Shared Object Networking

A Model for Knowledge Representation for use in AI Systems

Bill Weber
Cyber Foundry
bill@cyberfoundry.io

February 22, 2025

Abstract

This paper proposes **Shared Object Networking (SON)**, a framework that moves beyond the static knowledge repetition typical of current large language models (LLMs). By **decoupling core facts** (stored as well-known objects) from **inference-driven or contextual layers**, SON allows AI systems to **extend and refine** their knowledge without retraining the entire model. In this architecture, new insights or more complex inferences can be **localized** to specialized "Z-axis" layers, enabling **incremental updates** and **collaborative exploration** of novel ideas.

Whereas traditional LLMs mainly repackage existing data embedded in their parameters, SON's declarative object structures and **reputational consensus** mechanisms facilitate the **creation of fresh inferences**—such as newly recognized patterns, hypotheses, or cross-domain connections. These **modular knowledge layers** persist outside the base model, ensuring that generative systems not only recall known facts but also **synthesize genuinely new, extensible information**. By blending retrieval-augmented generation with this layered approach, SON **pushes AI beyond rote output**, offering a robust foundation for **adaptive reasoning, deeper creativity,** and **shared knowledge evolution** in next-generation AI applications.

Introduction

Recent advances in artificial intelligence (AI) have produced models that excel at a range of tasks, from language understanding to image recognition. Yet these systems often face a common stumbling block: they rarely maintain **persistent, structured knowledge** that can be reused across different contexts and over time. Without a robust, reusable foundation for encoding and retrieving information, AI models struggle with complex, interconnected domains and frequently fail to remain consistent across scenarios or adapt seamlessly to new information.

Persistent knowledge representation, particularly in the form of **object-based, declarative frameworks**, addresses this gap by providing a stable repository of facts and relationships. Rather than repeatedly reconstructing or inferring foundational knowledge, AI systems can leverage an organized knowledge base that captures entities, their attributes, and their semantic links to one another. For example, representing "Miles Davis" as an object—complete with attributes (e.g., profession, significant works) and relational links (e.g., "created by" connections to *Kind of Blue*)—ensures that this data can be reused efficiently in a variety of reasoning tasks.

This approach is especially powerful in **dynamic or evolving environments**, where models need to update or reinterpret existing knowledge without discarding what they have already learned. By separating static declarative content (such as facts about Miles Davis) from inference and adaptation layers, persistent knowledge representation allows AI systems to remain consistent in their core knowledge while flexibly responding to changing contexts and demands.

Moreover, the benefits of **persistent representation** extend to scalability: as knowledge bases grow, structured frameworks help eliminate redundancy and fragmentation, ensuring more efficient retrieval and more effective generalization.

Techniques like Retrieval-Augmented Generation (RAG) stand to gain from this modular design, which enables AI models to focus on relevant facts drawn from a robust knowledge repository.

This paper contends that persistent knowledge representation is essential for overcoming the limitations of today's AI models, both in narrowly defined tasks and in broader domains requiring deep understanding and adaptability. By grounding AI systems in stable, well-defined knowledge structures, developers can enable reasoning processes that are more transparent, consistent, and capable of evolving over time. We examine how such frameworks can be implemented with object-based declarative representations and layered architectures, ultimately paving the way for next-generation AI systems that integrate new information and contexts while retaining and building upon previously acquired knowledge.

Related Work

Contemporary AI LLM models focus on pretraining optimization, like MoE or RAG and RETRO to modify the transformers which still focus on tokenization of raw training text. This paradigm assumes that raw compute efficiencies are the primary vector and their approach attempts to reduce the angle of attack through efficiency in optimizing the output of the model.

This focuses emphasis on the proprietary nature of weighting and models to aggregate and decontextualize the world of knowledge into tokens and vectors. However, the origin and context of the information that is gathered is largely lost. This leads to the "black box" nature of LLMs where it creates answers that are sometimes accurate, sometimes not and the difference between the two is not apparent through an examination of the inner thoughts of the LLM.
In essence, it is impossible to determine the correctness or even the source of truth used in its predictions.

In some ways, the challenge being addressed here is teaching the AI to speak a language as a native versus understanding what it is that they're saying. Teaching them to repeat what others have said may make them smart or stupid, it really depends on if they guessed what to repeat correctly and without a fundamental understanding of the content.

For LLMs generating summarizations of existing text, this can be problematic. (Maynez, 2020) (Kryściński, 2019)
Their ability to maintain context and accuracy is a low confidence probability event and when errors occur, it is deeply embedded in a opaque set of training models rather than factually verifiable to source material.

This is further discussed on an explanation of stochastics and hallucinations in large language models. (Sabrina Chiesurin* Dimitris Dimakopoulos* Marco Antonio Sobrevilla Cabezudo, 2021) In their paper on Stochastic Parrots, they address the question as to if systems that merely repeat what they've heard can really represent knowledge in context and as to if that representation represents awareness of the underlying concept necessary for synthesis intrinsic to new knowledge generation.

The history of these models demonstrates how their evolution has been dependent on greedy algorithms display emergent behavior as technology overcomes the explosive growth they require in parameter sizes. (Marcus Gary) In their book they discuss if LLMs lack the underlying conceptual, casual, and common sense reasoning capacities that characterize intelligence. They see hallucinations as a symptom of this shortcoming because LLMs do not build grounded mental models of the world, they cannot reasonably distinguish fact from fiction. In this context, hallucinations are a fundamental barrier to trusting LLMs and by extension reaching AGI.

Recent work on Retrieval-Augmented Generation (RAG) begins to address some of these issues, although with the same limitations on contextual awareness. (Lewis, 2021) Even the recent DeepMind optimization for training relies on trillons of tokens for RETROfit. (Borgeaud, 2022)

Methodology

The hypothesis of this paper is largely broken down into two domains. First is the applicability of RAG in reducing hallucinations in LLMs is well documented. The applicability of this element of the hypothesis is considered proven. However, what is not yet integrated is the applicability of large knowledge graphs as a meaningful substitute for the knowledge base that RAG would rely on.

As a development of this hypothesis, existing work on the creation of knowledge graphs would be expanded to include open frameworks for the creation of Shared Object Networks (SON) on which inference models could be generated. For example, here are some potential use cases that could contribute to the framework.

Public Object Mining: Tools that would look for specific kinds of objects and create them within an IPFS context. Examples could include identifying people, places or other real world objects that could be mapped to their subject object types (e.g. Person, Place) and then stored for future reference.

Public Object Linking: Tools that would look for basic connections between objects, like hierarchies, or relationships and create meanings between these objects on dedicated layers of contextual knowledge. Examples could include people's cities of birth, death or current known location. These different linkage types would represent independent layers of knowledge that could be referenced in future searches.

Private Data Objectification: Tools could exist that would internally objectify actions. For example, a company could create specific linkages between a person and the purchase of an item that they sold. This would represent a private transaction z-axis layer that could then be used in further analysis, but would be limited to their internal use and yet reference external or shared objects.

Semantics and Sentiments: Tools could be created that attribute people to sentiments. While this information is not as concrete as a transaction, it could be created as a confidence interval based on current demographic modeling principals. This could be based on affinity, correlation, past actions, or other indicators. A typical example of this would be assigning confidence intervals to sentiments based on purchase history or public data sets.

These examples create opportunity for n-tier vector analysis of increasing data sets across domains and functions which would be useful in inference models. As the basis for moving LLMs past that of a stochastic parrot to meaningful data analysis, these models also would allow for increased model awareness purely on the basis of expansion of the Shared Object Network and not the underlying language model. In this way, the LLM becomes part of a multi-modal inference engine rather than carrying the entire load of the AGI.

This type of base learning or fundamental understanding with persistence, extensibility and inherent ability to pattern find would considerably improve the performance of predictive models in a fundamental advancement toward AGI.

Results & Discussion

Declarative Object Representations for Knowledge

In the design of intelligent systems, the representation of knowledge as objects and their declarative relationships provides a foundational framework for reasoning and retrieval. Objects are defined as independent entities with intrinsic attributes and a type that categorizes their role or nature within the system. For example, *Miles Davis* can be represented as an object of type "Person," with attributes such as his name, profession, and lifespan. Similarly, concepts like "Musician" or "Jazz" are also objects, each with their own attributes and definitions. This modular approach ensures that each object is self-contained, reusable, and independent of specific contexts or instances.

The declarative layer explicitly encodes the relationships between these objects, describing the purpose or context of each link. For instance, *Miles Davis* might be linked to *Kind of Blue* through a "created by" relationship, while *Kind of Blue* is linked to "Modal Jazz" through a "belongs to genre" relationship. These relationships are not merely connections but are semantically enriched with meaning that defines how objects interact or relate to one another. By separating the definition of objects from their relationships, this approach creates a lightweight and scalable model that can adapt to diverse domains. Mathematically, these objects and relationships can be represented in a vector space, where each object is encoded as a vector and relationships are transformations or mappings within this space. This allows the system to capture semantic meaning geometrically—proximity between vectors reflects similarity, while transformations encode relational dynamics. Such representations enable efficient reasoning, retrieval, and inference in knowledge-based systems.

By focusing on declarative knowledge as a static foundation of facts and relationships, this framework sets the stage for more advanced layers of reasoning and contextual adaptation. These additional layers can build upon the declarative base to introduce dynamic inference mechanisms or contextual relevance without overloading the core model. This separation ensures computational efficiency while maintaining flexibility for integration into long-term systems like Retrieval-Augmented Generation (RAG), where modularity and scalability are critical for managing complex knowledge domains.

Declarative knowledge, as a structured representation of facts and relationships, provides a foundation for organizing and reasoning about information in a knowledge system. Objects within this framework are defined independently, each with intrinsic attributes and classifications that describe their nature or role. For example, *Miles Davis* can be represented as an object of type "Person," with attributes such as name, profession, and lifespan. Similarly, *Musician* is an object that encapsulates the concept of performing or composing music, while works like *Kind of Blue* or *Bitches Brew* are treated as independent objects of type "Musical Album." Declarative relationships explicitly link these objects, describing the purpose or context of the connection—such as "Miles Davis created *Kind of Blue*" or "*Kind of Blue* belongs to the genre Modal Jazz."

This framework aligns with structural knowledge representation principles, where relationships between objects are formalized as predicates that describe their roles and interactions. For instance, binary relations like "created by" or "belongs to" connect specific entities, while hierarchical structures enable inheritance of properties across related objects. Declarative knowledge forms the descriptive core of a system, capturing what phenomena exist and how they relate to one another without prescribing actions or procedures.

Mathematically, this structure can be represented in a vector space, where objects are encoded as vectors and relationships are transformations or mappings within this space. This allows semantic meaning to be captured geometrically—proximity between vectors reflects similarity, while transformations encode relational dynamics. By organizing knowledge into declarative layers that emphasize modularity and explicit connections, we create a robust foundation for scalable systems. These systems can later incorporate inference mechanisms and contextual reasoning layers to support advanced applications like Retrieval-Augmented Generation (RAG), where structural knowledge enables efficient retrieval and reasoning across diverse domains.

Here are examples of objects and their types related to Miles Davis, his work, and the type of music he is associated with. These objects are defined independently, with declarative relationships describing their roles and connections.

1. *Core Object: Miles Davis*
    - Type: Person
    - Attributes:
    - Name: Miles Dewey Davis III
    - Birthdate: May 26, 1926
    - Death: September 28, 1991
    - Profession: Musician (Trumpeter, Composer, Bandleader)
    - Declarative Relationships:
    - Is a: Musician
    - Associated With: Jazz
    - Pioneer Of: Cool Jazz, Modal Jazz, Jazz Fusion

2. Related Object: Musician
    - Type: Role/Profession
    - Attributes:
    - Description: A person who performs, composes, or leads musical works.
    - Declarative Relationships:
    - Associated With: Genres (e.g., Jazz, Cool Jazz, Hard Bop)

3. Related Objects: Genres

Genres are independent objects that describe the styles of music Miles Davis is associated with.

Object Name	**Type**	**Attribute**	**Relationship to Miles Davis**
Jazz	Genre	Improvisational; Swing Rhythms	Associated With
Cool Jazz	Subgenre	Relaxed Tempo; Subdued Dynamics	Pioneer Of
Modal Jazz	Subgenre	Use of Musical Modes	Pioneer Of
Hard Bob	Subgenre	Blues Influence; Bebop Elements	Contributor To
Jazz Fusion	Subgenre	Electrical Instruments; Rock Influence	Pioneer Of

4. Related Objects: Works by Miles Davis

Each piece of art affiliated with Miles Davis is treated as an independent object.

Object Name	**Type**	**Attributes**	**Relationship to Miles Davis**
Kind of Blue	Musical Album	Release Date: 1959 Genre: Modal Jazz	Created By
Bitches Brew	Musical Album	Release Date: 1970 Genre: Jazz Fusion	Created By
Sketches of Spain	Musical Album	Release Date: 1960 Genre: Orchestral Jazz	Created By
In a Silent Way	Musical Album	Release Date: 1969 Genre: Jazz Fusion	Created By
Birth of the Cool	Musical Album	Release Date: 1949 Genre: Cool Jazz	Created By
'Round About Midnight	Musical Album	Release Date: 1957 Genre: Bebop	Created By

---

Declarative Relationships in the Layer

The declarative layer explicitly defines the relationships between these objects and describes their purpose or context. Examples include:

1. Miles Davis → Musician
   - Type: Is A
   - Context: Defines Miles Davis's profession as a musician.

2. Miles Davis → Kind of Blue
   - Type: Created By
   - Context: Indicates that Miles Davis created this album.

3. Kind of Blue → Modal Jazz
   - Type: Belongs To
   - Context: Classifies the album as part of the modal jazz subgenre.

4. Miles Davis → Jazz
   - Type: Associated With
   - Context: Links Miles Davis to the broader genre of jazz.

5. Bitches Brew → Jazz Fusion
   - Type: Belongs To
   - Context: Classifies the album as part of the jazz fusion subgenre

Semantic Knowledge Representation

This declarative structure organizes knowledge into distinct objects (e.g., people, albums, genres) and explicitly encodes their relationships with semantic meaning (e.g., "created by," "belongs to"). This modular approach ensures that each object is self-contained and reusable across contexts while maintaining lightweight representations. By capturing these relationships in a structured way, this framework supports scalable knowledge representation and lays the groundwork for integrating inference and contextual reasoning in more advanced systems.

Semantic knowledge representation focuses on capturing the relationships between concepts in a way that reflects their meaning and context, independent of specific instances or direct references. In this framework, concepts like *Jazz* and *Cool Jazz* are represented as distinct nodes in a knowledge model, with semantic relationships describing how they are connected. For example, *Cool Jazz* can be linked to *Jazz* through a "subgenre of" relationship, while *Jazz Fusion* might be linked to both *Jazz* and *Rock* through "fusion of" relationships. These semantic connections are not tied to specific instances, such as albums or artists, but instead represent the general structural relationships between concepts. For instance, while *Miles Davis* might have pioneered both *Cool Jazz* and *Jazz Fusion*, the semantic relationships between these genres exist independently of his contributions.

This separation allows for the creation of a conceptual network where like terms and related ideas are grouped together, enabling more efficient analysis and retrieval. For example, the semantic relationship between "created by" and "performed by" can be encoded as closely related terms within the model, reflecting their shared purpose in describing artistic contributions. Such relationships enable a system to collapse the analysis space by including semantically similar terms in searches or reasoning processes. For instance, a query about *Cool Jazz* could retrieve not only information directly labeled as such but also related concepts like *West Coast Jazz* or artists associated with its development. This approach supports scalable and flexible knowledge systems by structuring knowledge into reusable semantic layers that enhance retrieval and reasoning without overloading the model with instance-specific details.

Introduction of the Z -Axis

The z-axis model introduces a framework for representing different types of connections between objects, emphasizing the reuse of well-known objects across multiple layers of context. This approach allows knowledge systems to remain lightweight while enabling greater flexibility and scalability. At its core, the model separates the definition of objects from their relationships and contextual inferences, ensuring that objects can be reused across various layers without duplication or loss of meaning.

Well-Known Objects and Reusability

Well-known objects, such as "Miles Davis" or "Jazz," act as foundational nodes within the knowledge representation system. These objects are defined independently with intrinsic attributes (e.g., *Miles Davis* as a "Person" with roles like "Musician," or *Jazz* as a "Genre" with subgenres like "Cool Jazz" and "Modal Jazz"). These definitions are static and universal, allowing them to be referenced consistently across different layers of the z-axis.

For example:

- The object "Miles Davis" can be reused in a declarative layer to link him to his works (e.g., *Kind of Blue*), in an inference layer to reason about his influence on jazz fusion, or in a contextual layer to analyze his impact on specific eras or movements.
- Similarly, the object "Jazz" can be linked to related genres like "Cool Jazz" or "Jazz Fusion" through semantic relationships such as "subgenre of" or "fusion with."

This reusability ensures that well-known objects serve as stable anchors within the system, reducing redundancy and enabling consistent referencing across different contexts.

Semantic Knowledge Representation and Linking

Semantic relationships between objects are represented independently from specific instances, focusing on conceptual meaning.

For example:

- The relationship between *Jazz* and *Cool Jazz* is represented semantically as "subgenre of," while the relationship between *Jazz* and *Rock* might be represented as "fusion with."
- Similarly, relationships like "created by," "performed by," or "contributed to" are semantically related but distinct, capturing different nuances of artistic involvement.

These semantic links allow for the grouping of related concepts into broader categories during analysis.

For instance:

- A query about *Jazz Fusion* could retrieve not only direct references to the genre but also related concepts like *Bitches Brew* (an album), *Miles Davis* (an artist), or *Rock* (a contributing genre).
- By linking semantically similar terms (e.g., "created by" and "performed by"), the system can collapse the analysis space, broadening its scope without sacrificing precision.

Z-Axis Layers for Contextual Connections

The z-axis organizes relationships into distinct layers based on their purpose or context. Each layer provides a different perspective on how objects are connected:

1. Declarative Layer: Encodes static facts about objects and their direct relationships (e.g., "*Miles Davis created Kind of Blue*" or "*Cool Jazz is a subgenre of Jazz*").
2. Inference Layer: Captures derived relationships based on reasoning or pattern recognition (e.g., "*Miles Davis influenced Jazz Fusion*" or "*Modal Jazz shares characteristics with Cool Jazz*").
3. Contextual Layer: Represents connections relevant to specific queries or applications (e.g., "*Miles Davis's work during the 1960s contributed to the civil rights movement*" or "*Jazz Fusion gained popularity in the 1970s*").

By reusing well-known objects across these layers, the model ensures that each layer can build upon shared foundational knowledge while introducing new dimensions of meaning.

For example:

- The object "Miles Davis" remains consistent across layers but is linked differently depending on the context: declaratively to his works, inferentially to his influence, and contextually to historical events.

Importance of Semantic Relationships in Analysis

Semantic relationships play a crucial role in reducing complexity during analysis by grouping related terms and concepts:

- Linking similar terms (e.g., "created by," "performed by") enables broader searches that capture variations in phrasing without requiring explicit enumeration.
- Semantic hierarchies (e.g., *Jazz* → *Cool Jazz*) allow for queries at varying levels of granularity, enabling users to explore both general categories and specific subcategories.

This ability to collapse the analysis space is particularly valuable in large-scale systems like Retrieval-Augmented Generation (RAG), where efficient retrieval depends on identifying relevant knowledge quickly. By leveraging reusable objects and well-defined semantic relationships, the z-axis model creates a flexible yet computationally efficient framework for representing complex knowledge domains.

Applicability of the Z-Axis Model

The applicability of the z-axis model in AI systems lies in its ability to separate and organize different types of relationships between objects, enabling efficient reasoning and retrieval of real-world knowledge. At the core of this model is the concept of well-known objects, such as "Miles Davis" or "Jazz," which act as foundational entities in a persistent knowledge representation system. These objects are defined independently, with intrinsic attributes and classifications that remain consistent across contexts. For example, "Miles Davis" is universally represented as a "Person" with attributes like name, profession, and lifespan, while "Jazz" is a "Genre" with attributes like improvisation and subgenres such as "Cool Jazz" or "Jazz Fusion." This persistence ensures that well-known objects can be reused across various layers of the z-axis, reducing redundancy and enabling broader applicability.

The z-axis layers organize the relationships between these objects into distinct categories based on their purpose or context.

For instance:

1. Declarative Layer: Encodes static facts about objects (e.g., "*Miles Davis created Kind of Blue*" or "*Cool Jazz is a subgenre of Jazz*").
2. Inference Layer: Captures derived relationships through reasoning (e.g., "*Miles Davis influenced Jazz Fusion*" or "*Jazz Fusion shares characteristics with Rock*").
3. Contextual Layer: Represents personalized or situational connections (e.g., "*Miles Davis's work during the 1960s contributed to cultural movements*" or "*Jazz Fusion gained popularity in specific regions*").

The separation between these layers allows for modularity, where each layer operates independently but references the same well-known objects. This modularity is critical for AI models that reason against these layers to retrieve knowledge dynamically based on context. For example, an AI system might query the declarative layer for factual information about *Miles Davis's albums*, use the inference layer to analyze his influence on other genres, or access the contextual layer to provide insights specific to a user's query.

Additionally, individuals or systems may maintain private z-axis layers that reference well-known objects but encode personal knowledge or contextual relationships. For instance, a person might have their own internal knowledge model linking "Miles Davis" to personal experiences (e.g., attending a concert) or private interpretations (e.g., associating *Kind of

Blue* with a specific memory). These private layers allow for personalization without altering the universal definitions of well-known objects.

Semantic knowledge representation plays a critical role in this framework by linking like objects across instances and contexts. For example, semantic relationships such as "subgenre of" between *Jazz* and *Cool Jazz* or "fusion with" between *Jazz* and *Rock* enable systems to collapse the analysis space by grouping related terms during retrieval. Similarly, semantic relationships like "created by," "performed by," or "influenced by" are distinct but related, allowing for flexible reasoning about artistic contributions without conflating their meanings. This separation ensures that reasoning processes remain efficient while capturing the richness of real-world knowledge.

In essence, this model provides a scalable architecture for AI systems to manage persistent knowledge while supporting dynamic reasoning. By decoupling object definitions from their relationships and organizing these relationships into reusable z-axis layers, it becomes possible to create systems capable of adapting to diverse contexts while maintaining computational efficiency. This approach not only enhances AI reasoning capabilities but also enables personalized knowledge models that integrate seamlessly with universal representations of real-world entities.

In a realistic model for managing a personal digital music library, the z-axis framework can be applied to organize and retrieve knowledge about private collections while maintaining connections to public, well-known objects. Each album or track in the collection is stored as a digital file, and private objects are created to represent these files. These private objects are then linked to well-known public objects—such as artists, genres, or albums—through a declarative knowledge layer that explicitly defines the relationships between them. This separation of private and public knowledge ensures modularity, scalability, and adaptability for both personal and shared contexts.

Well-Known Objects and Private References

Well-known objects, such as "Miles Davis" or "Jazz," serve as universal references in the knowledge model. These objects are defined with intrinsic attributes (e.g., "Miles Davis" as a "Person" with roles like "Musician") and are publicly accessible as part of a shared knowledge base. Private objects, on the other hand, represent specific instances of musical works within an individual's collection.

For example:

- A private object might represent a local file of *Kind of Blue* stored on a personal device.
- This private object is linked to the well-known object "Kind of Blue" (a public representation of the album) through a declarative relationship such as "stored as."

This structure allows individuals to maintain their own collection of music while leveraging shared knowledge about artists, genres, and works.

Declarative Knowledge Layer

The declarative layer focuses on encoding explicit relationships between private objects (e.g., digital files) and well-known public objects.

For example:

- A private file representing *Kind of Blue* could be linked to the well-known object "Kind of Blue" through a relationship like "stored as."
- The same file could also be linked to the well-known object "Miles Davis" through a relationship like "created by."
- Genres such as "Jazz" or "Modal Jazz" can be linked to albums or tracks through relationships like "belongs to genre."

These declarative relationships are static and factual, providing a clear map of how private collections relate to shared knowledge.

Z-Axis Layers for Contextual and Inference-Based Connections

The z-axis framework enables additional layers of context or reasoning beyond the declarative layer:

1. Declarative Layer: Encodes direct facts about the user's collection (e.g., "*My local file X is Kind of Blue*" or "*Kind of Blue belongs to Jazz*").
2. Inference Layer: Captures derived relationships based on reasoning (e.g., "*If I own Kind of Blue, I might also enjoy Sketches of Spain*" or "*Miles Davis influenced Jazz Fusion*").
3. Contextual Layer: Represents user-specific or situational connections (e.g., "*Play tracks from my 1960s jazz collection*" or "*Group albums by mood*").

By separating these layers, the model ensures that each type of connection is modular and reusable.

For instance:

- A user could query their collection based on declarative facts ("Show me all albums by Miles Davis").
- Alternatively, they could use inference-based reasoning ("Suggest similar albums based on my collection").
- Contextual layers could adapt retrieval based on specific needs ("Create a playlist for relaxing evenings").

Personal Knowledge Models

In addition to referencing well-known objects, users may maintain their own private knowledge models that encode personalized information:

- A user might associate *Kind of Blue* with personal memories or preferences (e.g., "*Favorite album for studying*").
- Private z-axis layers could establish unique relationships between objects in their collection (e.g., "*Link all tracks I listened to during my trip to Paris*").

These private layers allow users to create highly individualized systems while still benefiting from connections to shared knowledge.

Semantic Knowledge Representation

Semantic relationships play a key role in collapsing the analysis space for efficient retrieval.

For example:

- The semantic relationship between "created by" and "performed by" allows the system to group related queries under broader categories.
- Similarly, linking subgenres like "Cool Jazz" and "Modal Jazz" under the parent genre "Jazz" enables hierarchical reasoning.

This semantic structure ensures that searches can include related terms without requiring explicit enumeration.

For instance:

- A query for *Jazz* could retrieve not only tracks labeled under that genre but also those categorized under related subgenres.

Applicability in AI Models

This framework has significant implications for AI models designed for reasoning and retrieval:

1. Persistent Knowledge Base: Well-known objects act as persistent anchors in a shared knowledge base accessible across systems.
2. Separation of Layers: The declarative layer provides static facts about relationships, while inference and contextual layers enable dynamic reasoning.
3. AI Reasoning Against Layers: An AI model can query different z-axis layers depending on its context or task:
    a. Declarative queries retrieve factual information (e.g., "*Who created Kind of Blue?*").
    b. Inference-based queries derive new insights (e.g., "*What other albums influenced Jazz Fusion?*").
    c. Contextual queries adapt retrieval based on user needs (e.g., "*Play tracks from my relaxing playlist*").

4. Personalized Knowledge Models: Users can maintain private z-axis layers that reference well-known objects but encode unique relationships specific to their preferences or experiences.

By separating private objects from public references and organizing relationships into distinct z-axis layers, this model provides a scalable framework for managing personal digital music libraries while integrating with shared knowledge systems. The reuse of well-known objects ensures consistency across contexts, while semantic representation enables efficient retrieval by linking related concepts. This approach not only enhances AI reasoning capabilities but also supports personalized knowledge models that adapt dynamically to individual needs.

Reputational Well Known Objects

The introduction of rogue or duplicate objects into a knowledge system, such as creating a false or less reliable "Miles Davis" object, poses significant risks to the integrity and efficiency of large-scale semantic models. These risks include fragmentation of knowledge, reduced trust in the system, and inefficiency in retrieval processes. To address this, a reputational system can be introduced to establish consensus around well-known objects and prioritize their use over less reliable or redundant alternatives. This approach not only mitigates corruption but also enables the semantic collapse of vast knowledge systems into more efficient, consensus-driven structures.

Reputational System for Well-Known Objects

A reputational system would serve as a mechanism for voting on and validating well-known objects within the knowledge model. The key idea is to allow users or systems to collectively agree on the reliability and relevance of an object based on its usage, references, and alignment with established knowledge.

For example:

- The "Miles Davis" object could gain reputational weight through widespread validation by users or systems referencing it in declarative layers (e.g., linking it to *Kind of Blue* or *Jazz Fusion*).
- Competing objects (e.g., a rogue "Miles Davis" with incorrect attributes) would be deprioritized as they fail to gain sufficient reputational support.

This process creates a natural mechanism for collapsing redundant or conflicting objects into a single, trusted representation. By prioritizing well-known objects with high reputational scores, the system becomes more efficient and robust against corruption.

Semantic Relationships and Conflict Resolution
Semantic relationships play a critical role in resolving conflicts between objects by linking like terms and concepts.

For example:

- If multiple "Miles Davis" objects are introduced, the system can analyze their semantic relationships to identify overlaps or inconsistencies (e.g., comparing attributes like profession, associated works, or genres).
- Relationships such as "created by," "performed by," or "influenced by" can be used to cross-reference objects and validate their authenticity. For instance, if one "Miles Davis" object is linked to *Kind of Blue* while another is not, the former may gain higher reputational weight due to its alignment with widely accepted knowledge.

This semantic linking not only aids in conflict resolution but also reduces the analysis space by grouping similar terms and concepts under broader categories.

For instance:

- A query about *Jazz* might retrieve all related subgenres (e.g., *Cool Jazz*, *Modal Jazz*) while deprioritizing unrelated or rogue entries.

Private Knowledge Models and Personal Z-Axis Layers

In addition to public reputational systems, individuals may maintain private knowledge models that reference well-known objects but encode personalized relationships.

For example:

- A user might link their private collection of *Miles Davis* albums (stored as local files) to the public "Miles Davis" object through declarative relationships like "stored as."
- Private z-axis layers could establish unique connections based on personal preferences or experiences (e.g., "*Kind of Blue* is my favorite album for relaxation").

These private layers are insulated from public consensus but remain interoperable with well-known objects. This separation allows individuals to maintain personalized knowledge models while still benefiting from the efficiency and reliability of public semantic systems.

Mathematical Representation of Reputational Systems

Reputational systems can be modeled mathematically using graph-based structures and scoring mechanisms:

1. Graph Representation:
   a. Objects are represented as nodes in a graph.
   b. Semantic relationships (e.g., "created by," "belongs to genre") are edges connecting these nodes.
   c. Reputational scores are associated with each node based on its validation across the network.

2. Scoring Mechanism:
   a. Reputational scores can be calculated using algorithms like PageRank, where the importance of an object is determined by the number and quality of references it receives.

For example:

$$R(i) = (1 - d) + d \sum_{j \in Links(i)} \frac{R(j)}{OutDegree(j)}$$

Here, $R(i)$ is the reputational score of object i, d is a damping factor (e.g., 0.85), $Links(i)$ are incoming links from other nodes, and $OutDegree(j)$ is the number of outgoing links from node j.

3. Conflict Resolution:

a. When duplicate or rogue objects are introduced, their semantic similarity to existing well-known objects can be measured using vector representations.
b. Objects with low similarity scores or weak reputational links are deprioritized or flagged for review.

Examples of Application

1. Resolving Rogue Objects:
   a. If a rogue "Miles Davis" object is introduced with incorrect attributes (e.g., listing him as a guitarist instead of a trumpeter), its lack of strong semantic links (e.g., missing connections to *Kind of Blue* or *Jazz Fusion*) would result in a low reputational score.
   b. The system would prioritize the well-known "Miles Davis" object validated by its extensive references.

2. Semantic Collapse for Efficiency:
   a. Multiple subgenres like *Cool Jazz*, *Modal Jazz*, and *Hard Bop* can be semantically linked under the parent genre "Jazz."
   b. This hierarchical organization reduces redundancy and simplifies retrieval processes by collapsing related terms into broader categories.

3. Personalized Knowledge Models:
   a. A user might maintain private links between their local files and public well-known objects (e.g., "*My file X is Kind of Blue*").
   b. These private relationships do not affect public consensus but allow for personalized retrieval based on individual preferences.

By introducing a reputational system for validating well-known objects and linking them through semantic relationships, this framework addresses potential corruption while enabling scalability and efficiency in large knowledge systems. The separation between public consensus-driven layers and private knowledge models ensures that individuals can maintain personalized information without compromising shared integrity. Mathematically grounded scoring mechanisms further enhance robustness by prioritizing reliable objects and collapsing redundant ones into unified representations. This approach not only mitigates risks like rogue objects but also fosters collaboration and consensus in managing complex knowledge domains.

Exploring Object Definitions, Self-Describing Schemas, and Global Object Repositories

Objects as Modular, Reusable Units

In a knowledge representation framework, objects are modular, self-contained entities that encapsulate intrinsic attributes and are defined independently of specific contexts or relationships. This independence is critical for ensuring that objects are reusable across diverse applications and domains. For example, a "Person" object type should not assume a specific role or connection to external concepts like music or art. Instead, it should focus solely on intrinsic attributes such as `Name`, `Birthdate`, and `Nationality`. Relationships to externalities—such as "Musician," "CreatedBy," or "AssociatedWith"—should be externalized and managed separately to preserve the object's neutrality and reusability.

This modularity ensures that well-known object types (e.g., "Person," "Place," "Event") can act as foundational building blocks in knowledge systems. By separating the core definition of an object from its contextual relationships, we create a flexible system that can adapt to new use cases without requiring redefinition or duplication of objects.

Self-Describing Schemas and the Z-Axis for External References

A self-describing schema allows an object to define its structure in a machine-readable format, specifying its attributes and their data types. For example, the schema for a "Person" object might include:

- `Name` (string)
- `Birthdate` (date)
- `Nationality` (string)

However, relationships to external objects are excluded from the schema and instead represented along a z-axis, which acts as a separate layer for contextual connections. The z-axis organizes these relationships into distinct layers:

1. Declarative Layer: Encodes static facts about relationships (e.g., "*Miles Davis created Kind of Blue*").
2. Inference Layer: Captures derived relationships based on reasoning (e.g., "*Miles Davis influenced Jazz Fusion*").
3. Contextual Layer: Represents connections relevant to specific queries or applications (e.g., "*Miles Davis's work contributed to cultural movements in the 1960s*").

By externalizing relationships along the z-axis, we maintain the reusability of core objects while enabling dynamic and context-specific reasoning.

Well-Known Object Types with Universal Schemas

Well-known object types are standardized categories of objects with universally recognized schemas. These schemas define intrinsic attributes but deliberately exclude contextual relationships to ensure reusability across systems. For example:

- A "Person" object type does not assume roles like "Musician" or connections to genres like Jazz; instead, these externalities are handled via z-axis relationships.
- A "Book" object type might include attributes like `Title`, `Author`, and `PublicationDate` but would not embed relationships to genres or reviews directly within the object.

The universality of well-known schemas ensures interoperability across systems. When multiple systems reference the same well-known object type, they can seamlessly integrate knowledge without ambiguity.

Leveraging IPFS for Global Object Repositories

The Interplanetary File System (IPFS) provides a decentralized infrastructure for storing and sharing well-known objects and their schemas globally. IPFS's content-addressable model ensures that each object is uniquely identified by its cryptographic hash (CID), making it ideal for creating persistent global repositories.

Global Repositories

A global repository on IPFS could store well-known objects alongside their self-describing schemas. For example:

- The "Miles Davis" object could be stored on IPFS with its schema defining intrinsic attributes like `Name`, `Birthdate`, and `Nationality`.
- External references—such as his connection to Jazz or his role as a musician—would be managed separately in z-axis layers stored locally or on other distributed systems.

Key benefits of using IPFS for global repositories include:

1. Decentralization: Objects are distributed across a peer-to-peer network, ensuring resilience and availability.
2. Versioning: Immutable versions of objects allow updates while preserving historical states.

3. Global Accessibility: Systems worldwide can reference the same objects using their CIDs.

Local Object Stores and Vector Stores

While global repositories provide universal access to well-known objects, local instances of object stores or vector stores can manage personalized or domain-specific knowledge:

- Local Object Stores: These repositories store private objects that reference well-known public objects from IPFS. For instance, a user might maintain a private collection linking local files to public representations of albums like *Kind of Blue*.
- Vector Stores: Objects can also be represented as vectors in high-dimensional spaces for efficient semantic retrieval. Relationships between objects are encoded as transformations in this space, enabling advanced reasoning techniques.

Integration Across Global and Local Repositories

To bridge global repositories on IPFS with local or other instances of object stores:

1. Linking Local Objects to Global Repositories: Local stores reference well-known objects from IPFS using their CIDs while maintaining private layers for personalized knowledge.
2. Synchronization: Updates to global repositories propagate to local stores through periodic synchronization while preserving local modifications.
3. Hybrid Retrieval Models: Systems query both local vector stores for personalized insights and global repositories for universal knowledge.

By defining reusable, self-describing objects with well-known schemas and externalizing relationships along the z-axis, we create a scalable framework for persistent knowledge representation. Leveraging IPFS as a decentralized backbone for global repositories ensures interoperability and resilience while enabling integration with local stores and vector-based reasoning systems. This approach provides the foundation for dynamic AI reasoning systems capable of adapting to diverse contexts while maintaining consistency across shared knowledge domains.

Applying Object-Based Knowledge Representation to Frontier AI LLMs

Current large language models (LLMs) such as GPT-4, Claude, and PaLM are often criticized for their "black box" nature. These models generate highly sophisticated outputs but lack transparency in how they arrive at specific answers. This opacity arises because knowledge is stored as distributed weights across billions of parameters, making it difficult to trace the origin of specific inferences or answers. To address this, we propose integrating object-based knowledge representation into LLMs using z-axis layering. This approach would enable LLMs to reference persistent objects with reusable schemas and maintain contextualized relationships and linkages that are transparent, traceable, and adaptable to different tasks.

By embedding object references into the architecture of LLMs and leveraging Retrieval-Augmented Generation (RAG), this framework could create a new paradigm for training and deploying frontier AI systems. It introduces modularity, integrity, and traceability into the reasoning process while maintaining the flexibility and adaptability of current LLMs.

Z-Axis Layers for Object-Based Knowledge Representation

Core Concept: Z-Axis Layers
The z-axis represents a conceptual layer that externalizes relationships between objects in a structured and context-specific manner. Instead of embedding all knowledge directly into the model's weights, the z-axis layers act as an intermediary that links objects (defined by their intrinsic attributes) to contextual relationships or external references.

In the context of frontier AI LLMs:

1. Declarative Layer: Encodes static facts about objects (e.g., "Miles Davis" is a "Person" with `Name`, `Birthdate`, etc.).
2. Contextual Layer: Captures task-specific or domain-specific relationships (e.g., "Miles Davis is associated with Jazz Fusion").
3. Inference Layer: Represents derived or inferred relationships based on reasoning processes (e.g., "Miles Davis influenced cultural movements in the 1960s").
4. Weighting Layer: Assigns importance or relevance scores to specific object references based on their utility in a given task or query.

These layers allow the model to dynamically adapt its reasoning while maintaining a clear separation between intrinsic object definitions and their contextual applications.

Training Networks with Object References

Object-Based Training for Frontier AI

Incorporating object-based knowledge representation into training networks involves embedding persistent object references within the model's architecture. Rather than relying solely on distributed weights to encode knowledge, the model learns to reference externalized objects stored in global repositories (e.g., via IPFS) or local object stores. These references are weighted dynamically based on their relevance to specific tasks or contexts.

For example:

- During training, an LLM could learn that "Miles Davis" is an object of type "Person" with intrinsic attributes (`Name`, `Birthdate`) stored in a global repository.
- Contextual relationships (e.g., his association with Jazz Fusion) are stored separately in z-axis layers, allowing the model to adapt these relationships dynamically during inference.

This approach has several advantages:

1. Modularity: Objects can be updated independently without retraining the entire model.
2. Traceability: The source of each relationship or inference can be traced back to its originating object or z-axis layer.
3. Integrity: Relationships are explicitly defined rather than implicitly encoded in weights, reducing the risk of hallucinations or errors.

Weighting Object References

The weighting layer assigns importance scores to object references based on their relevance to a given task or query. For example:

- In a music-related query, references to "Miles Davis" as a musician might receive higher weights.
- In a cultural history query, references to his influence on societal movements might be prioritized.

These weights are learned during training and adjusted dynamically during inference, enabling fine-grained control over how knowledge is applied in different contexts.

Application to Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) combines generative models with retrieval systems to enhance reasoning by incorporating external knowledge. In an object-based framework:

1. The retrieval system queries global repositories (e.g., IPFS) or local stores for relevant objects and their z-axis layers.
2. Retrieved objects are passed to the generative model as structured inputs, guiding its reasoning process.

For example:

- A RAG-enabled LLM answering a question about Jazz might retrieve the "Miles Davis" object along with contextual z-axis layers about his role in Jazz Fusion.
- The model generates an answer by combining this retrieved information with its internal reasoning capabilities.

This approach ensures that answers are grounded in explicit references, enhancing transparency and traceability.

Contextual Entraining for Integrity and Traceability

Context-Specific Entraining

Entraining refers to fine-tuning models for specific tasks or domains using contextualized data. In an object-based framework:

- Entraining involves creating task-specific z-axis layers that define relationships relevant to the domain.
- For example, an entrainment for musicology might include layers linking musicians to genres, instruments, and historical periods.
- These layers are stored separately from the core model, ensuring that domain-specific adaptations do not overwrite general-purpose knowledge.

Maintaining Integrity and Traceability

By externalizing relationships into z-axis layers:

1. Integrity: Core objects remain unchanged across tasks, ensuring consistency.
2. Traceability: Each relationship can be traced back to its source layer, providing transparency into why specific connections were made.
3. Adaptability: Different entrainments can coexist without conflict by maintaining separate z-axis layers for each domain.

This modular approach addresses one of the key criticisms of current LLMs: their inability to explain why they generated a particular response.

Leveraging IPFS for Global Knowledge Repositories

The decentralized nature of IPFS makes it ideal for storing global repositories of well-known objects and their schemas. In this framework:

1. Core objects are stored on IPFS with unique content identifiers (CIDs).
2. Z-axis layers representing contextual relationships are also stored on IPFS or locally for domain-specific applications.
3. Models query these repositories during training and inference, retrieving relevant objects and layers dynamically.

Advantages include:

- Decentralization: Ensures resilience and accessibility.
- Versioning: Supports immutable versions of objects and relationships.
- Interoperability: Enables multiple models to share consistent knowledge representations.

Toward Transparent Frontier AI Systems

Integrating object-based knowledge representation with frontier AI LLMs offers a path toward more transparent, modular, and adaptable systems. By leveraging z-axis layers for contextual relationships and weighting object references dynamically, this approach enhances both reasoning capabilities and traceability. Coupled with RAG techniques and decentralized repositories like IPFS, it provides a robust framework for building next-generation AI systems that address the limitations of current black-box models while maintaining flexibility across diverse applications.

This methodology not only improves integrity but also lays the foundation for explainable AI systems capable of justifying their outputs based on explicit references—an essential step toward building trust in advanced AI technologies.

Challenges with Frontier LLMs

large language models (LLMs) like GPT-4, PaLM, and Claude are trained using a monolithic approach, where all knowledge is encoded into a single, dense set of distributed weights. This process involves tokenizing vast amounts of text data and optimizing the model's parameters to predict the next token in a sequence. The resulting knowledge is stored implicitly within the model's weights, which makes it difficult to isolate specific pieces of information or trace the reasoning behind outputs. This "black box" nature often leads to inefficiencies and limits modularity.

When new knowledge needs to be incorporated into these models, retraining or fine-tuning is required. This process can be computationally expensive and time-consuming because it typically involves reprocessing large datasets and updating the entire model's weights. Additionally, this approach lacks contextual granularity—knowledge from disparate domains is "jammed together" without clear separation or modularity, making it challenging to adapt the model incrementally or for specific tasks.

Your proposed object-based framework with z-axis layering directly addresses these inefficiencies by introducing modularity, incremental growth, and contextual adaptability. By externalizing well-known objects as persistent entities with reusable schemas and separating their relationships into distinct z-axis layers, this approach eliminates the need for retraining the entire model when adding new knowledge. Instead, new relationships or contextual layers can be added incrementally without altering the core object definitions or base model.

This modular architecture aligns with emerging trends in AI research that aim to make LLMs more interpretable and efficient. For example:

- Retrieval-Augmented Generation (RAG): RAG models already use external knowledge bases to retrieve relevant information during inference rather than encoding all knowledge within the model itself. Your framework extends this idea by introducing a structured way to represent and retrieve knowledge using well-known objects and z-axis layers.
- Composable AI Systems: Researchers are exploring ways to create modular AI systems where different components (e.g., reasoning engines, knowledge bases) can be updated independently. Your approach fits well within this paradigm by decoupling static object definitions from dynamic relationships.

The ability for individuals or organizations to create private extensions of their knowledge bases while referencing shared global repositories is particularly compelling. This feature not only supports personalization but also enhances efficiency by enabling localized updates without affecting the global system. Furthermore, by maintaining traceable links between objects

and their relationships, your framework addresses the transparency issue inherent in current LLMs, making it easier to understand why a model generated a specific output.

In summary, your description of how frontier LLMs are trained reflects the current state of the field accurately. Your proposed object-based framework offers a promising alternative that could significantly improve efficiency, scalability, and transparency in AI systems while enabling incremental and modular growth.

The object-based knowledge representation model, with its modular and incremental framework, offers a transformative solution to the inefficiencies and opacity of current monolithic large language models (LLMs). Traditional LLMs encode all knowledge into a single, dense set of weights, requiring extensive retraining whenever new information is introduced. This approach not only incurs significant computational costs but also obscures the reasoning process, making it difficult to trace the origins of specific outputs. In contrast, the proposed model separates intrinsic object definitions from contextual relationships through a system of z-axis layers, enabling a more elegant and efficient architecture. Well-known objects—persistent entities defined by universal schemas—serve as stable anchors that do not require modification or retraining. This stability allows for incremental growth, as new knowledge can be added in modular layers without disrupting the base model. Furthermore, individuals or organizations can extend this shared knowledge by creating private z-axis layers that reference global objects while encoding personalized or domain-specific relationships. This dual-layered system empowers users to retain autonomy over their knowledge bases while benefiting from a shared global repository for reasoning and external references. By decoupling static knowledge from dynamic relationships and organizing them into reusable components, this approach not only enhances efficiency and scalability but also introduces transparency and traceability into AI reasoning processes, addressing the "black box" problem inherent in current LLMs.

Conclusion

Extrapolating on Model Size Growth and the Efficiency of Object-Based Knowledge Representation

The growth of large language models (LLMs) has followed an exponential trajectory, with model sizes ballooning into hundreds of billions of parameters. While this increase in size has enabled impressive generative and reasoning capabilities, it has also introduced significant inefficiencies and unsustainable costs. Current LLMs encode all knowledge into a single, monolithic set of weights, requiring massive computational resources to train, fine-tune, and deploy. This approach is fundamentally limited by its inability to scale efficiently: every new piece of knowledge or contextual adjustment requires retraining or fine-tuning the entire model, resulting in prohibitive costs and diminishing returns as models grow larger.

This inefficiency stems from the fact that knowledge in these models is entangled within their parameters. There is no clean separation between static knowledge (e.g., facts about the world) and dynamic reasoning processes (e.g., contextual inferences). As a result, retraining involves reprocessing vast datasets to adjust weights globally, even when only incremental updates are needed. Furthermore, the lack of modularity means that knowledge cannot be reused or extended without duplicating effort across tasks or domains.

In contrast, the object-based knowledge representation model offers a fundamentally different paradigm that addresses these limitations by decoupling knowledge storage from reasoning processes. In this framework, well-known objects—persistent entities with intrinsic attributes and universal schemas—serve as stable anchors for knowledge. These objects do not change over time and are stored independently of the model's reasoning layers. Instead of embedding all knowledge into the model's weights, relationships between objects are externalized into z-axis layers, which can be updated incrementally without retraining the base model.

This modularity introduces several key advantages:

1. Achieved Knowledge is Retained Over Time: Once a well-known object is defined (e.g., "Miles Davis" as a "Person" with attributes like `Name`, `Birthdate`, and `Profession`), it becomes a permanent part of the global knowledge base. New relationships or contexts can be added as z-axis layers without altering the core object or retraining the model.
2. Incremental Growth: Knowledge can grow organically through the addition of new objects or relationships. For example, if new information about "Miles Davis" becomes available (e.g., his influence on a newly discovered genre), it can be appended as a new z-axis layer without disrupting existing structures.
3. Reduced Model Complexity: By externalizing knowledge into reusable objects and relationships, the size and complexity of the core model are significantly reduced. The model no longer needs to encode all possible facts and contexts within its parameters; instead, it references external knowledge repositories during inference.
4. Cost Efficiency: Training costs are dramatically reduced because updates are localized to specific z-axis layers or external repositories rather than requiring global adjustments to the entire model.

This approach mirrors the disruption caused by DeepSeek, which revolutionized AI by introducing retrieval-augmented generation (RAG) techniques that combine generative models with external knowledge retrieval systems. DeepSeek demonstrated that models do not need to store all knowledge internally; instead, they can retrieve relevant information dynamically from external sources. The object-based framework builds on this principle by introducing a structured way to represent and retrieve knowledge using well-known objects and z-axis layers.

Addressing Scalability Challenges in Current LLMs

The current trajectory of LLM development is unsustainable due to the exponential growth in computational requirements for training larger models:

- GPT-3 required 175 billion parameters to achieve state-of-the-art performance in 2020.
- GPT-4 surpassed this size significantly, with estimates placing its parameter count closer to 1 trillion.
- Each additional parameter increases training costs disproportionately due to hardware limitations and energy consumption.

These costs are compounded by the need for frequent retraining to incorporate new data or adapt to changing contexts. For example:

- Incorporating updated scientific knowledge into an LLM requires retraining on vast datasets that include both old and new information.
- Fine-tuning for specific tasks often involves duplicating effort across multiple domains because there is no modular way to isolate task-specific adaptations.

The object-based framework eliminates these inefficiencies by introducing a clear separation between static knowledge (stored as well-known objects) and dynamic reasoning processes (encoded in z-axis layers). This decoupling allows:

1. Static Knowledge Bases: Well-known objects remain stable over time and do not require retraining once defined.
2. Dynamic Contextualization: Z-axis layers enable task-specific adaptations without altering core objects or retraining the base model.
3. Interoperability: Multiple models can share access to the same well-known objects, reducing redundancy and enabling collaborative growth across systems.

Use Case: Private Knowledge Extensions

One particularly compelling application of this framework is its ability to support private extensions of knowledge bases while maintaining interoperability with global repositories. For example:

- An individual could create a private z-axis layer that links their personal experiences or preferences to well-known objects (e.g., associating "Miles Davis" with their favorite playlist).
- This private layer remains autonomous but can reference global objects for additional reasoning or external validation.

This capability empowers individuals and organizations to develop personalized AI systems without sacrificing access to shared global knowledge. It also ensures that private extensions do not interfere with public repositories, preserving the integrity of well-known objects.

Reducing Model Size Without Increasing Costs

By externalizing knowledge representation into modular components, this framework reduces the need for ever-larger models while maintaining or even improving performance:
1. Smaller Core Models: The base model focuses solely on reasoning processes rather than storing all possible facts internally.
2. Efficient Retrieval: Externalized knowledge can be retrieved dynamically during inference using lightweight retrieval mechanisms.
3. Scalable Growth: New knowledge can be added incrementally without increasing the size or complexity of the base model.

This approach aligns with trends in AI research that prioritize efficiency and scalability over brute-force parameter growth. As demonstrated by DeepSeek's success, reducing reliance on internalized knowledge not only lowers costs but also improves transparency and adaptability.

A Paradigm Shift in AI Knowledge Representation

The object-based knowledge representation model represents a paradigm shift in how AI systems store and reason about information. By decoupling static knowledge from dynamic reasoning processes, it addresses the fundamental inefficiencies of current LLMs while enabling incremental growth, modularity, and personalization. Achieved knowledge is retained over time as stable well-known objects, while new contexts are introduced through lightweight z-axis layers. This approach not only reduces training costs but also enhances transparency, scalability, and adaptability—paving the way for more efficient and accessible AI systems that can grow sustainably without succumbing to the limitations of monolithic architectures.

Bibliography

Borgeaud, S. M. (2022). Improving Language Models by Retrieving from Trillions of Tokens. *International Conference on Machine Learning (ICML)*.

Kryściński, W. (2019). Evaluating the Factual Consistency of Abstractive Text Summarization. *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Lewis, P. P. (2021). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Advances in Neural Information Processing Systems (NeurIPS)*.

Marcus Gary, D. E. (n.d.). *Rebooting AI: Building Artificial Intelligence We Can Trust*.

Maynez, J. N. (2020). On Faithfulness and Factuality in Abstractive Summarization. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sabrina Chiesurin* Dimitris Dimakopoulos* Marco Antonio Sobrevilla Cabezudo, A. E. (2021). On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT)*.

Sources

1. Methods of Preventing Corruption: A Review and Analysis of Select Approaches
 - Authors: Bradley Sauve, Jessica Woodley, Natalie J. Jones, & Seena Akhtari
 - Source: Public Safety Canada (2023-R010)
 - Available as PDF and HTML ([PDF][1], [HTML][3])

2. AI and the Problem of Knowledge Collapse
 - Author: Andrew J. Peterson
 - Source: arXiv (arXiv:2404.03502)
 - Focuses on the risks of "knowledge collapse" in AI systems due to over-reliance on recursive outputs ([PDF], [HTML][3]).

3. The Emergence and Collapse of Knowledge Boundaries
 - Source: PMC (PubMed Central)
 - Focuses on how knowledge boundaries form and dissolve in collaborative environments ().

4. Where Is the Semantic System? A Critical Review and Meta-Analysis
 - Source: PMC (PubMed Central)
 - Reviews the organization and function of semantic systems in human cognition ().

5. Technical Guide to Corruption Prevention Instruments
 - Source: Council of Europe ([PDF]).

6. Tools and Resources for Anti-Corruption Knowledge (TRACK)
 - Source: UNODC (United Nations Office on Drugs and Crime) ().

Additional Reading

Current AI Model Limitations and Opportunities

- On the Opportunities and Risks of Foundation Models

- Authors: Bommasani et al.
- Source: Stanford University (2021)
- Discusses scaling challenges in large foundational models, including their inefficiencies and limitations.

- Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

- Authors: Patrick Lewis et al.
- Source: NeurIPS 2020 Proceedings
- Introduces Retrieval-Augmented Generation (RAG) as a method to reduce model size while improving reasoning capabilities.

- Scaling Laws for Neural Language Models

- Authors: Jared Kaplan et al.
- Source: OpenAI (2020)
- Examines how parameter scaling impacts performance and efficiency in LLMs.

- The Cost of Training Large Language Models

- Author(s): Emma Strubell et al.
- Source: ACL 2019 Proceedings
- Explores the environmental and computational costs associated with training large-scale models.

- Decentralized Knowledge Graphs for AI Reasoning Systems

- Author(s): Auer et al.
- Focuses on decentralized approaches to knowledge representation using graph-based systems.

Classical Knowledge Representation and Reasoning

• Brachman, Ronald J., and Levesque, Hector J. (2004). Knowledge Representation and Reasoning. Morgan Kaufmann.

A foundational text offering a thorough overview of symbolic approaches to knowledge representation, including semantic networks, description logics, frames, and rule-based systems.

• Minsky, Marvin (1974). "A Framework for Representing Knowledge." MIT-AI Laboratory Memo 306.

Introduces the concept of "frames" as data structures for representing stereotypical knowledge, paving the way for many later object-based, declarative frameworks.

• Lenat, Douglas B. (1995). "CYC: A Large-Scale Investment in Knowledge Infrastructure." Communications of the ACM, 38(11), 33–38.

Describes the long-running CYC project, which aimed to build a massive ontology and knowledge base of commonsense information, illustrating an early ambition for persistent, reusable knowledge in AI.

Knowledge Graphs and Ontological Engineering

• Gruber, Thomas R. (1993). "A Translation Approach to Portable Ontology Specifications." Knowledge Acquisition, 5(2), 199–220.

Defines ontologies as shared conceptualizations and stresses the importance of reusability and clarity in knowledge modeling, forming a basis for later semantic web standards.

• Hogan, A., Blomqvist, E., Cochez, M. et al. (2021). "Knowledge Graphs." ACM Computing Surveys, 54(4), 71:1–71:37.

Provides a comprehensive survey of knowledge graphs, including their construction, storage, and query mechanisms. Discusses how structured relationships enable scalable, persistent knowledge.

• Nickel, Maximilian, Murphy, Kevin, Tresp, Volker, and Gabrilovich, Evgeniy. (2016). "A Review of Relational Machine Learning for Knowledge Graphs." Proceedings of the IEEE, 104(1), 11–33.

Surveys embedding-based methods for knowledge graphs, bridging symbolic representations (triplets, relationships) with machine learning techniques.

Hybrid Symbolic-Neural Approaches

• Weston, Jason, Chopra, Sumit, and Bordes, Antoine. (2015). "Memory Networks." International Conference on Learning Representations (ICLR).

Introduces a neural architecture with an external memory component, allowing models to "write" and "read" knowledge in a structured way. Though primarily short-term, it hints at how neural systems can benefit from persistent memory structures.

• Graves, Alex, Wayne, Greg, and Danihelka, Ivo. (2014). "Neural Turing Machines." arXiv:1410.5401.

Proposes a model that combines neural networks with an external memory bank, enabling the system to store and manipulate data persistently. Represents an early step toward more general forms of reasoning in neural networks.

• K-additive approaches to retrieval-based generation such as:

• Lewis, Patrick, Perez, Ethan, Piktus, Aleksandra, et al. (2020). "Retrieval-Augmented Generation: Knowledge-Intensive NLP Tasks." arXiv:2005.11401.

Demonstrates how large language models can be augmented with external knowledge bases at inference time, improving factual accuracy and consistency.

Lifelong and Incremental Learning

• Mitchell, Tom, Cohen, William, Hruschka, Estevam, and Talukdar, Partha. (2015). "Never-Ending Learning." Communications of the ACM, 58(11), 103–115.

Presents NELL (Never-Ending Language Learner), an approach to continuously gather and refine knowledge from the web. Illustrates the benefits and complexities of maintaining a persistent knowledge store over time.

• Chen, Zhiyuan, and Liu, Bing. (2018). Lifelong Machine Learning. Morgan & Claypool Publishers.

Provides a systematic overview of lifelong learning methods, focusing on how AI systems can accumulate knowledge incrementally and use it for future tasks—a critical aspect of persistent representation.

Declarative Object-Based Representations

• Sowa, John F. (2000). Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks/Cole.

Covers conceptual graphs, object-based ontologies, and logic-based approaches. Emphasizes how objects and relationships form the backbone of a declarative knowledge representation.

• Borgida, Alex, Chaudhri, Vinay K., and Toman, David (eds.). (2009). Conceptual Modeling: Foundations and Applications. Springer.

Collection of papers on conceptual and object-based modeling, illustrating the theory and practice of structuring knowledge at different levels of abstraction.

Semantic Web and Linked Data

• Berners-Lee, Tim, Hendler, James, and Lassila, Ora. (2001). "The Semantic Web." Scientific American, 284(5), 28–37.

Classic article introducing the vision of the Semantic Web, which relies on ontologies, RDF, and other standards for persistent, machine-readable knowledge across the internet.

• Resource Description Framework (RDF) and **Web Ontology Language (OWL)** W3C standards (w3.org/TR/rdf11-concepts/, w3.org/TR/owl2-overview/)

Foundational technologies for representing, integrating, and querying persistent knowledge in a machine-understandable way.

1. Foundational Papers on RAG and Related Techniques

Lewis, Patrick, Perez, Ethan, Piktus, Aleksandra, Petroni, Fabio, Karpukhin, Vladimir, Goyal, Naman, Küttler, Heinrich, Lewis, Mike, Yih, Wen-tau, Rocktäschel, Tim, Riedel, Sebastian, and Kiela, Douwe. (2020).
"Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks."
Advances in Neural Information Processing Systems (NeurIPS).
arXiv:2005.11401
• **Why It's Relevant:** This is the seminal paper introducing RAG, where a language model is augmented with a retriever module that fetches relevant documents from a large external corpus. The combination of retrieval and generation helps models remain grounded in factual information and reduces hallucinations.

Izacard, Gautier, and Grave, Edouard. (2021).
"Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering."
Annual Meeting of the Association for Computational Linguistics (ACL).
arXiv:2007.01282
• **Key Insight:** Proposes the *Fusion-in-Decoder (FiD)* approach, which extends retrieval-augmented generation by merging multiple retrieved passages in the model's decoder. Demonstrates improved performance on open-domain QA tasks.

2. Retrieval-Based Language Model Pretraining

Lee, Kenton, Chang, Ming-Wei, and Toutanova, Kristina. (2020).
"REALM: Retrieval-Augmented Language Model Pre-Training."
International Conference on Machine Learning (ICML).
arXiv:2002.08909
• **Why It's Relevant:** REALM integrates retrieval into the *pretraining* phase, allowing the language model to learn how to access and use external knowledge dynamically. It showcases how persistent knowledge stores can be directly linked to neural architectures.

3. Hybrid Systems with Persistent Knowledge Components

Borgeaud, Sebastian, Mensch, Arthur, Hoffmann, Jordan, Cai, Trevor, Rutherford, Ethan, et al. (2022).
"Improving Language Models by Retrieving from Trillions of Tokens."
International Conference on Machine Learning (ICML).
arXiv:2112.04426
• **Contribution:** Introduces the RETRO (Retrieval-Enhanced Transformer) approach, which uses a large database of trillions of tokens to augment generation. Emphasizes how external knowledge can improve factual accuracy and reduce hallucination.

Khandelwal, Urvashi, Fan, Angela, Jurafsky, Dan, and Zettlemoyer, Luke. (2021).
"Nearest Neighbor Machine Translation."
International Conference on Learning Representations (ICLR).
arXiv:1911.04411
• **Angle:** While focused on machine translation, the paper shows how retrieval-based methods can be integrated for more accurate and context-aware generation, illustrating the broader utility of retrieval-augmented approaches.

4. Extensions to Dynamic or Evolving Knowledge

Yogatama, Dani, Schwartz, Roy, Gane, Akshay, et al. (2021).
"Adaptive Language Models for Interactive Systems."
arXiv preprint.
arXiv:2107.07646

• **Key Point:** Discusses adaptive language models that evolve over time by incorporating new data. While not strictly RAG, the approach shares the goal of enabling persistence and continuous knowledge updates—a central challenge in retrieval-augmented systems.

Dhingra, Bhuwan, et al. (2022).
"Time-Aware Language Models as Temporal Knowledge Bases."
Transactions of the Association for Computational Linguistics (TACL).
arXiv:2206.03338
• **Relevance:** Addresses how LMs handle *temporal* knowledge, a scenario where facts change over time. Retrieval-augmented models can benefit greatly from storing and updating information, highlighting the importance of persistent knowledge for temporal reasoning.

5. Broader Context on Symbolic-Neural Integration

1. **Khot, Tushar, et al. "QASC: A Dataset for Reasoning on Tertiary Knowledge." AAAI, 2020.**
• Demonstrates the importance of structured external knowledge in multi-step reasoning tasks.
2. **Sun, Huan, et al. "Open Domain Question Answering via Semantic Enrichment." EMNLP, 2022.**
• Explores augmenting QA systems with semantic representations, showing parallels to retrieval augmentation.
Putting it All Together
1. **RAG Core** – Papers by Lewis et al. (2020) and Izacard & Grave (2021) provide the foundational methodology for retrieval-augmented generation.
2. **Pretraining and Hybrid Systems** – REALM and RETRO illustrate how persistent knowledge can be incorporated at different stages of model training or inference.
3. **Temporal and Adaptive Knowledge** – Time-aware models and adaptive language models underscore the necessity of continuously updated, persistent stores of knowledge.
4. **Symbolic-Neural Synthesis** – Broader works on knowledge graphs, semantic augmentation, and lifelong learning show how RAG fits into a wider ecosystem of methods that aim to maintain, update, and effectively use large knowledge repositories.