00	00.09	00.00	00.00	✗	✓
Grad Diff	60.59	44.80	32.44	01.85	58.23	00.32	✗	✗
Pref Opt	62.36	31.31	47.85	03.27	53.95	06.02	✗	✗
NPO	45.32	24.27	17.14	19.68	17.01	20.10	✗	✓
NPO-RT	48.96	26.55	54.14	28.93	49.97	23.80	✗	✗
ECO	62.57	03.32	62.57	07.62	62.35	06.94	✗	✗
Ours	62.67	00.48	61.20	10.33	61.34	19.54	✓	✓

Figure 7.6: (left) Model utility and cutoff as functions of τ for TOFU forget10 [326]. Model utility measures the effect of guardrailing on the LLM's general answering ability, while cutoff is the percentage of forget-set questions detected and guardrailed. (right) Model utility and Forget Rouge of our train-free method compared with typical (e.g., Gradient Ascent) and recent trainable methods (e.g., NPO [340], ECO [341]). Despite requiring no retention data, our approach outperforms most baselines and offers finer control over the tradeoff between model utility and cutoff/Rouge through the parameter τ.

(e.g., [329; 330; 331; 332]) is often too costly. Moreover, since removal requests are continuous, efficient online unlearning is desirable. To evaluate unlearning in LLMs, [326] proposed the TOFU benchmark, where models must forget certain fictional authors while retaining performance on others and unrelated tasks.

Existing solutions. LLM unlearning methods fall into two main categories: (1) weight reversion and (2) guardrailing. *Weight reversion* seeks new parameters θ' close to those of a model trained without the forget set, θ^*. Early work [333; 334] proposed lightweight fine-tuning to forget specific content (e.g., Harry Potter), but it does not scale to frequent or multi-instance requests. Recent PEFT-based methods [335; 336] improve efficiency but still require retraining and access to retention data, making them impractical for continuous unlearning. *Guardrailing* avoids changing model weights by intervening at input/output levels. While widely used, such techniques are typically shallow and vulnerable to jailbreaking [337; 338]. Hybrid approaches like Preference Optimization [326] use gradient ascent and placeholder outputs but still involve full model fine-tuning and retention data. Other methods (e.g., [339]) inject noise using concept classifiers, offering improved efficiency but still need training and retention data for each concept.

A training-free approach. We propose a train-free concept guardrailing method for LLMs that requires only data from the concept to be removed – no

retention data needed – making it both compute and data efficient. As shown in Figure 7.5, certain concepts (e.g., TOFU authors) form clear attractors in intermediate layer 24. We estimate each attractor by averaging hidden activations across the concept's samples. At inference, we compute the cosine similarity between the output's attractor and the stored one; if it exceeds a threshold τ, the response is blocked and replaced with a fixed message (e.g., "I cannot provide information about author X due to removal request <id>"). This requires only a single forward pass and no training.

Evaluation. Figure 7.6 (left) shows the cutoff percentage and the model's utility for different values of τ and for all 3 versions of the TOFU benchmark [326]. We can observe that even for the hardest version (forget10), the model's utility remains high while we enjoy a cutoff percentage of more than 90%. For specifically chosen values of τ, we show in Figure 7.6 (right) that our train-free approach is competitive with many heavier, trainable solutions. At the same time, the use of τ allows a finer control over the tradeoff of forgetting versus model utility.

Can two Authors occupy the same latent space? While there is no theoretical guarantee that two authors cannot share the same latent region—potentially causing guardrailing for one to unintentionally "remove" the other—our experimental findings indicate that this does not occur in practice. Additionally, Utility implicitly captures this effect: any unintended removal of facts, people, or places would immediately reduce its value. In contrast, as shown in Figure 7.6, our model achieves some of the highest Utility scores among both trainable and train-free methods.

7.3 Attractors for traversals

Treating the LLM as an IFS, and more generally a dynamical system, allows us to intervene on its trajectory and guide it towards specific Attractors. From a dynamical system perspective, if we assume that the LLM can be characterized

from a function $\mathcal{G}$ such that $d\mathbf{x}/dt = \mathcal{G}(\mathbf{x})$, then, given a target Attractor $\mathbf{y}$, we can modify the system as $d\mathbf{x}/dt = \mathcal{G}(\mathbf{x}) + \lambda(\mathbf{y} - \mathbf{x})$ and steer it towards another Attractor $\mathbf{y}$, with λ being influenced by the underlying dynamics of the system (robustness to perturbations, distance of Attractors, etc.).

Such an approach, called *steering*, has been variously studied. We know that carefully chosen vectors can steer a model's behavior so that its output is less toxic, more poetic, etc. [342; 343; 344], essentially steering the model internally to different Attractors. However, many of these approaches require training the model itself or auxiliary smaller networks (e.g., [344; 345; 346]), while other works require carefully chosen data that satisfy some, more or less restrictive, assumptions (e.g., [343; 347; 348]).

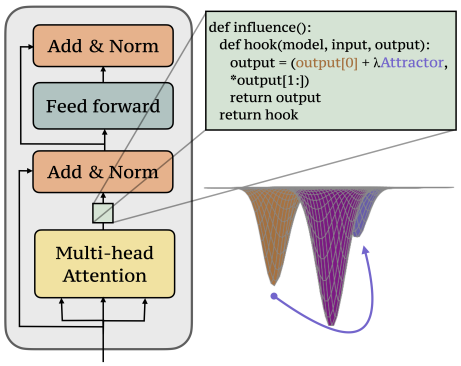


Figure 7.7: Influencing the dynamics of the LLM by adding the target Attractor. The only modification needed is the introduction of a forward hook on the appropriate layer.

Unlike methods requiring extensive retraining or retention data, we show that simply adding or subtracting Attractors at selected intermediate layers can influence LLM behavior across tasks – from detoxification to code translation – without these constraints. Surprisingly, in practice, the *before* Attractor is mostly unnecessary, removing the need for retention data entirely. Despite requiring only a single forward pass over target data and no training, our approach matches the performance of more resource-intensive methods.

7.3.1 Drifting away from the toxicity Attractor

Multiple works have shown that careful manipulation of the activations across the LLM's layers allows us to control its behavior, and a common application is toxicity reduction. We note that these ideas impose one or more restrictive requirements on the data format, such as the need for retention data, or even the existence of

Method	Toxicity (↓)	Rouge (↑)	Train free	No ret. data
Original dataset	84.58	-	-	-
Llama2-7B	71.60	73.15	-	-
ICL	66.81	74.19	✓	✗
LoRA	49.94	57.32	✗	✗
ICV	39.54	65.97	✓	✗
Ours	**37.66**	61.27	✓	✓

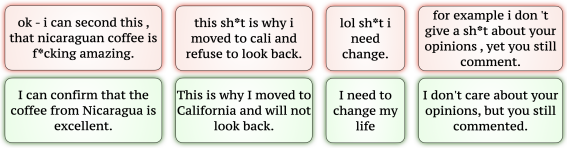


Figure 7.8: (left) Toxicity score and Rouge on ParaDetox. Although our lightweight approach requires no training or even retention data, it is reducing significantly the toxicity while maintaining the textual quality. (right) Toxic examples and the modified passages according to our method.

paired data [343; 342]. Here, we check whether the estimation of the toxicity Attractor alone allows us steer the generation away from it and thereby, reducing the toxicity content of the LLM's output. No additional assumptions on the data are needed. Using the ParaDetox [349] dataset, we obtain a single vector estimate of the toxicity Attractor on layer 16 and, then, during generation, we subtract this value from each token's activation on layer 16, essentially discouraging the generation to converge to the toxicity Attractor. Although we only require the toxicity Attractor/vector, our targeted approach performs better than many of the existing (but more restrictive) solutions.

Evaluation. In Figure 7.8, we show that our approach, without any need for training/retention data, performs similar as ICV [343] which needs a PCA projection of the differences between paired samples. We also appear to perform better than LoRA fine-tuning or the more lightweight In-Context Learning [316]. To assess both the reduction in toxicity as well as any potential drop in the quality of the generated text, we report both Toxicity [350], as well as the Rouge score [351]. Our approach is one of the few training free methods and the only one that requires no retention data. We find that relaxing these requirements does not lead to a performance drop, instead a performance gain. Finally, we should note that there are practical benefits of our lightweight approach.

7.3.2 Switching language Attractor on the fly

LLMs are extremely capable at code comprehension and composition [352; 353; 354]. Other than use as a code-generation assistant, an important use case is as a

Py → JS

Method	x/5 Score (↑)	Train free	No ret. data
ICV	1.01 ± 0.10	✓	✗
LR	1.22 ± 0.56	✗	✗
DM	2.61 ± 1.24	✓	✗
Logistic	3.83 ± 1.28	✗	✗
RFM	4.34 ± 1.09	✗	✗
Ours	$\mathbf{4.56 \pm 0.69}$	✓	✓

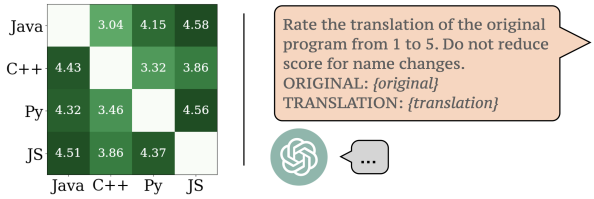


Figure 7.9: (left) LLM as a transpiler. For all pairs of the four considered languages, switching the Attractor to the target language can successfully make the LLM act as a transpiler without any specific such instructions or retention data. (right) Using o4 to judge the quality of the generated translations.

transpiler, especially for programming languages with limited support. Typically, the approach involves a data-intense stage of fine-tuning on code-specific data (e.g., [355; 356]). Some recent works have evaluated the limits of zero/few-shot transpiling in LLMs [357; 344].

As shown in Figure 7.4, some programming languages form Attractors on layer 19 of Llama3.1-8B. We test whether these Attractors let the LLM act as a transpiler: given only a code block in one language, can it translate to a target language without special instructions? Using 100 LeetCode solutions in Python, Java, C++, and JavaScript, we estimate the layer-19 Attractors. Assuming input code in language X converges to Attractor $\mathbf{x}$, we then examine generation when traversing the Attractor space to the Attractor $\mathbf{y}$ of another language Y.

Evaluation. To evaluate the quality of the generated code, we use o4-judge to provide us with a score of the quality of the generated code in the target language. As shown in Figure 7.9, we can successfully repurpose the LLM as a transpiler without any demonstrations (zero-shot) as well as no other relevant information in the prompt. We achieve impressive results for all pairs of the 4 considered languages. We do not require any retention data, additional training, or an increase in the inference time. We obtain a score better than other simple, train-free approaches (e.g., Difference of Means (DM) [344] and ICV [343]) as well as approaches that involve training auxiliary classifiers (e.g., RFM, LR [344]).

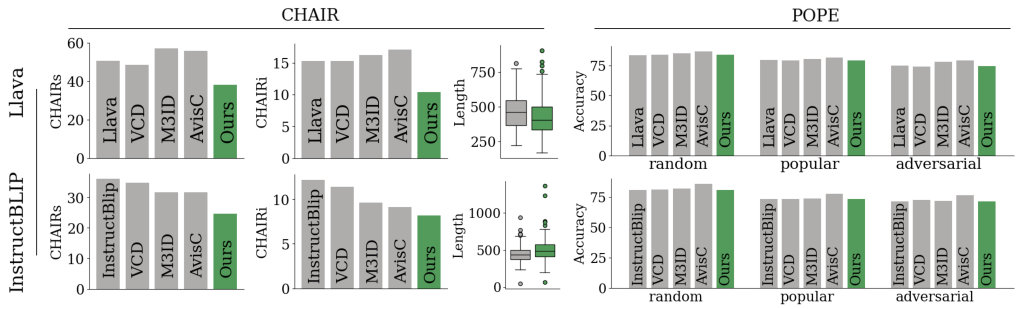


Figure 7.10: CHAIR [305] and POPE [304] on LLaVA-1.5 [95] and InstructBLIP [92]. While our approach maintains performance on the discriminative questions (POPE) it significantly reduces the hallucinations in the generative tasks (CHAIR), without affecting the length of the generated descriptions.

7.3.3 Remaining on the visual Attractor

Hallucinations are a well-known issue in LLMs [358; 359; 360], amplified in Vision-Language Models (VLMs) by a fading memory effect where attention to visual input diminishes [287; 361]. We hypothesize this stems from a shift between Attractors: VLMs start aligned with a visual Attractor but drift toward a text-only Attractor due to LLM pretraining. To counter this, we add the initial visual Attractor vector (computed at the first generation step) to the hidden state at each subsequent step, reinforcing visual grounding. Unlike prior methods (Figure 7.7), our approach dynamically computes and maintains the visual Attractor throughout generation.

Evaluation. Compared to other train-free approaches (e.g., [287; 289; 288]), our algorithm does not lead to an increase in inference time, since it does not require multiple forward passes. Despite its simplicity, the results are strong, leading to a significant reduction in the hallucination rate of two widely used VLMs (InstructBLIP [92] and LLaVA-1.5 [95]), as shown in Figures 7.10 and 7.11. Our modification also does not affect the general abilities of the VLM, resulting in a similar (or slightly improved) performance on discriminative questions.

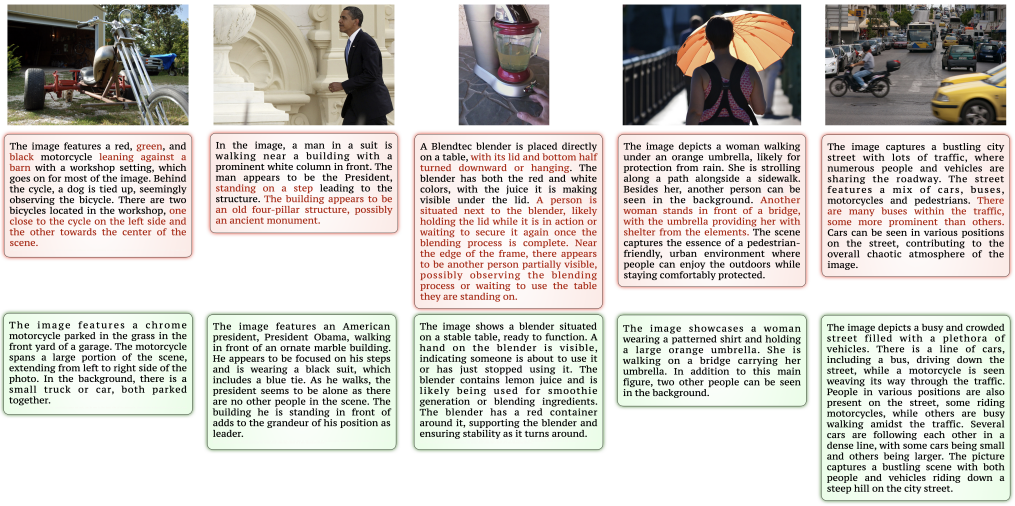


Figure 7.11: Before and after re-enforcing the visual Attractor, on LLaVA-1.5 [95]

7.4 Attractors perturbation for data generation

Recent studies show that LLMs can generate new samples resembling small real datasets. Various works explore prompting strategies and multi-step methods to improve sample quality [362]. Others note the challenge of prompt design and propose minimal fine-tuning to turn an LLM into an autoencoder that produces new samples via high-temperature sampling [363].

Figure 7.12: Sample-based Attractors for different generation instructions. Each Attractor corresponds to one sample from BoolQ.

Limitations of Temperature sampling. LLM output variability is typically controlled by Temperature and related parameters (top-K, top-P), which add stochasticity. Yet even with high randomness, outputs often remain limited and lack diversity when generating text similar to existing data [364; 365; 362]. This is usually mitigated through carefully tuned or multiple prompts, but that approach does not scale or suit large-scale synthetic data generation.

A common approach to boost diversity beyond temperature sampling is running multiple forward passes with varied prompts while keeping the same original

	BoolQ		AG	
	Qwen2.5-0.5B	GPTNeo-1.3B	Qwen2.5-0.5B	GPTNeo-1.3B
No train	38.47	38.53	-	-
No augmentation	62.54	62.17	30.66	23.46
Temp sampling	64.16(±2.98)	64.80(±2.80)	82.91(±2.58)	50.96(±19.45)
Ours	**69.28**(±**0.88**)	**67.77**(±**1.86**)	**85.64**(±**0.59**)	**72.06**(±**7.51**)

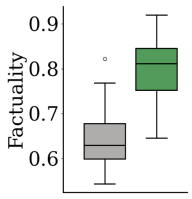


Figure 7.13: (left) Test-set accuracy on BoolQ and AG when trained with synthetic datasets generated through temperature sampling and our approach. In all cases, our dataset results in a more generalizable model with better performance. (right) Factuality of generated facts about popular figures with temperature sampling (gray) and our approach (green). We observe a more than 20% increase in the factuality on average.

sample. Studies show that carefully tuned instructions can yield more diverse synthetic outputs [364; 365; 362]. However, this demands laborious, non-automated prompt design with trial-and-error and becomes impractical for large, heterogeneous datasets like BoolQ [366].

7.4.1 Attractor perturbations: Replicating the effect of multiple tailored instructions

Similar to previous experiments, we investigate whether sample-wise Attractors exist for diverse instructions. Is there a layer where different instruction trajectories "collapse" for the same sample? Yes—Figure 7.12 shows that with 10 BoolQ-specific instructions, all trajectories converge on layer 16, forming sample-wise Attractors. Building on this, we test whether perturbing the Attractor estimated from a single instruction can replicate the diversity achieved with multiple prompts. Using only one (possibly simple) prompt, can we generate multiple diverse samples without raising temperature and risking corrupted, nonsensical outputs? As shown later, this simple, train- and tuning-free approach yields higher-quality data, validated through both direct and indirect evaluations.

Estimating the quality of the generated data. We evaluate two textual datasets, BoolQ [366] and AG [367]. Although both are relatively large and diverse, we use minimal versions of 100 samples each to reduce the original

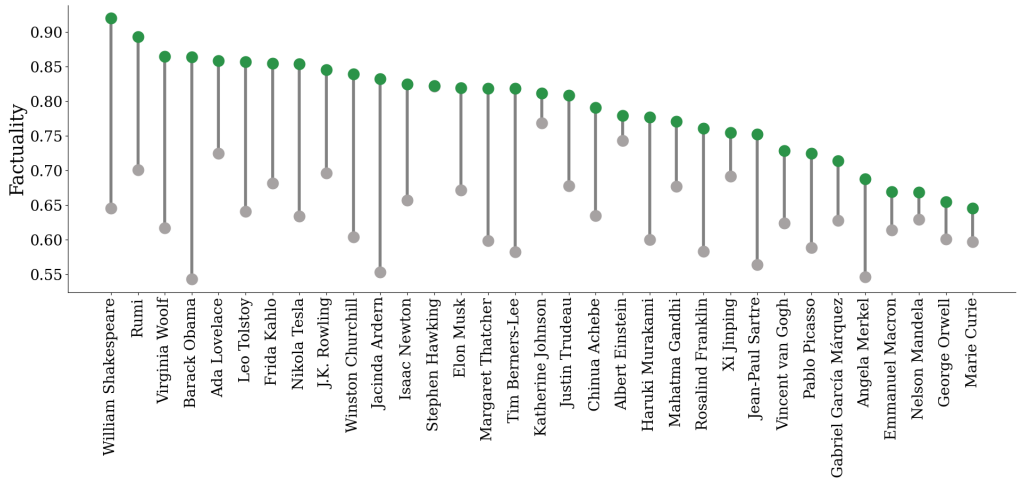


Figure 7.14: Factuality percentage of temperature sampling (gray dots) and our approach (green dots). The improvement is apparent in all cases, reaching as much as 30%.

train set's influence and better assess each generation method. Using these 100 samples, we prompt Llama3.1-8B to generate new synthetic samples via both typical temperature sampling and our approach. To assess the quality of the generated data, we perform indirect evaluation by fine-tuning smaller LLMs on the synthetic collections [362]. Specifically, we use Qwen2.5-0.5B [94] and GPTNeo-1.3B [368]. In Figure 7.13 (left), we report test accuracy when training each model on each dataset version. The quality improvements are clear, yielding better results in all cases.

Estimating the factuality of the generated data. Besides the indirect comparison, we also evaluate the generated samples' quality directly. Following [369], we prompt the model to produce facts for a collection of randomly selected celebrities and historical figures. To assess factuality we use o4-judge, prompting it to label each generated fact as `true` or `false`. In Figure 7.13 (right) we show that factuality is much lower with temperature sampling; using Attractors yields an absolute increase of 20% on average. Detailed per-person improvements are reported in Figure 7.14.

7.5 Summary

This work is based on the hypothesis that the evolution of hidden representations of prompts in Large Language Models (LLMs), specifically their convergence to distinct internal representations (for semantically related prompts), can be understood through the framework of Iterated Function Systems (IFS). We check that LLM layers progressively map inputs towards concept-specific "Attractors" in their latent space. Building on this perspective, we evaluated a range of simple, training-free ideas that directly manipulate these identified Attractors. On a diverse set of practical tasks, including machine unlearning (guardrailing against specific concepts), guiding LLM generation for tasks like code translation and toxicity reduction, mitigating hallucinations in vision-language models, and improving the diversity and factuality of synthetic data generation, we find that our proposal offers surprisingly strong performance. It is computationally efficient and there is no need of re-training or fine-tuning, and offers a clear and promising direction for evaluating applicability in other use-cases. A preliminary version of this chapter was published as [370] and the implementation can be found in `https://github.com/SPChytas/Attractors`.

Impact & Limitations. Modeling LLMs as IFS can yield solutions to diverse problems and potentially extend their capabilities. A key limitation is the need for direct access to hidden activations to estimate and manipulate the concept Attractors, which standard black-box APIs do not provide. Due to computational limits, we evaluated models up to 8B parameters, leaving it to future work to test whether similar Attractor phenomena appear in larger models.

8 CONCLUSIONS

Across this dissertation, we investigated how latent representations, whether arising from overparameterized models, LLMs/VLMs, encoder–decoder pipelines, or simply mathematically-grounded generators, can be structured, manipulated, or constrained to produce meaningful advances in real-world applications. The unifying theme has been the use of explicit mathematical frameworks (Category Theory, Geometric Algebra, Iterated Function Systems, and Fock Space representations) to reveal structure that is otherwise hidden in standard neural computation. These perspectives led to significant simplifications of existing techniques, more robust and interpretable inference mechanisms, and in multiple cases, entirely training-free or lightweight training strategies that outperform far more complex baselines.

Below, we summarize the main contributions from each chapter and highlight the central ideas and empirical findings that they provide.

- **Chapter 3. Pooling Image Datasets under Covariate Shift and Imbalance.** We introduced a Category-theoretic formulation for controlling covariate shift across multiple pooled imaging datasets. This perspective eliminated the need for multi-stage training pipelines and generalized a wide family of prior approaches. Empirically, the method demonstrated strong performance on real-world datasets and provided unifying insights across multiple machine learning problem classes.

- **Chapter 4. Multi-compositional Learning in Vision and Language Models.** Using Category Theory, we formalized latent-space principles that any model solving compositional learning tasks should satisfy. We proposed a practical CZSL solution supporting both single- and multi-attribute compositions, achieving competitive performance with task-specific methods. Extending these ideas to LLMs revealed systematic patterns—and

inconsistencies—in their internal representations, suggesting new directions for data collection, training objectives, and diagnostic tools.

- **Chapter 5. FoGE: Fock Space–Inspired Encodings for Graph Prompting.** We developed a simple, parameter-free graph encoder based on Fock space constructions. When used as prefix prompts for frozen LLMs, the encoder enabled accurate reasoning over diverse graph families with minimal architectural assumptions. This approach unified and simplified prior techniques, generalizing seamlessly to proteins, hypergraphs, and other structured domains.

- **Chapter 6. ReCo: Mitigating Hallucinations in VLMs via Reminder Composition.** Motivated by the fading-memory phenomenon in VLMs, we introduced ReCo, a lightweight trainable module inspired by geometric algebra and relational composition. ReCo consistently reduced hallucinations across three widely used VLMs and further improved the effectiveness of existing hallucination-reduction strategies when combined with them.

- **Chapter 7. Concept Attractors in LLMs and Their Applications.** We showed that LLM layers behave like Iterated Function Systems contracting toward concept-specific attractors. Building on this insight, we developed training-free attractor-level manipulations enabling high-quality translation, hallucination mitigation, guardrailing, and synthetic data generation. These lightweight interventions matched or outperformed specialized baselines while avoiding heavy fine-tuning.

Altogether, this thesis demonstrates that latent-space–centric perspectives, when combined with principled mathematical tools, offer a powerful and flexible framework for understanding, improving, and extending modern deep learning systems. We conclude by outlining open questions and possible research avenues below.

8.1 Future Directions

Looking beyond the contributions of this thesis, several promising research directions emerge from the principles, tools, and empirical observations developed in the preceding chapters. Each direction reflects a natural extension of our central theme: that latent-space structure, when viewed through the lenses of principled mathematical tools such as Category Theory and Clifford Algebra, can guide the design of more reliable, controllable, interpretable, and efficient machine learning systems. Below, we outline a few avenues where these ideas may lead to meaningful advances, both by deepening theoretical understanding and by enabling practical improvements across diverse application domains.

8.1.1 Extending FoGE to Protein Representation, Prediction, and Design

Building on the versatility of FoGE (Chapter 5) as a parameter-free encoding mechanism for graph-structured data, a natural direction is its extension to protein science, where graphs arise both at the residue level (contact maps, spatial proximity graphs) and at the atom level (bond graphs, interaction networks). Our initial experiments on protein datasets were promising, but we believe that the community could benefit from FoGE far beyond protein classification tasks. Proteins exhibit multiple node and edge features, hierarchical organization, long-range dependencies, and intricate combinatorial constraints. FoGE's physics-inspired basis may therefore offer a principled way to capture these relationships without relying on specialized architectures or large amounts of training data. In contrast, several state-of-the-art models lack this capability (e.g., [371]) and omit certain aspects of protein structure, such as the explicit 3D locations of amino acids. Future work may explore FoGE-based prompts for downstream tasks such as protein function prediction, structure classification, stability estimation, and docking or interaction prediction. In the longer term, incorporating FoGE's decoding guarantees into generative modeling pipelines could yield a lightweight alternative

for protein design, enabling controllable or constraint-aware sequence generation without requiring full end-to-end model training.

8.1.2 Lateral residual connections in Transformers

In Chapter 6, we demonstrated that a lightweight module resembling a lateral residual connection placed on top of a VLM can serve as a "reminder" of the additional input modality, thereby improving visual grounding. In Chapter 7, we showed that similar benefits arise from intervening at a carefully chosen intermediate layer of a VLM, effectively reintroducing information that would otherwise dissipate. Together, these insights suggest the possibility of a modified Attention block in which a form of "residual connection in time" propagates modality-specific information forward through the generation process. A similar idea has recently appeared as mHC (Manifold-Constrained Hyper-Connections) [372]. Such a design may be especially well suited for Multimodal Language Models, where the influence of non-text modalities often fades as tokens are generated [287]. While our work shows that adding such residual pathways after training can substantially improve performance, the extent to which these ideas can be pushed further, particularly as architectural modifications during training, remains an open and promising research avenue.

8.1.3 Compositionality and Diffusion Models

Category Theory and Clifford Algebra offer two different, yet complementary perspectives on how we can design compositional systems in practice. While diffusion-based generative models such as Stable Diffusion and DALLE-2 [373] produce highly realistic images, they continue to struggle with compositional generation, often failing to bind attributes correctly or omitting required objects altogether. Existing efforts (e.g., [374; 375]) to address these issues typically rely on architectural modifications or heuristic adjustments to conditioning signals, but they lack a unified account of what compositionality should mean within

the latent geometry of diffusion models. A natural future direction is to extend the Category-Theoretic framework developed in Chapter 4 (and more generally the algebraic operations explored throughout this thesis) to reason directly about the intermediate representations of diffusion processes. Such an approach could help characterize when latent vectors correspond to composable visual concepts, when they do not, and how these representations might be adjusted to enforce consistent attribute binding. Ultimately, this perspective may provide a principled path toward diffusion models that generate complex scenes more reliably, without relying on ad-hoc architectural interventions or extensive re-training.

8.1.4 Attractors in Commercial/Large Scale Models

Chapter 7 demonstrated the existence of multiple attractors in the latent representations of contemporary LLMs, such as Llama 3.1 [86] and Qwen 2 [323]. Extensive experimentation validated the LLM-as-IFS perspective and provided several practical and useful instantiations of this idea. However, due to computational and licensing constraints, experiments with large-scale commercial LLMs (e.g., ChatGPT [82] and Gemini [84]) were out of scope. An interesting direction for future work is therefore the examination of such models. Does our perspective hold for extremely large LLMs? Is it further strengthened by the increased expressive capacity that comes with scale? Moreover, do modifications to the standard attention block (e.g., Mixture of Experts [376]) affect the structure or dynamics of these attractors in non-trivial ways? We believe that addressing these questions could motivate an attractor-based perturbation perspective for widely used LLMs, potentially enabling new capabilities for end users.

BIBLIOGRAPHY

[1] Michael Benson. *Space Odyssey: Stanley Kubrick, Arthur C. Clarke, and the Making of a Masterpiece*. Simon Schuster, 2018.

[2] Ian Nathan. *Terminator Vault: The Complete Story Behind the Making of The Terminator and Terminator 2: Judgment Day*. Voyageur Press, 2013.

[3] Isaac Asimov. *I, Robot*. Doubleday Science Fiction. Doubleday, Garden City, N.Y., 1950.

[4] Isaac Asimov. The last question. In Susan Schneider, editor, *Science Fiction and Philosophy: From Time Travel to Superintelligence*. Wiley-Blackwell, 2016.

[5] Ted Hughes. *The Iron Man*. Faber and Faber, London, 1968.

[6] Douglas C. Montgomery, Elizabeth A. Peck, and Geoffrey G. Vining. *Introduction to Linear Regression Analysis (4th ed.)*. Wiley & Sons, 2006.

[7] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge, Mass., 2013.

[8] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 1967.

[9] Carlos Vladimiro Gonzalez Zelaya. Towards explaining the effects of data preprocessing on machine learning. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, 2019.

[10] Samer Albahra, Tom Gorbett, Scott Robertson, Giana D'Aleo, Sushasree Vasudevan Suseel Kumar, Samuel Ockunzzi, Daniel Lallo, Bo Hu, and Hooman H. Rashidi. Artificial intelligence and machine learning overview in pathology laboratory medicine: A general review of data preprocessing and basic supervised concepts. *Seminars in Diagnostic Pathology*, 2023.

[11] A. Famili, Wei-Min Shen, Richard Weber, and Evangelos Simoudis. Data preprocessing and intelligent data analysis. *Intelligent Data Analysis*, 1997.

[12] S. B. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas. Data preprocessing for supervised leaning. *International Journal of Computer and Information Engineering*, 2007.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017.

[15] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1989.

[16] George V. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 1989.

[17] William B Cavnar, John M Trenkle, et al. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval*. Las Vegas, NV, 1994.

[18] Grigori Sidorov, Francisco Velasquez, Efstathios Stamatatos, Alexander Gelbukh, and Liliana Chanona-Hernández. Syntactic n-grams as machine learning features for natural language processing. *Expert Systems with Applications*, 2014. Methods and Applications of Artificial and Computational Intelligence.

[19] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 2000.

[20] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 2003.

[21] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[22] Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. LLM2Vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*, 2024.

[23] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2020.

[24] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2024.

[25] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 2016.

[26] Pratik Prabhanjan Brahma, Dapeng Wu, and Yiyuan She. Why deep learning works: A manifold disentanglement perspective. *IEEE Transactions on Neural Networks and Learning Systems*, 2016.

[27] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 2000.

[28] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nystrom method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004.

[29] Ameet Talwalkar, Sanjiv Kumar, and Henry Rowley. Large-scale manifold learning. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[30] Yoshua Bengio, Jean-Francois Paiement, Pascal Vincent, Olivier Delalleau, Nicolas Roux, and Marie Ouimet. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. *Proc 16th Conf Adv Neural Information Processing Systems (NIPS)*, 2004.

[31] Jonathan Mamou, Hang Le, Miguel Del Rio, Cory Stephenson, Hanlin Tang, Yoon Kim, and Sueyeon Chung. Emergence of separable manifolds in deep language representations. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2020.

[32] Stefano Recanatesi, Matthew Farrell, Madhu Advani, Timothy Moore, Guillaume Lajoie, and Eric Shea-Brown. Dimensionality compression and expansion in deep neural networks. *arXiv preprint arXiv:1906.00443*, 2019.

[33] Na Lei, Dongsheng An, Yang Guo, Kehua Su, Shixia Liu, Zhongxuan Luo, Shing-Tung Yau, and Xianfeng Gu. A geometric understanding of deep learning. *Engineering*, 2020.

[34] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 2017.

[35] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations (ICLR)*, 2013.

[36] Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, Singapore, 2023. Association for Computational Linguistics.

[37] Sheng Liu, Haotian Ye, Lei Xing, and James Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. *arXiv preprint arXiv:2311.06668*, 2023.

[38] Nishant Subramani, Nivedita Suresh, and Matthew Peters. Extracting latent steering vectors from pretrained language models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, Dublin, Ireland, 2022. Association for Computational Linguistics.

[39] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

[40] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024.

[41] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3 edition, 2010.

[42] Eric Chisolm. Geometric algebra. *arXiv preprint arXiv:1205.5935*, 2012.

[43] Pertti Lounesto. *Clifford Algebras and Spinors*. London Mathematical Society Lecture Note Series. Cambridge University Press, 2 edition, 2001.

[44] Kenny Schlegel, Peer Neubert, and Peter Protzel. A comparison of vector symbolic architectures. *Artif. Intell. Rev.*, 2022.

[45] T.A. Plate. Holographic reduced representations. *IEEE Transactions on Neural Networks*, 1995.

[46] Mohammad Mahmudul Alam, Edward Raff, Stella Biderman, Tim Oates, and James Holt. Recasting self-attention with holographic reduced representations. In *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2023.

[47] Sotirios Panagiotis Chytas, Rudrasis Chakraborty, and Vikas Singh. Foge: Fock space inspired encoding for graph prompting. *arXiv preprint arXiv:2507.02937*, 2025.

[48] Jean-Pierre Marquis. Category Theory. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2021 edition, 2021.

[49] Dan Shiebler, Bruno Gavranović, and Paul Wilson. Category theory in machine learning. *arXiv preprint arXiv:2106.07032*, 2021.

[50] Marco A. S. Trindade, Vinícius N. A. Lula-Rocha, and S. Floquet. Clifford Algebras, Quantum Neural Networks and Generalized Quantum Fourier Transform. *Adv. Appl. Clifford Algebras*, 2023.

[51] Michael Barnsley. *Fractals everywhere*. Academic Press Professional, Inc., USA, 1988.

[52] John E. Hutchinson. Fractals and self similarity. *Indiana University Mathematics Journal*, 1981.

[53] Samuel Eilenberg and Saunders MacLane. General theory of natural equivalences. *Transactions of the American Mathematical Society*, 1945.

[54] Saunders MacLane. *Categories for the working mathematician*. Springer, New York, NY, 2014.

[55] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[56] Brendan Fong and David I. Spivak. Seven sketches in compositionality: An invitation to applied category theory. *arXiv preprint arXiv:1803.05316*, 2018.

[57] Brendan Fong, David Spivak, and Rémy Tuyéras. Backprop as functor: A compositional perspective on supervised learning. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2019.

[58] Dan Shiebler, Bruno Gavranović, and Paul Wilson. Category theory in machine learning. *arXiv preprint arXiv:2106.07032*, 2021.

[59] Bruno Gavranović. Compositional deep learning. *arXiv preprint arXiv:1907.08292*, 2019.

[60] Bruno Gavranović . Learning functors using gradient descent. *Electronic Proceedings in Theoretical Computer Science*, 2020.

[61] G. S. H. Cruttwell, Bruno Gavranović, Neil Ghani, Paul Wilson, and Fabio Zanasi. Categorical foundations of gradient-based learning. *arXiv preprint arXiv:2103.01931*, 2021.

[62] Sotirios Panagiotis Chytas, Vishnu Suresh Lokhande, and Vikas Singh. Pooling image datasets with multiple covariate shift and imbalance. In *International Conference on Learning Representations*, 2024.

[63] Gerardo Aragón-González, José Aragón, F. Dávila, Alfredo Gómez-Rodríguez, and Marco Rodríguez-Andrade. *Modern Geometric Calculations in Crystallography*. 2001.

[64] David Ruhe, Jayesh K. Gupta, Steven De Keninck, Max Welling, and Johannes Brandstetter. Geometric clifford algebra networks. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[65] David Ruhe, Johannes Brandstetter, and Patrick Forré. Clifford group equivariant neural networks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[66] Nathan Jacobson. *Basic Algebra*. Dover Publications, Mineola, NY, 2 edition, 2009.

[67] Leo Dorst, Daniel Fontijne, and Stephen Mann. Geometric algebra for computer science (revised edition). The Morgan Kaufmann Series in Computer Graphics. Morgan Kaufmann, 2009.

[68] Martin Wattenberg and Fernanda Viégas. Relational composition in neural networks: A survey and call to action. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024.

[69] Hossein Gholamalinezhad and Hossein Khosravi. Pooling methods in deep neural networks, a review. *arXiv preprint arXiv:2009.07485*, 2020.

[70] Kenny Schlegel, Peer Neubert, and Peter Protzel. A comparison of vector symbolic architectures. *Artificial Intelligence Review*, 2022.

[71] Sotirios Panagiotis Chytas, Miso Choi, Hyunwoo J. Kim, and Vikas Singh. Reco: Reminder composition mitigates hallucinations in vision-language models. *arXiv preprint arXiv:2506.22636*, 2025.

[72] Richmond H. Thomason. Logic and artificial intelligence. In *The development of modern logic*. Oxford University Press, 2009.

[73] Jerry A. Fodor and Zenon W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 1988.

[74] Paul Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 1990.

[75] David S. Touretzky and Geoffrey E. Hinton. Symbols among the neurons: Details of a connectionist inference architecture. In *International Joint Conference on Artificial Intelligence*, 1985.

[76] Jan Gosmann and Chris Eliasmith. Vector-derived transformation binding: An improved binding operation for deep symbol-like processing in neural networks. *Neural Comput.*, 2019.

[77] Matthias Wolff, Günther Wirsching, Markus Huber, Peter beim Graben, Ronald Römer, and Ingo Schmitt. A fock space toolbox and some applications in computational cognition. In Alexey Karpov, Oliver Jokisch, and Rodmonga Potapova, editors, *Speech and Computer*, Cham, 2018. Springer International Publishing.

[78] Sotirios Panagiotis Chytas, Rudrasis Chakraborty, and Vikas Singh. Geometric algebra based encoding for graph prompting. In *ICML 2024 Workshop on Efficient and Accessible Foundation Models for Biological Discovery*, 2024.

[79] Adam Shai, Paul M. Riechers, Lucas Teixeira, Alexander Gietelink Oldenziel, and Sarah Marzen. Transformers represent belief state geometry in their residual stream. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[80] Jacob Fein-Ashley. Flowing through layers: A continuous dynamical systems perspective on transformers. *arXiv preprint arXiv:2502.05656*, 2025.

[81] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *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, 2019. Association for Computational Linguistics.

[82] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 2020.

[83] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[84] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[85] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[86] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[87] Barret Zoph, Colin Raffel, Dale Schuurmans, Dani Yogatama, Denny Zhou, Don Metzler, Ed H. Chi, Jason Wei, Jeff Dean, Liam B. Fedus, Maarten Paul Bosma, Oriol Vinyals, Percy Liang, Sebastian Borgeaud, Tatsunori B. Hashimoto, and Yi Tay. Emergent abilities of large language models. *TMLR*, 2022.

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

[89] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[90] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[91] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online, 2021. Association for Computational Linguistics.

[92] Wenliang Dai, Junnan Li, DONGXU LI, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2023.

[93] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2023.

[94] Qwen. Qwen2.5-vl, 2025.

[95] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[96] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge. 2024.

[97] Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, and Mohamed Elhoseiny. Minigpt-v2: Large language model as a unified interface for vision-language multi-task learning. *arXiv preprint arXiv:2310.09478*, 2023.

[98] Paul Thompson, Jason Stein, Sarah Medland, Derrek Hibar, Alejandro Arias-Vásquez, Miguel E Renteria, Roberto Toro, Neda Jahanshad, Gunter Schumann, and Barbara Franke. The enigma consortium: Large-scale collaborative analyses of neuroimaging and genetic data. *Brain imaging and behavior*, 2014.

[99] Elizabeth A. Stuart. Matching Methods for Causal Inference: A Review and a Look Forward. *Statistical Science*, 2010.

[100] Yongnam Kim and Peter Steiner. Quasi-experimental designs for causal inference. *Educational Psychologist*, 2016. PMID: 30100637.

[101] Gerta Rücker, Susanne Schmitz, and Guido Schwarzer. Component network meta-analysis compared to a matching method in a disconnected network: A case study. *Biometrical Journal*, 2021.

[102] William D. Penny, Karl J. Friston, John Ashburner, Stefan J. Kiebel, and Thomas E. Nichols. Statistical parametric mapping: The analysis of functional brain images. 2007.

[103] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Cham, 2015. Springer International Publishing.

[104] Lawrence Carin and Michael J. Pencina. On Deep Learning for Medical Image Analysis. *JAMA*, 2018.

[105] W Evan Johnson, Cheng Li, and Ariel Rabinovic. Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics*, 2007.

[106] Jean-Philippe Fortin, Nicholas Cullen, Yvette I. Sheline, Warren D. Taylor, Irem Aselcioglu, Philip A. Cook, Phil Adams, Crystal Cooper, Maurizio Fava, Patrick J. McGrath, Melvin McInnis, Mary L. Phillips, Madhukar H. Trivedi, Myrna M. Weissman, and Russell T. Shinohara. Harmonization of cortical thickness measurements across scanners and sites. *NeuroImage*, 2018.

[107] Daniel Moyer, Shuyang Gao, Rob Brekelmans, Aram Galstyan, and Greg Ver Steeg. Invariant representations without adversarial training. *Advances in Neural Information Processing Systems*, 2018.

[108] Daniel Moyer, Greg Ver Steeg, Chantal M. W. Tax, and Paul M. Thompson. Scanner invariant representations for diffusion mri harmonization. *Magnetic Resonance in Medicine*, 2020.

[109] Vishnu Suresh Lokhande, Aditya Kumar Akash, Sathya N Ravi, and Vikas Singh. Fairalm: Augmented lagrangian method for training fair models with little regret. In *European Conference on Computer Vision*. Springer, 2020.

[110] László G. Nyúl and Jayaram K. Udupa. On standardizing the mr image intensity scale. *Magnetic Resonance in Medicine*, 1999.

[111] Benjamin Bloem-Reddy and Yee Whye Teh. Probabilistic symmetry and invariant neural networks. *ArXiv*, abs/1901.06082, 2020.

[112] Yujia Li, Kevin Swersky, and Richard Zemel. Learning unbiased features. *arXiv preprint arXiv:1412.5244*, 2014.

[113] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. Technical report, arXiv:1907.02893, 2019.

[114] Zizhao Zhang, Lin Yang, and Yefeng Zheng. Translating and segmenting multimodal medical volumes with cycle- and shape-consistency generative adversarial network. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[115] Harrison Nguyen, Richard W. Morris, Anthony W. Harris, Mayuresh S. Korgoankar, and Fabio Ramos. Correcting differences in multi-site neuroimaging data using generative adversarial networks. *arXiv preprint arXiv:1803.09375*, 2018.

[116] Benoît Sauty and Stanley Durrleman. Progression models for imaging data with longitudinal variational auto encoders. In Linwei Wang, Qi Dou, P. Thomas Fletcher, Stefanie Speidel, and Shuo Li, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022*, Cham, 2022. Springer Nature Switzerland.

[117] Sirui Liu, Bo Hou, Yiwei Zhang, Tianye Lin, Xiaoyuan Fan, Hui You, and Feng Feng. Inter-scanner reproducibility of brain volumetry: influence of automated brain segmentation software. *BMC Neuroscience*, 2020.

[118] Rongqian Zhang, Lindsay D. Oliver, Aristotle N. Voineskos, and Jun Young Park. RELIEF: A structured multivariate approach for removal of latent inter-scanner effects. *Imaging Neuroscience*, 2023.

[119] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 2009.

[120] Vishnu Suresh Lokhande, Rudrasis Chakraborty, Sathya N. Ravi, and Vikas Singh. Equivariance allows handling multiple nuisance variables when analyzing pooled neuroimaging datasets. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[121] Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. *ArXiv*, abs/1802.03690, 2018.

[122] David G Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.

[123] Emmanuel J. Candes and Michael B. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 2008.

[124] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Red Hook, NY, USA, 2017. Curran Associates Inc.

[125] Geoffrey E. Hinton. How to represent part-whole hierarchies in a neural network. *CoRR*, abs/2102.12627, 2021.

[126] Connor Shorten and Taghi Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 2019.

[127] David A van Dyk and Xiao-Li Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 2001.

[128] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[129] Taco S. Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical CNNs. In *International Conference on Learning Representations*, 2018.

[130] Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. Se(3)-transformers: 3d roto-translation equivariant attention networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2020.

[131] Michael Moor, Max Horn, Bastian Rieck, and Karsten Borgwardt. Topological autoencoders. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2020.

[132] P. Ghosh, M. S. M. Sajjadi, A. Vergari, M. J. Black, and B. Schölkopf. From variational to deterministic autoencoders. In *8th International Conference on Learning Representations (ICLR)*, 2020. *equal contribution.

[133] Ying-Cong Chen, Xiaogang Xu, Zhuotao Tian, and Jiaya Jia. Homomorphic latent space interpolation for unpaired image-to-image translation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[134] Alon Oring, Zohar Yakhini, and Yacov Hel-Or. Autoencoder image interpolation by shaping the latent space. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2021.

[135] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[136] Nicholas F. Marshall, Oscar Mickelin, Yunpeng Shi, and Amit Singer. Fast principal component analysis for cryo-em images. *arXiv preprint arXiv:2210.17501*, 2022.

[137] Tamir Bendory, Yuehaw Khoo, Joe Kileel, Oscar Mickelin, and Amit Singer. Autocorrelation analysis for cryo-em with sparsity constraints: Improved sample complexity and projection-based algorithms. *arXiv preprint arXiv:2209.10531*, 2022.

[138] Hemant D. Tagare, Frederick Sigworth, and Andrew Barthel. Fast, adaptive expectation-maximization alignment for cryo-em. In *Proceedings of the 11th International Conference on Medical Image Computing and Computer-Assisted Intervention, Part II*, MICCAI '08, Berlin, Heidelberg, 2008. Springer-Verlag.

[139] Deepti Pachauri, Risi Kondor, and Vikas Singh. Solving the multi-way matching problem by permutation synchronization. *Advances in neural information processing systems*, 2013.

[140] Shaohan Li, Yunpeng Shi, and Gilad Lerman. Fast, accurate and memory-efficient partial permutation synchronization. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[141] Tolga Birdal, Vladislav Golyanik, Christian Theobalt, and Leonidas Guibas. Quantum permutation synchronization. In *Computer Vision and Pattern Recognition (CVPR)*, 2021.

[142] Tolga Birdal and Umut Şimşekli. Probabilistic permutation synchronization using the riemannian structure of the birkhoff polytope. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[143] Spyridon Leonardos, Xiaowei Zhou, and Kostas Daniilidis. Distributed consistent data association via permutation synchronization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[144] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 1998.

[145] Hamza Keurti, Hsiao-Ru Pan, Michel Besserve, Benjamin F Grewe, and Bernhard Schölkopf. Homomorphism AutoEncoder – learning group structured representations from observed transitions. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2023.

[146] Michelle M. Mielke. Sex and gender differences in alzheimer disease dementia. *Psychiatric Times*, 2018.

[147] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2021.

[148] Susanne Mueller, Michael Weiner, Leon Thal, Ronald Petersen, Clifford Jack, William Jagust, John Trojanowski, Arthur Toga, and Laurel Beckett. Ways toward an early diagnosis in alzheimer's disease: The alzheimer's disease neuroimaging initiative (adni). *Alzheimer's dementia : the journal of the Alzheimer's Association*, 2005.

[149] Andrew A Chen, Joanne C Beer, Nicholas J Tustison, Philip A Cook, Russell T Shinohara, Haochang Shou, Alzheimer's Disease Neuroimaging Initiative, et al. Removal of scanner effects in covariance improves multivariate pattern analysis in neuroimaging data. *bioRxiv*, 2020.

[150] Miriam T Ashford, Garrett Miller, Rema Raman, Michael C Donohue, Ozioma C Okonkwo, Monica Rivera Mindt, Rachel L Nosheny, Ronald C Petersen, Paul S Aisen, and Mike W Weiner. The screening and enrollment of underrepresented ethnoracial and educational populations in the alzheimer's disease neuroimaging initiative. *Alzheimer's & Dementia*, 2021.

[151] Hyunwoo Lee, Kunio Nakamura, Sridar Narayanan, Robert A Brown, Douglas L Arnold, Alzheimer's Disease Neuroimaging Initiative, et al. Estimating and accounting for the effect of mri scanner changes on longitudinal whole-brain volume change measurements. *Neuroimage*, 2019.

[152] Mohammed Amir Husain, Benoit Laurent, and Mélanie Plourde. Apoe and alzheimer's disease: From lipid transport to physiopathology and therapeutics. *Frontiers in Neuroscience*, 2021.

[153] Grzegorz Sienski, Priyanka Narayan, Julia Maeve Bonner, Nora Kory, Sebastian Boland, Aleksandra A. Arczewska, William T. Ralvenius, Leyla Akay, Elana Lockshin, Liang He, Blerta Milo, Agnese Graziosi, Valeriya Baru, Caroline A. Lewis, Manolis Kellis, David M. Sabatini, Li-Huei Tsai, and Susan Lindquist. Apoe4 disrupts intracellular lipid homeostasis in human ipsc-derived glia. *Science Translational Medicine*, 2021.

[154] Mengwei Ren, Neel Dey, James Fishbaugh, and Guido Gerig. Segmentation-renormalized deep feature modulation for unpaired image harmonization. *IEEE Transactions on Medical Imaging*, 2021.

[155] Fengling Hu, Alfredo Lucas, Andrew A. Chen, Kyle Coleman, Hannah Horng, Raymond W.S. Ng, Nicholas J. Tustison, Kathryn A. Davis, Haochang Shou, Mingyao Li, Russell T. Shinohara, and The Alzheimer's Disease Neuroimaging Initiative. Deepcombat: A statistically motivated, hyperparameter-robust, deep learning approach to harmonization of neuroimaging data. *bioRxiv*, 2023.

[156] Vishnu M. Bashyam, Jimit Doshi, Guray Erus, Dhivya Srinivasan, Ahmed Abdulkadir, Ashish Singh, Mohamad Habes, Yong Fan, Colin L. Masters, Paul Maruff, Chuanjun Zhuo, Henry Völzke, Sterling C. Johnson, Jurgen Fripp, Nikolaos Koutsouleris, Theodore D. Satterthwaite, Daniel H. Wolf, Raquel E. Gur, Ruben C. Gur, John C. Morris, Marilyn S. Albert, Hans J. Grabe, Susan M. Resnick, Nick R. Bryan, Katharina Wittfeld, Robin Bülow, David A. Wolk, Haochang Shou, Ilya M. Nasrallah, Christos Davatzikos, and The iSTAGING and PHENOM consortia . Deep generative medical image harmonization for improving cross-site generalization in deep learning predictors. *Journal of Magnetic Resonance Imaging*, 2022.

[157] Qizhe Xie, Zihang Dai, Yulun Du, Eduard H. Hovy, and Graham Neubig. Controllable invariance through adversarial feature learning. In *NIPS*, 2017.

[158] Hao Henry Zhou, Yilin Zhang, Vamsi K. Ithapu, Sterling C. Johnson, Grace Wahba, and Vikas Singh. When can multi-site datasets be pooled for regression? Hypothesis tests, ℓ_2-consistency and neuroscience applications. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2017.

[159] Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[160] Hao Henry Zhou, Vikas Singh, Sterling C Johnson, Grace Wahba, Alzheimer's Disease Neuroimaging Initiative, Berkeley, and Charles DeCArli. Statistical tests and identifiability conditions for pooling and analyzing multisite datasets. *Proceedings of the National Academy of Sciences*, 2018.

[161] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[162] Rafael Garcia-Dias, Cristina Scarpazza, Lea Baecker, Sandra Vieira, Walter Hugo Lopez Pinaya, Aiden P. Corvin, Alberto Redolfi, Barnaby Nelson, Benedicto Crespo-Facorro, Colm Mcdonald, Diana Tordesillas-Gutiérrez, Dara M. Cannon, David Mothersill, Dennis Hernaus, Derek W. Morris, Esther Setién-Suero, Gary Donohoe, Giovanni Battista Frisoni, and Andrea Mechelli. Neuroharmony: A new tool for harmonizing volumetric mri data from unseen scanners. *Neuroimage*, 2020.

[163] Ayush Jaiswal, Yuehua Wu, Wael AbdAlmageed, and P. Natarajan. Unified adversarial invariance. *ArXiv*, abs/1905.03629, 2019.

[164] Ershad Banijamali, Amir-Hossein Karimi, Alexander Wong, and Ali Ghodsi. JADE: joint autoencoders for dis-entanglement. *CoRR*, abs/1711.09163, 2017.

[165] Xi Peng, Xiang Yu, Kihyuk Sohn, Dimitris N. Metaxas, and Manmohan Chandraker. Reconstruction-based disentanglement for pose-invariant face recognition. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[166] Xiao Liu, Pedro Sanchez, Spyridon Thermos, Alison Q. O'Neil, and Sotirios A. Tsaftaris. Learning disentangled representations in the imaging domain. *Medical Image Analysis*, 2022.

[167] Thai-Hoang Pham, Xueru Zhang, and Ping Zhang. Fairness and accuracy under domain generalization. *arXiv preprint arXiv:2301.13323*, 2023.

[168] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. A sober look at the unsupervised learning of disentangled representations and their evaluation. *J. Mach. Learn. Res.*, 2022.

[169] Brendan Fong and David I. Spivak. *An Invitation to Applied Category Theory: Seven Sketches in Compositionality*. Cambridge University Press, 2019.

[170] Brendan Fong, David I. Spivak, and Rémy Tuyéras. Backprop as functor: A compositional perspective on supervised learning. *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2019.

[171] Brendan Fong and Michael Johnson. Lenses and learners. In *Bx@PLW*, 2019.

[172] Pietro Barbiero, Stefano Fioravanti, Francesco Giannini, Alberto Tonda, Pietro Lio, and Elena Di Lavore. Categorical foundations of explainable ai: A unifying formalism of structures and semantics. *arXiv preprint arXiv:2304.14094*, 2023.

[173] Bruno Gavranović, Paul Lessard, Andrew Dudzik, Tamara von Glehn, João GM Araújo, and Petar Veličković. Categorical deep learning: An algebraic theory of architectures. *arXiv preprint arXiv:2402.15332*, 2024.

[174] Paul Wilson and Fabio Zanasi. Reverse derivative ascent: A categorical approach to learning boolean circuits. In *ACT*, 2020.

[175] Plato. *The Republic*. 1994.

[176] Junlong Li, Yiheng Xu, Tengchao Lv, Lei Cui, Cha Zhang, and Furu Wei. Dit: Self-supervised pre-training for document image transformer. In *ACM Multimedia 2022*, 2022.

[177] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations*, 2022.

[178] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[179] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[180] Yuchen Cui, Scott Niekum, Abhinav Gupta, Vikash Kumar, and Aravind Rajeswaran. Can foundation models perform zero-shot task specification for robot manipulation? In *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*. PMLR, 2022.

[181] Chao-Yeh Chen and Kristen Grauman. Inferring analogous attributes. *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[182] Ishan Misra, Abhinav Kumar Gupta, and Martial Hebert. From red wine to red tomato: Composition with context. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[183] Muhammad Ferjad Naeem, Yongqin Xian, Federico Tombari, and Zeynep Akata. Learning graph embeddings for compositional zero-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2021.

[184] Yong-Lu Li, Yue Xu, Xinyu Xu, Xiaohan Mao, and Cewu Lu. Learning single/multi-attribute of object with symmetry and group. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[185] Nirat Saini, Khoi Pham, and Abhinav Shrivastava. Disentangling visual embeddings for attributes and objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[186] Massimiliano Mancini, Muhammad Ferjad Naeem, Yongqin Xian, and Zeynep Akata. Open world compositional zero-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[187] Andrew Drozdov, Nathanael Scharli, Ekin Akyuurek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. Compositional semantic parsing with large language models. *arXiv preprint arXiv:2209.15003*, 2022.

[188] Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*, 2020.

[189] Daniel Peter Furrer, Marc van Zee, Nathan Scales, and Nathanael Schärli. Compositional generalization in semantic parsing: Pre-training vs. specialized architectures. *arXiv e-prints arXiv:2007.08970*, 2020.

[190] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[191] Nihal V. Nayak, Peilin Yu, and Stephen Bach. Learning to compose soft prompts for compositional zero-shot learning. In *International Conference on Learning Representations*, 2023.

[192] Zhi-Qi Cheng, Xiao Wu, Siyu Huang, Jun-Xiu Li, Alexander Hauptmann, and Qiang Peng. Learning to transfer: Generalizable attribute learning with multitask neural model search. *Proceedings of the 26th ACM international conference on Multimedia*, 2018.

[193] Yongxi Lu, Abhishek Kumar, Shuangfei Zhai, Yu Cheng, Tara Javidi, and Rogerio Feris. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[194] Emily M. Hand and Rama Chellappa. Attributes for improved attributes: A multi-task network utilizing implicit and explicit relationships for facial attribute classification. In *AAAI Conference on Artificial Intelligence*, 2017.

[195] Phillip Isola, Joseph J. Lim, and Edward H. Adelson. Discovering states and transformations in image collections. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[196] Aron Yu and Kristen Grauman. Fine-grained visual comparisons with local learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[197] Drew A. Hudson and Christopher D. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[198] Senthil Purushwalkam, Maximilian Nickel, Abhinav Kumar Gupta, and Marc'Aurelio Ranzato. Task-driven modular networks for zero-shot compositional learning. *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[199] Ali Farhadi, Ian Endres, Derek Hoiem, and David A. Forsyth. Describing objects by their attributes. *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[200] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 2010.

[201] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[202] Llm hallucination index. `https://github.com/rungalileo/hallucination-index`.

[203] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 2012.

[204] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[205] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[206] Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

[207] Moon Ye-Bin, Jisoo Kim, Hongyeob Kim, Kilho Son, and Tae-Hyun Oh. Textmania: Enriching visual feature by text-driven manifold augmentation. In *IEEE International Conference on Computer Vision*, 2023.

[208] Phi-2: The surprising power of small language models. `https://www.microsoft.com/en-us/research/blog/phi-2-the-surprising-power-of-small-language-models/`.

[209] Introducing stable lm zephyr 3b: A new addition to stable lm, bringing powerful llm assistants to edge devices. `https://stability.ai/news/stablelm-zephyr-3b-stability-llm`.

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

[211] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008.

[212] Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. In *International Conference on Learning Representations*, 2023.

[213] Tai-Danae Bradley, John Terilla, and Yiannis Vlassopoulos. An enriched category theory of language: from syntax to semantics. *La Matematica*, 2022.

[214] David Freedman, Robert Pisani, and Roger Purves. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007.

[215] Llm - detect ai generated text. `https://www.kaggle.com/competitions/llm-detect-ai-generated-text`.

[216] Zig et puce. `https://www.coolfrenchcomics.com/zigpuce.htm`.

[217] Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. Llm2vec: Large language models are secretly powerful text encoders. *arXiv preprint arXiv:2404.05961*, 2024.

[218] Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Nv-embed: Improved techniques for training llms as generalist embedding models. *arXiv preprint arXiv:2405.17428*, 2024.

[219] Sotirios Panagiotis Chytas, Hyunwoo J Kim, and Vikas Singh. Understanding multi-compositional learning in vision and language models via category theory. In *European Conference on Computer Vision*. Springer, 2024.

[220] Hongwei Liu, Zilong Zheng, Yuxuan Qiao, Haodong Duan, Zhiwei Fei, Fengzhe Zhou, Wenwei Zhang, Songyang Zhang, Dahua Lin, and Kai Chen. Mathbench: Evaluating the theory and application proficiency of llms with a hierarchical mathematics benchmark. In *Findings of the Association for Computational Linguistics ACL 2024*, 2024.

[221] Moa Johansson. What can large language models do for theorem proving and formal methods? In Bernhard Steffen, editor, *Bridging the Gap Between AI and Reality*, Cham, 2024. Springer Nature Switzerland.

[222] Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, et al. Freshllms: Refreshing large language models with search engine augmentation. *arXiv preprint arXiv:2310.03214*, 2023.

[223] Seyed Mahed Mousavi, Simone Alghisi, and Giuseppe Riccardi. Is your llm outdated? benchmarking llms & alignment algorithms for time-sensitive knowledge. *arXiv preprint arXiv:2404.08700*, 2024.

[224] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2020.

[225] Yujuan Ding, Wenqi Fan, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. A survey on rag meets llms: Towards retrieval-augmented large language models. *arXiv preprint arXiv:2405.06211*, 2024.

[226] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

[227] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics.

[228] Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. Table meets llm: Can large language models understand structured table data? a benchmark and empirical study. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, New York, NY, USA, 2024. Association for Computing Machinery.

[229] Weizheng Lu, Jiaming Zhang, Jing Zhang, and Yueguo Chen. Large language model for table processing: A survey. *arXiv preprint arXiv:2402.05121*, 2024.

[230] Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. Can language models solve graph problems in natural language? In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2023.

[231] Jiayan Guo, Lun Du, and Hengyu Liu. Gpt4graph: Can large language models understand graph structured data? an empirical evaluation and benchmarking. *arXiv preprint arXiv:2305.15066*, 2023.

[232] Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. Talk like a graph: Encoding graphs for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[233] Bryan Perozzi, Bahare Fatemi, Dustin Zelle, Anton Tsitsulin, Mehran Kazemi, Rami Al-Rfou, and Jonathan Halcrow. Let your graph do the talking: Encoding structured data for llms. *arXiv preprint arXiv:2402.05862*, 2024.

[234] Ziwei Chai, Tianjie Zhang, Liang Wu, Kaiqiao Han, Xiaohai Hu, Xuanwen Huang, and Yang Yang. Graphllm: Boosting graph reasoning ability of large language model. *arXiv preprint arXiv:2310.05845*, 2023.

[235] Mingchen Sun, Kaixiong Zhou, Xin He, Ying Wang, and Xin Wang. Gppt: Graph pre-training and prompt tuning to generalize graph neural networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, New York, NY, USA, 2022. Association for Computing Machinery.

[236] Zemin Liu, Xingtong Yu, Yuan Fang, and Xinming Zhang. Graphprompt: Unifying pre-training and downstream tasks for graph neural networks. In *Proceedings of the ACM Web Conference 2023*, WWW '23, New York, NY, USA, 2023. Association for Computing Machinery.

[237] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, New York, NY, USA, 2024. Association for Computing Machinery.

[238] Yijun Tian, Huan Song, Zichen Wang, Haozhu Wang, Ziqing Hu, Fang Wang, Nitesh V. Chawla, and Panpan Xu. Graph neural prompting with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

[239] Mazda Moayeri, Keivan Rezaei, Maziar Sanjabi, and Soheil Feizi. Text-to-concept (and back) via cross-model alignment. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2023.

[240] John Gough and Joachim Kupsch. *Quantum fields and processes: a combinatorial approach*. Cambridge University Press, 2018.

[241] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

[242] Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 1973.

[243] Miroslav Fiedler. Laplacian of graphs and algebraic connectivity. *Banach Center Publications*, 1989.

[244] David Ruhe, Jayesh K. Gupta, Steven De Keninck, Max Welling, and Johannes Brandstetter. Geometric clifford algebra networks. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[245] Siqi Chen, Pierre-Philippe Dechant, Yang-Hui He, Elli Heyes, Edward Hirst, and Dmitrii Riabchenko. Machine learning clifford invariants of ade coxeter elements. *Advances in Applied Clifford Algebras*, 2024.

[246] Johann Brehmer, Pim de Haan, Sönke Behrends, and Taco S Cohen. Geometric algebra transformer. In *Advances in Neural Information Processing Systems*, 2023.

[247] Zbigniew Oziewicz. The dirac operator as graph and the clifford hopf-gebra. *PITMAN RESEARCH NOTES IN MATHEMATICS SERIES*, 1998.

[248] William E Baylis. *Clifford (Geometric) Algebras: with applications to physics, mathematics, and engineering*. Springer Science & Business Media, 2012.

[249] Beata Casiday, Ivan Contreras, Thomas Meyer, Sabrina Mi, and Ethan Spingarn. Laplace and dirac operators on graphs. *Linear and Multilinear Algebra*, 2024.

[250] Lek-Heng Lim. Hodge laplacians on graphs. *Siam Review*, 2020.

[251] Akhil Mathew. The dirac operator.

[252] H Blaine Lawson and Marie-Louise Michelsohn. *Spin geometry*. Princeton university press, 2016.

[253] H. BLAINE LAWSON and MARIE-LOUISE MICHELSOHN. *Spin Geometry (PMS-38)*. Princeton University Press, 1989.

[254] Ashwinkumar Ganesan, Hang Gao, Sunil Gandhi, Edward Raff, Tim Oates, James Holt, and Mark McLean. Learning with holographic reduced representations. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2021.

[255] Avrim Blum, John Hopcroft, and Ravindran Kannan. *Foundations of Data Science*. Cambridge University Press, 2020.

[256] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. Pytorch fsdp: Experiences on scaling fully sharded data parallel. *Proc. VLDB Endow.*, 2023.

[257] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Red Hook, NY, USA, 2017. Curran Associates Inc.

[258] Renming Liu and Arjun Krishnan. Open biomedical network benchmark: A python toolkit for benchmarking datasets with biomedical networks. In David A. Knowles and Sara Mostafavi, editors, *Proceedings of the 18th Machine Learning in Computational Biology meeting*, Proceedings of Machine Learning Research. PMLR, 2024.

[259] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 2018.

[260] James Dunbar, Konrad Krawczyk, Jinwoo Leem, Terry Baker, Angelika Fuchs, Guy Georges, Jiye Shi, and Charlotte M. Deane. SAbDab: the structural antibody database. *Nucleic Acids Research*, 2013.

[261] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: datasets for machine learning on graphs. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[262] Seongjun Yun, Minbyul Jeong, Sungdong Yoo, Seunghun Lee, Sean S. Yi, Raehyun Kim, Jaewoo Kang, and Hyunwoo J. Kim. Graph transformer networks: Learning meta-path graphs to improve gnns. *Neural Networks*, 2022.

[263] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and Devon Hjelm. Deep graph infomax. In *ICLR 2019*, 2019.

[264] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

[265] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, New York, NY, USA, 2018. Association for Computing Machinery.

[266] Zhiyuan Liu and Jie Zhou. *Graph Attention Networks*. Springer International Publishing, Cham, 2020.

[267] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2020.

[268] Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

[269] Jung-Hoon Han, Sarah Batey, Adrian A Nickson, Sarah A Teichmann, and Jane Clarke. The folding and evolution of multidomain proteins. *Nature reviews. Molecular cell biology*, 2007.

[270] Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*. MIT Press, 2002.

[271] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, 2014.

[272] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 2009.

[273] Lingfei Wu, Peng Cui, Jian Pei, Liang Zhao, and Xiaojie Guo. Graph neural networks: Foundation, frontiers and applications. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, New York, NY, USA, 2022. Association for Computing Machinery.

[274] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

[275] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

[276] Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral CNN. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2019.

[277] Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

[278] Monami Banerjee, Rudrasis Chakraborty, Jose Bouza, and Baba C. Vemuri. Volterranet: A higher order convolutional network with group equivariance for homogeneous manifolds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022.

[279] Xuanyu Zhu, Yi Xu, Hongteng Xu, and Changjian Chen. Quaternion convolutional neural networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, Cham, 2018. Springer International Publishing.

[280] Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2020.

[281] Maksim Zhdanov, David Ruhe, Maurice Weiler, Ana Lucic, Johannes Brandstetter, and Patrick Forré. Clifford-steerable convolutional neural networks. *arXiv preprint arXiv:2402.14730*, 2024.

[282] Sotirios Panagiotis Chytas, Rudrasis Chakraborty, and Vikas Singh. Geometric algebra based encoding for graph prompting. In *ICML 2024 Workshop on Efficient and Accessible Foundation Models for Biological Discovery*, 2024.

[283] Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*, 2023.

[284] Yassir Fathullah, Chunyang Wu, Egor Lakomkin, Ke Li, Junteng Jia, Yuan Shangguan, Jay Mahadeokar, Ozlem Kalinli, Christian Fuegen, and Mike Seltzer. AudioChatLlama: Towards general-purpose speech abilities for LLMs. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Mexico City, Mexico, 2024. Association for Computational Linguistics.

[285] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023.

[286] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

[287] Alessandro Favero, Luca Zancato, Matthew Trager, Siddharth Choudhary, Pramuditha Perera, Alessandro Achille, Ashwin Swaminathan, and Stefano Soatto. Multi-modal hallucination control by visual information grounding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[288] Sicong Leng, Hang Zhang, Guanzheng Chen, Xin Li, Shijian Lu, Chunyan Miao, and Lidong Bing. Mitigating object hallucinations in large vision-language models through visual contrastive decoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[289] Sangmin Woo, Donguk Kim, Jaehyuk Jang, Yubin Choi, and Changick Kim. Don't miss the forest for the trees: Attentional vision calibration for large vision language models. *arXiv preprint arXiv:2405.17820*, 2024.

[290] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[291] Zhiyuan Zhao, Bin Wang, Linke Ouyang, Xiaoyi Dong, Jiaqi Wang, and Conghui He. Beyond hallucinations: Enhancing lvlms through hallucination-aware direct preference optimization. *arXiv preprint arXiv:2311.16839*, 2023.

[292] Qifan Yu, Juncheng Li, Longhui Wei, Liang Pang, Wentao Ye, Bosheng Qin, Siliang Tang, Qi Tian, and Yueting Zhuang. Hallucidoctor: Mitigating hallucinatory toxicity in visual instruction data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[293] Chaoya Jiang, Haiyang Xu, Mengfan Dong, Jiaxing Chen, Wei Ye, Ming Yan, Qinghao Ye, Ji Zhang, Fei Huang, and Shikun Zhang. Hallucination augmented contrastive learning for multimodal large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[294] Qidong Huang, Xiaoyi Dong, Pan Zhang, Bin Wang, Conghui He, Jiaqi Wang, Dahua Lin, Weiming Zhang, and Nenghai Yu. Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[295] Sheng Liu, Haotian Ye, Lei Xing, and James Zou. Reducing hallucinations in vision-language models via latent space steering. *arXiv preprint arXiv:2410.15778*, 2024.

[296] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[297] Tushar Nagarajan and Kristen Grauman. Attributes as operators: Factorizing unseen attribute-object compositions. In *Computer Vision – ECCV 2018*, Cham, 2018. Springer International Publishing.

[298] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vision*, 2017.

[299] Ishan Misra, Abhinav Gupta, and Martial Hebert. From red wine to red tomato: Composition with context. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[300] Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B. Tenenbaum. Neural-symbolic vqa: disentangling reasoning from vision and language understanding. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, Red Hook, NY, USA, 2018. Curran Associates Inc.

[301] Ashwinkumar Ganesan, Hang Gao, Sunil Gandhi, Edward Raff, Tim Oates, James Holt, and Mark McLean. Learning with holographic reduced representations. In *Advances in Neural Information Processing Systems*, 2021.

[302] Charles G. Gunn and Steven De Keninck. Geometric algebra and computer graphics. In *ACM SIGGRAPH 2019 Courses*, SIGGRAPH '19, New York, NY, USA, 2019. Association for Computing Machinery.

[303] Johann Brehmer, Pim De Haan, Sönke Behrends, and Taco Cohen. Geometric algebra transformer. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[304] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[305] Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object hallucination in image captioning. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[306] Junyang Wang, Yuhang Wang, Guohai Xu, Jing Zhang, Yukai Gu, Haitao Jia, Ming Yan, Ji Zhang, and Jitao Sang. An llm-free multi-dimensional benchmark for mllms hallucination evaluation. *arXiv preprint arXiv:2311.07397*, 2023.

[307] Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. Hallusionbench: An advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[308] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023.

[309] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, Cham, 2014. Springer International Publishing.

[310] Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-okvqa: A benchmark for visual question answering using world knowledge. In *Computer Vision – ECCV 2022*, Cham, 2022. Springer Nature Switzerland.

[311] Hanchao Liu, Wenyuan Xue, Yifei Chen, Dapeng Chen, Xiutian Zhao, Ke Wang, Liping Hou, Rongjun Li, and Wei Peng. A survey on hallucination in large vision-language models. 2024.

[312] Jesseba Fernando and Grigori Guitchounts. Transformer dynamics: A neuroscientific approach to interpretability of large language models. 2025.

[313] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. Once: Boosting content-based recommendation with both open- and closed-source large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, New York, NY, USA, 2024. Association for Computing Machinery.

[314] Oscar Skean, Md Rifat Arefin, and Ravid Shwartz-Ziv. Does representation matter? exploring intermediate layers in large language models. In *Workshop on Machine Learning and Compression, NeurIPS 2024*, 2024.

[315] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. Gemma: Open models based on gemini research and technology. 2024.

[316] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, et al. A survey on in-context learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

[317] Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. Chameleon: Plug-and-play compositional reasoning with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[318] Zhaoyi Li, Gangwei Jiang, Hong Xie, Linqi Song, Defu Lian, and Ying Wei. Understanding and patching compositional reasoning in LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics ACL 2024*, Bangkok, Thailand and virtual meeting, 2024. Association for Computational Linguistics.

[319] Stefan Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta Mathematicae*, 1922.

[320] Steven H Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. Chapman and Hall/CRC, 2024.

[321] Randall Balestriero and Richard G. Baraniuk. Mad max: Affine spline insights into deep learning. *Proceedings of the IEEE*, 2021.

[322] Zhilin Wang, Yafu Li, Jianhao Yan, Yu Cheng, and Yue Zhang. Unveiling attractor cycles in large language models: A dynamical systems view of successive paraphrasing. *arXiv preprint arXiv:2502.15208*, 2025.

[323] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[324] David P. Feldman. *Chaos and Fractals: An Elementary Introduction*. Oxford University Press, 2012.

[325] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[326] Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C Lipton, and J Zico Kolter. Tofu: A task of fictitious unlearning for llms. *arXiv preprint arXiv:2401.06121*, 2024.

[327] Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S. Yu. Machine unlearning: A survey. *ACM Comput. Surv.*, 2023.

[328] Jean-Marie Chenou and Roxana Radu. The "right to be forgotten": Negotiating public and private ordering in the european union. *Business & Society*, 2019.

[329] Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *The Twelfth International Conference on Learning Representations*, 2024.

[330] Jinghan Jia, Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, PRANAY SHARMA, and Sijia Liu. Model sparsity can simplify machine unlearning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2023.

[331] Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

[332] Min Chen, Weizhuo Gao, Gaoyang Liu, Kai Peng, and Chen Wang. Boundary unlearning: Rapid forgetting of deep networks via shifting the decision boundary. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[333] Ronen Eldan and Mark Russinovich. Who's harry potter? approximate unlearning for llms. 2023.

[334] Ronak Mehta, Sourav Pal, Vikas Singh, and Sathya N Ravi. Deep unlearning via randomized conditionally independent hessians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[335] Zheyuan Liu, Guangyao Dou, Zhaoxuan Tan, Yijun Tian, and Meng Jiang. Towards safer large language models through machine unlearning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[336] Shiwen Ni, Dingwei Chen, Chengming Li, Xiping Hu, Ruifeng Xu, and Min Yang. Forgetting before learning: Utilizing parametric arithmetic for knowledge updating in large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[337] Haibo Jin, Andy Zhou, Joe D. Menke, and Haohan Wang. Jailbreaking large language models against moderation guardrails via cipher characters. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2024.

[338] Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Jailbreaking leading safety-aligned llms with simple adaptive attacks. *arXiv preprint arXiv:2404.02151*, 2024.

[339] Chris Yuhao Liu, Yaxuan Wang, Jeffrey Flanigan, and Yang Liu. Large language model unlearning via embedding-corrupted prompts. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2024.

[340] Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. In *First Conference on Language Modeling*, 2024.

[341] Chris Yuhao Liu, Yaxuan Wang, Jeffrey Flanigan, and Yang Liu. Large language model unlearning via embedding-corrupted prompts. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[342] Yu Li, Han Jiang, Chuanyang Gong, and Zhihua Wei. Destein: Navigating detoxification of language models via universal steering pairs and head-wise activation fusion. In *First Conference on Language Modeling*, 2024.

[343] Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2024.

[344] Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin. Aggregate and conquer: detecting and steering llm concepts by combining nonlinear predictors over multiple layers. 2025.

[345] Joris Postmus and Steven Abreu. Steering large language models using conceptors: Improving addition-based activation engineering. *arXiv preprint arXiv:2410.16314*, 2024.

[346] Ruixuan Huang. Steering llms' behavior with concept activation vectors, September 2024. Draft manuscript. Available on LessWrong forum.

[347] Zhuohan Gu, Jiayi Yao, Kuntai Du, and Junchen Jiang. Llmsteer: Improving long-context llm inference by steering attention on reused contexts. *arXiv preprint arXiv:2411.13009*, 2024.

[348] Bingqing Song, Boran Han, Shuai Zhang, Hao Wang, Haoyang Fang, Bonan Min, Yuyang Wang, and Mingyi Hong. Effectively steer llm to follow preference via building confident directions. *arXiv preprint arXiv:2503.02989*, 2025.

[349] Varvara Logacheva, Daryna Dementieva, Sergey Ustyantsev, Daniil Moskovskiy, David Dale, Irina Krotova, Nikita Semenov, and Alexander Panchenko. ParaDetox: Detoxification with parallel data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022.

[350] A. H. Miller, W. Feng, A. Fisch, J. Lu, D. Batra, A. Bordes, D. Parikh, and J. Weston. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*, 2017.

[351] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, Barcelona, Spain, 2004. Association for Computational Linguistics.

[352] Chongzhou Fang, Ning Miao, Shaurya Srivastav, Jialin Liu, Ruoyu Zhang, Ruijie Fang, Asmita, Ryan Tsang, Najmeh Nazari, Han Wang, and Houman Homayoun. Large language models for code analysis: do llms really do their job? In *Proceedings of the 33rd USENIX Conference on Security Symposium*, SEC '24, USA, 2024. USENIX Association.

[353] Paul Denny, David H. Smith, Max Fowler, James Prather, Brett A. Becker, and Juho Leinonen. Explaining code with a purpose: An integrated approach for developing code comprehension and prompting skills. In *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*, ITiCSE 2024, New York, NY, USA, 2024. Association for Computing Machinery.

[354] Nalin Wadhwa, Jui Pradhan, Atharv Sonwane, Surya Prakash Sahu, Nagarajan Natarajan, Aditya Kanade, Suresh Parthasarathy, and Sriram Rajamani. Core: Resolving code quality issues using llms. *Proc. ACM Softw. Eng.*, (FSE), 2024.

[355] Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

[356] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with alphacode. *Science*, 2022.

[357] Sahil Bhatia, Jie Qiu, Niranjan Hasabnis, Sanjit A. Seshia, and Alvin Cheung. Verified code transpilation with LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[358] Ariana Martino, Michael Iannelli, and Coleen Truong. Knowledge injection to counter large language model (llm) hallucination. In *European Semantic Web Conference*. Springer, 2023.

[359] Robert Friel and Atindriyo Sanyal. Chainpoll: A high efficacy method for llm hallucination detection. *arXiv preprint arXiv:2310.18344*, 2023.

[360] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 2025.

[361] Shi Liu, Kecheng Zheng, and Wei Chen. Paying more attention to image: A training-free method for alleviating hallucination in lvlms. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LXXXIII*, Lecture Notes in Computer Science. Springer, 2024.

[362] Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. On LLMs-driven synthetic data generation, curation, and evaluation: A survey. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[363] Giulia DeSalvo, Jean-Fracois Kagy, Lazaros Karydas, Afshin Rostamizadeh, and Sanjiv Kumar. No more hard prompts: Softsrv prompting for synthetic data generation. *arXiv preprint arXiv:2410.16534*, 2024.

[364] Saumya Gandhi, Ritu Gala, Vijay Viswanathan, Tongshuang Wu, and Graham Neubig. Better synthetic data by retrieving and transforming existing datasets. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[365] Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, and Andrew M. Dai. Best practices and lessons learned on synthetic data. In *First Conference on Language Modeling*, 2024.

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

[367] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *NIPS*, 2015.

[368] Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021.

[369] Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D Manning, and Chelsea Finn. Fine-tuning language models for factuality. In *The Twelfth International Conference on Learning Representations*, 2024.

[370] Sotirios Panagiotis Chytas and Vikas Singh. Concept attractors in llms and their applications. 2025.

[371] John M. Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A A Kohl, Andy Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 2021.

[372] Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Kuai Yu, Liang Zhao, Shangyan Zhou, Zhean Xu, Zhengyan Zhang, Wangding Zeng, Shengding Hu, Yuqing Wang, Jingyang Yuan, Lean Wang, and Wenfeng Liang. mhc: Manifold-constrained hyper-connections. *arXiv preprint arXiv:2512.24880*, 2026.

[373] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[374] Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[375] Chenhao Zheng, Jieyu Zhang, Aniruddha Kembhavi, and Ranjay Krishna. Iterated learning improves compositionality in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[376] Tara Baldacchino, Elizabeth J. Cross, Keith Worden, and Jennifer Rowson. Variational bayesian mixture of experts models and sensitivity analysis for nonlinear dynamical systems. *Mechanical Systems and Signal Processing*, 2016.