# Aligning AI Model's Knowledge and Conceptual Model's Symbols

***Abstract*—Artificial Intelligence (AI) models and Conceptual Models (CMs) offer complementary strengths: while AI learns statistical patterns from multimodal data, CMs provide precise, human-readable symbolic representations. However, a semantic gap exists between the high-dimensional vector embeddings produced by AI models and the symbolic elements used in modeling languages such as UML, BPMN, and OntoUML. This paper introduces ALeKS (Aligned Embeddings of Knowledge and Symbols), a framework for aligning AI knowledge representations with conceptual modeling symbols in a joint embedding space. By learning a transformation function—termed a *semantic shift*—we align natural language and conceptual representations. Our evaluation includes alignment accuracy, generalizability across modeling languages, and a user study based on the Technology Acceptance Model. Results show that enriched conceptual terms significantly improve alignment, and that practitioners find the approach both useful and easy to adopt. We provide a tool for alignment inspection and offer insights into integrating AI with symbolic modeling in practice.**

***Index Terms*—Multimodal data, AI, Symbolic representations, Conceptual modeling, Aligning embeddings.**

## I. Introduction

Artificial intelligence (AI) and conceptual modeling (CM) each offer distinct yet complementary strengths. AI models, such as generative pre-trained transformers, are capable of learning from vast amounts of unstructured data, capturing statistical patterns and contextual knowledge [1]. In contrast, conceptual models, such as BPMN, UML, ArchiMate, and SysML provide precise, human-understandable representations of domains through symbolic elements, structured relationships, and formal constraints [2]. As AI models grow more capable of interpreting real-world data in multiple formats—text, images, video, even audio—there's a disconnect between how AI understands a concept, and how that same concept is represented in modeling languages. Large AI models trained on massive datasets across modalities—often referred to as *foundation models*—learn to represent different types of inputs (like a paragraph or a photo) as high-dimensional vectors in a shared embedding space [1]. In that space, items that have *similar meaning* are placed close together, regardless of whether they were originally text, images, or something else. These embeddings effectively serve as the AI model's internal representation of knowledge. On the other hand, conceptual models encode structured knowledge through items like *classes*, *flows*, *actors*, and *stereotypes*. While symbolic, these elements can also be embedded using specialized encoders [3], or prompting techniques [4]–[6], that take into account their textual labels, graphical features, and meta-model constraints.

In this paper, we propose a method to align the embedding space of AI models with that of conceptual models, in order to support joint reasoning on tasks relevant to decision-makers and stakeholders. Aligning these representations enables new forms of collaboration between AI and CM. For example, it leads towards the possibility to recognize conceptual structures—such as processes or architectures—directly within video recordings or other multimodal observations, or comparison of formal norms (e.g., GDPR [7] rules) with observed behaviors or textual policies, facilitating conformance checking at scale. Ultimately, this may result in conceptual models more grounded in practice, and AI models more transparent in meaning. We evaluate our approach using a curated set of conceptual models from multiple modeling languages (UML, OntoUML, BPMN, ArchiMate, and SysML), alongside standard AI models for natural language and image understanding (large language models [8]). This setup enables us to investigate the alignment across heterogeneous input modalities and conceptual modeling paradigms. Our work is guided by the following research questions (RQ):

- **RQ1:** How can we formally define and compute a semantic alignment between conceptual model elements and AI foundation model embeddings derived from multimodal data?
- **RQ2:** To what extent does the enriched serialization of conceptual model terms (CMT) improve alignment with AI embeddings compared to raw natural language terms (NLT)?
- **RQ3:** How robust and generalizable is the proposed alignment approach across multiple modeling languages and domains (e.g., UML, BPMN, ArchiMate)?
- **RQ4:** What is the perceived usefulness, usability, and intention to adopt such alignment tools among modeling practitioners and researchers?

The remainder of the paper is structured as follows. Section II details our general perspective methodology. Section III presents the developed framework for *Aligning Embeddings of Knowledge and Symbols* (ALeKS). Section IV describes evaluation and reports results. We review related work in Section V and conclude in Section VI. Supplementary material[1] provides implementation and extended reports on the evaluation results.

## II. Methodology: The General Perspective

We propose a general perspective methodology to systematically align symbols from conceptual modeling artifacts with

[1]https://anonymous.4open.science/r/aligning_ai_cm-C7DE/

latent knowledge representations within AI models.

### A. Preliminaries

We define the principal constructs underlying our alignment methodology between conceptual modeling artifacts and AI-driven semantic spaces.

***Definition 1 (Embedding Space):*** Let $\mathbb{R}^d$ denote a $d$-dimensional real-valued vector space, where $d \in \mathbb{N}^+$. An ***embedding space*** $\mathcal{E} \subseteq \mathbb{R}^d$ is a continuous vector space in which elements—such as symbols, textual labels, or perceptual inputs—are represented as dense vectors.

***Definition 2 (Joint Embedding Space):*** A ***joint embedding space*** $\mathcal{E}$ is an embedding space in which heterogeneous modalities (e.g., text, vision, audio, structured data) are aligned into a shared semantic space. We can represent it as,

$$\mathcal{E} = \mathcal{E}_{\text{TEXT}} \cup \mathcal{E}_{\text{VISION}} \cup \mathcal{E}_{\text{AUDIO}} \cup \mathcal{E}_{\text{SYMBOLIC}} \subseteq \mathbb{R}^d.$$

***Definition 3 (Conceptual Model — CM):*** A ***conceptual model*** $CM$ is a structured knowledge artifact defined by elements such as entities, relationships, attributes, and behaviors. A collection of models is denoted by $C_{\text{CM}} = \{CM_1, CM_2, \ldots, CM_n\}$, each conforming to the formal semantics of a specific modeling language (e.g., UML, BPMN).

***Definition 4 (Natural Language Term — NLT):*** A ***natural language term*** is a textual element $a \in C_{\text{NLT}}$ extracted from a conceptual model, usually from attributes such as names or labels. The set $C_{\text{NLT}} = \{a_1, a_2, \ldots, a_k\}$ captures the linguistic surface forms used in the modeling domain.

***Definition 5 (Conceptual Model Term — CMT):*** A ***conceptual model term*** $a' \in C_{\text{CMT}}$ is an enriched version of a natural language term $a \in C_{\text{NLT}}$, extended with semantics from the modeling language (e.g., stereotypes, roles, taxonomies). $C_{\text{CMT}}$ includes domain-specific and structural knowledge encoded in conceptual artifacts.

***Definition 6 (Multimodal Data — MM): Multimodal data*** is a collection $C_{\text{MM}} = \{m(a_1), m(a_2), \ldots, m(a_k)\}$ of perceptual instances (e.g., images, videos, audio clips) associated with natural language terms $a_i \in C_{\text{NLT}}$. Each $m(a_i)$ is projected into the joint embedding space $\mathcal{E}$ using a multimodal encoder.

***Definition 7 (Semantic Shift):*** Given $x, x' \in \mathcal{E}$, the ***semantic shift*** from $x$ to $x'$ is the vector:

$$\vec{s}(x \to x') = E(x') - E(x),$$

where $E(\cdot)$ denotes the embedding function. The shift encodes how meaning evolves through semantic enrichment or modality transformation.

***Definition 8 (Shift Function):*** The ***shift function*** $s : \mathcal{E} \to \mathcal{E}$ maps between embeddings to quantify semantic transformation. Formally,

$$s(x \to x') = E(x') - E(x),$$

where $x$ is a natural language or perceptual term, and $x'$ its conceptualized or target form.

***Definition 9 (Set of Semantic Shifts — $S_{\text{E}}$):*** The total set of semantic shifts is defined as:

$$S_{\text{E}} = \{s(x \to x')\} \cup \{s(m(x) \to x)\} \cup \{s(m(x) \to m')\},$$

where $x \in C_{\text{NLT}}$, $x' \in C_{\text{CMT}}$, $m(x), m' \in C_{\text{MM}}$. This set captures how terms evolve through symbolic and perceptual transformation within the joint embedding space.

### B. The Methodology Overview

We detail the methodology steps as illustrated in Fig. 1. CM-STEPS process primarily conceptual models, while MM-STEPS multimodal data.

*a) Collection of Conceptual Models:* Our approach begins with a curated *Collection of Conceptual Models* ($C_{CM}$), which includes heterogeneous modeling diagrams from real-world domains, such as [9]. Each model serves as both a symbolic knowledge structure and a grounding point for semantic alignment. For illustration, we use a UML sequence diagram from the healthcare domain [10], as represented in Fig. 1, which models the procedural steps of a DNA collection homekit. This diagram specifies the user's interactions with a *test kit*, *tube*, and *biohazard bag*, and forms a narrative structure embedded with task semantics. These conceptual models, which can range from class diagrams to sequence diagrams, provide the foundational symbols (e.g., stereotypes, classes, attributes, messages) that represent domain knowledge.

*b)* CM-STEP 1*: Instantiating a Conceptual Model:* To process each model within the collection $C_{CM}$, we instantiate a data structure that explicitly represents its symbolic elements—this serves as the operationalization of a `foreach` function over the model collection. Each instantiation captures the syntactic and semantic features of the modeling language in use, enabling structured traversal and downstream processing.

*c)* CM-STEP 2*: Extracting Natural Language Terms:* From each instantiated model, we extract a *Collection of Natural Language Terms* ($C_{NLT}$) by identifying attributes that carry semantically meaningful text. This typically involves selecting the "name" attribute from modeling elements as observed in [9], [11]–[15], though the specific attribute may vary across modeling languages.

*d)* CM-STEP 3-4*: Tokenizing and Embedding Natural Language Elements:* Once the natural language terms ($C_{NLT}$) are collected, we proceed with tokenization through [8], wherein each textual term is decomposed into a sequence of tokens. This step prepares the text for numerical representation and aligns with standard NLP preprocessing workflows. Following tokenization, we transform each tokenized sequence into a high-dimensional vector space using embedding models. Specifically, we use `mxbai-embed-large` (334M parameters) and `all-minilm` (23M parameters) via the Ollama platform. These models are pre-trained for semantic embedding tasks and convert text into arrays of numbers, or embeddings, that encode the contextual meaning of the terms.

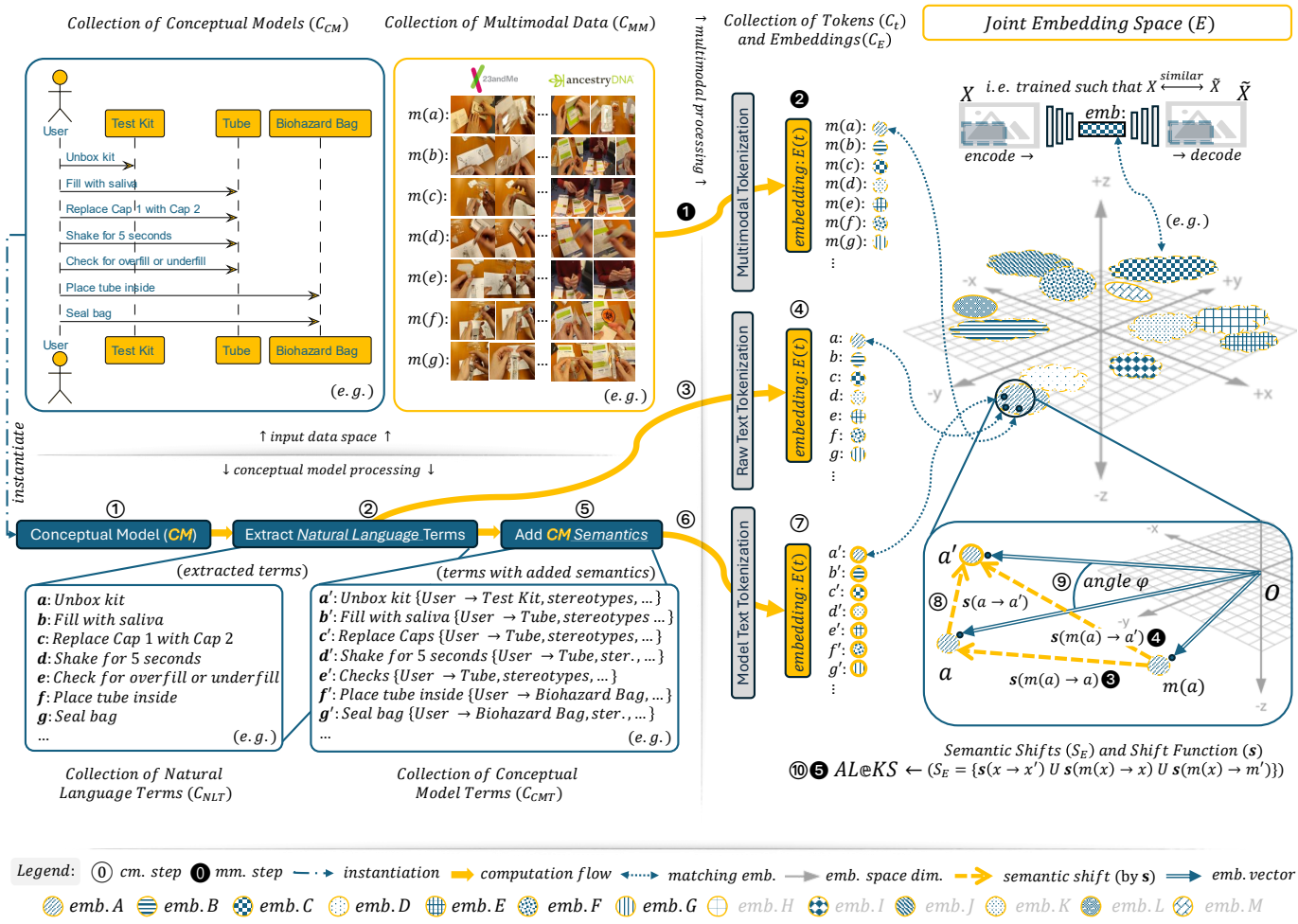


Fig. 1. **Overview of the proposed methodology.** The architecture aligns conceptual modeling terms with multimodal embeddings in a joint semantic space. Patterned regions (A–M) denote embedding (emb.) areas. This methodology constructs *ALeKS* pipeline, which unifies symbolic structures from conceptual models with perceptual data from multimodal corpora into a shared embedding space. The pipeline operates in parallel along two streams: one driven by conceptual modeling semantics (cm. steps) and the other by perceptual grounding via multimodal data (mm. steps).

*e)* CM-STEP 5-7*: Collection of Conceptual Model Terms:* With the *Collection of Natural Language Terms* ($C_{NLT}$) prepared and semantically embedded, we enrich this collection by incorporating structural semantics derived from the modeling language, resulting in the *Collection of Conceptual Model Terms* ($C_{CMT}$). For each term in $C_{NLT}$, we add contextual metadata such as stereotypes and types from [9], classifying whether an element represents, for instance, a Kind, Role, or Relator. Furthermore, model-specific annotations from diverse modeling repositories (e.g., [11]) are included, such as *eClass* identifiers and fully qualified names, which position elements within broader domain taxonomies. From enterprise architecture datasets (e.g., [12]), we incorporate higher-level organizational features like *ArchiMate* concept of *layer* and *type*. The resulting $C_{CMT}$ encapsulates both the natural language expressions and their conceptual modeling semantics. Once the $C_{CMT}$ is assembled, each enriched term—now combining natural language content with modeling semantics—undergoes the same tokenization and embedding process as described for $C_{NLT}$. This ensures consistency in how linguistic and structural elements are numerically represented. By treating both $C_{NLT}$ and $C_{CMT}$ with the same embedding pipeline, we maintain a shared representational space, which is used for reliable comparison, clustering, and semantic alignment in downstream tasks.

*f)* MM-STEP 1–2*: Collection of Multimodal Data:* To complement the symbolic and linguistic representations, we compile a *Collection of Multimodal Data* ($C_{MM}$), comprising images, audio, and video segments that are semantically linked to the conceptual model terms in $C_{NLT}$. Each modality is preprocessed and embedded using *ImageBind* [1], a foundation model that creates a unified embedding space for multiple data modalities. Unlike modality-specific encoders, ImageBind binds data across vision, audio, text, and thermal inputs into a shared semantic vector space through a contrastive training regime. This enables a direct comparison between, for instance, an image of a test kit and the term ”test kit” extracted from a UML sequence diagram. Tokenization in this context involves encoding non-textual data into fixed-length representations via modality-specific encoders, after which all embeddings are projected into a shared latent space using modality-agnostic transformers. Through this unified

embedding approach, $C_{MM}$ is mapped into the same vector space as $C_{NLT}$ and $C_{CMT}$, allowing semantic comparison across modalities. For each natural language term $a \in C_{NLT}$, we identify its multimodal counterpart $m(a) \in C_{MM}$, such as a corresponding image or video clip. This mapping enables the cross-modal embedding $E(m(a)) \in \mathbb{R}^d$, which can now be directly compared to the text-based embeddings $E(a)$ and $E(a')$ through vector operations in $\mathbb{R}^d$.

*g) Why Multimodal Data Matters:* Incorporating multimodal data allows us to generalize beyond the inherently biased and limited textual information extracted from conceptual models. While $C_{NLT}$ captures terms grounded in modeling practices, it reflects a narrow linguistic perspective shaped by domain-specific conventions. By retrieving the closest multimodal match $m(a)$ for each term $a \in C_{NLT}$—even if no explicit image or video exists in the original model—we tap into real-world representations that enrich and diversify semantic understanding. Although not every term will have a perfect multimodal counterpart, the use of a shared embedding space (e.g., ImageBind) ensures that the retrieved $m(a)$ is the most semantically relevant instance available. To maintain control and interpretability, we weight each multimodal contribution proportionally to its embedding similarity $\cos(E(a), E(m(a)))$, effectively modulating its influence based on relevance. This strategy enhances generalization by leveraging perceptual grounding while preserving alignment with the conceptual model's symbolic core.

*h)* CM-STEP 8*: Computing Semantic Shift from $a \to a'$:* To evaluate the impact of introducing conceptual modeling semantics, we define the semantic shift vector $\vec{s}(a \to a') = E(a') - E(a)$, where $E(a)$ and $E(a')$ are the respective embeddings of the natural language term $a$ and its enriched conceptual model form $a'$. This shift vector quantifies the transformation imposed by conceptual modeling, capturing how embedding semantics evolve when a term is recontextualized within formal modeling constructs (e.g., stereotypes, qualified names, abstraction layers). The vector $\vec{s}(a \to a')$ is used for tracing how symbolic enrichment repositions a term in the embedding space. As shown in Fig. 1, we interpret this semantic shift as a directed edge in $\mathbb{R}^d$ from $a$ to $a'$. If the modeling process adds significant semantic grounding, we expect the vector norm $\|\vec{s}(a \to a')\|$ to be non-trivial, and its direction to align with the conceptually meaningful transformation path. Importantly, $a$ and $a'$ remain semantically linked, and the shift encodes this transformation's nature—be it abstraction, disambiguation, or specialization.

*i)* CM-STEP 9*: Cosine Similarity as Alignment Indicator:* To measure how much semantic content is retained or realigned through modeling, we compute the cosine similarity

$$\cos(\varphi) = \frac{E(a) \cdot E(a')}{\|E(a)\| \|E(a')\|}$$

between the embeddings of $a$ and $a'$. This angle $\varphi$ serves as a metric of conceptual proximity. A smaller angle implies semantic preservation, whereas a larger angle implies a substantial shift due to modeling. Hence, $\varphi$ becomes an empirical indicator of how much the modeling semantics "reframe" the original natural term.

By calculating $\cos(\varphi)$ across all $(a, a')$ pairs, we can generate a statistical profile of modeling impact. Terms with high similarity are already semantically close to their conceptual representation, while those with low similarity indicate deeper semantic reinterpretation introduced by modeling languages or ontologies.

*j)* MM-STEP 3–4*: Multimodal Matching from $m(a) \to a'$:* In parallel to the conceptual path $a \to a'$, we compute the multimodal shift from a natural language term's multimodal representation $m(a)$ to its modeled form $a'$. This is denoted as $\vec{s}(m(a) \to a') = E(a') - E(m(a))$, to capture how much conceptual modeling repositions a term relative to its embodied (e.g., visual or auditory) representation. The embedding $E(m(a))$ arises from the shared embedding space, ensuring compatibility with $E(a')$.

For illustrative purposes, Fig. 1 visualize the vector $\vec{s}(m(a) \to a) = E(a) - E(m(a))$, which represents the shift between a multimodal instance and its textual form. For example, an image might evoke associations or affordances not present in the original textual label.

*k)* CM-STEP 10 + MM-STEP 5*: Defining Semantic Shifts $S_E$ and Shift Function $s$:* To integrate both the conceptual and multimodal perspectives, we define the full set of semantic shifts as:

$$S_E = \{s(x \to x')\} \cup \{s(m(x) \to x)\} \cup \{s(m(x) \to m')\}$$

where $x$ is a natural language term, $x'$ is its modeled version, $m(x)$ is its multimodal instantiation, and $m'$ is the modeled multimodal target. The shift function $s : \mathbb{R}^d \to \mathbb{R}^d$ maps embedding vectors to their target transformations, capturing both the symbolic (linguistic $\to$ conceptual) and perceptual (multimodal $\to$ linguistic or conceptual) evolution of meaning.

In practice, the three types of shifts—$s(x \to x')$, $s(m(x) \to x)$, and $s(m(x) \to m')$—form the semantic backbone for model evaluation. The magnitude and direction of each shift vector, as well as their aggregated statistics, offer insights into where alignment succeeds, where ambiguity persists, and where new semantics are introduced by modeling processes.

### C. Aligning AI Model's Knowledge and Conceptual Model's Symbols

Fig. 2 showcases two key conceptual events—*Shaking* and *SealingBag*—drawn from a UML activity model for a DNA collection test kit. Multimodal data and conceptual modeling semantics coalesce to generate semantically aligned embeddings, with alignment forces visualized via attractive (green) and repulsive (red) vectors.

*a) Conceptual Model Serialization:* To formalize the enrichment from natural language to conceptual modeling terms, we instantiate each conceptual model term using a controlled sentence template. We employ a templating mechanism to serialize conceptual model terms by enriching extracted natural language terms with conceptual modeling semantics.

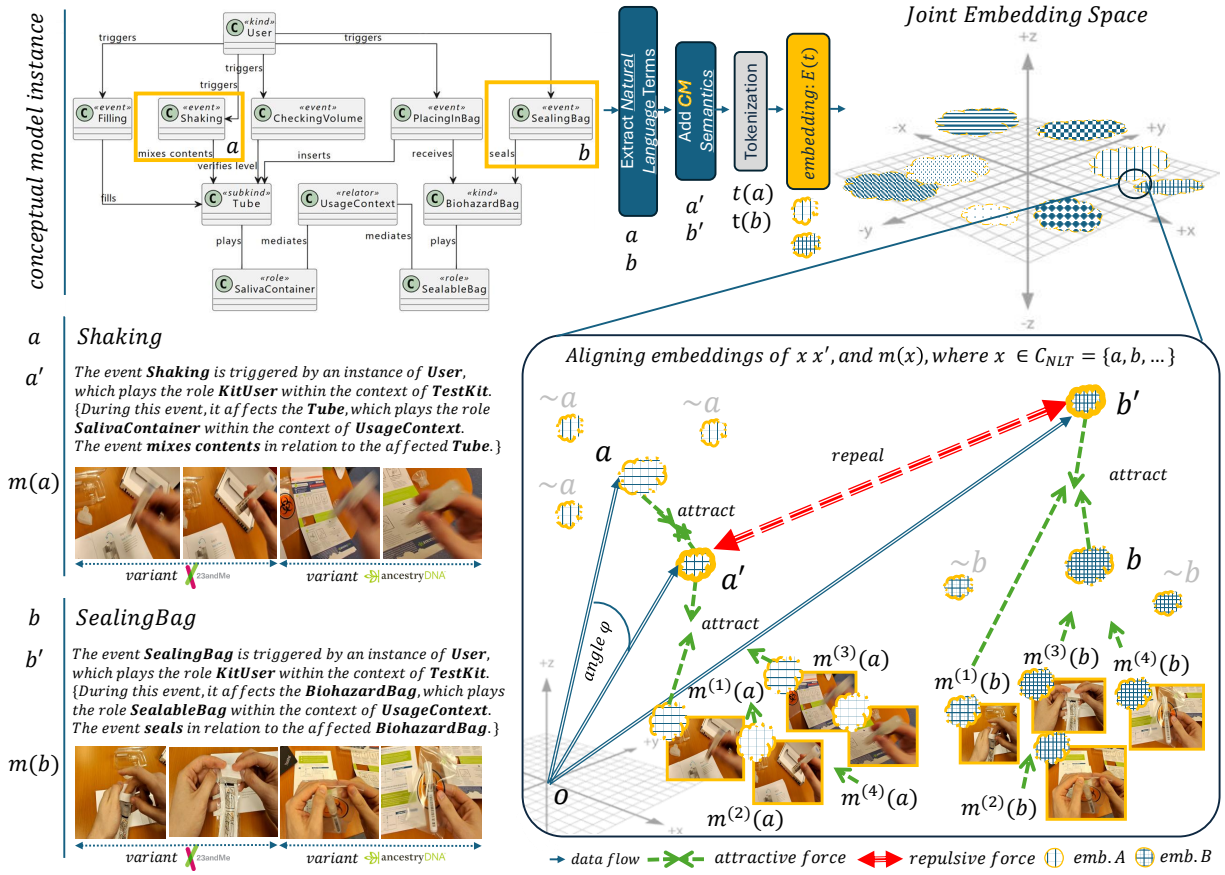


Fig. 2. Shows how natural language terms ($a$, $b$), their semantically enriched versions ($a'$, $b'$), and corresponding multimodal instances ($m(a)$, $m(b)$) are aligned in a joint embedding space. Two exemplar events—Shaking and SealingBag—from a conceptual model of a DNA test kit usage scenario are shown. *Note: The conceptual model in the upper left is scaled down to emphasize the overall layout. For a detailed view, please refer to the supplementary material.*

The following template is used for illustration in Fig. 2 to generate structured sentences:

> *The event <<**EventName**>> is triggered by an instance of <<**KindName**>>, which plays the role <<**RoleName**>> within the context of <<**RelatorName**>>. During this event, it affects the <<**AffectedSubkind**>>, which plays the role <<**RoleName**>> in the context of <<**findContext()**>>. The event <<**Function**>> in relation to the affected <<**AffectedSubkind**>>.*

The template serves as a domain-agnostic, syntactically consistent structure to convert symbolic conceptual model instances into textual descriptions. The placeholders (e.g., `EventName`, `KindName`, etc.) are dynamically replaced with values derived from the instantiated model. The section enclosed in braces ... denotes a `foreach` loop iterating over all `AffectedSubkind` elements that participate in the same event instance. For each such element, a corresponding sentence is generated to describe its involvement in the event. This approach ensures that the generated text reflects the multiplicity and contextual semantics present in the conceptual model.

As a result, each enriched term $a' \in C_{CMT}$ becomes a semantically grounded textual artifact that not only embeds linguistic content but also captures the structural and behavioral roles defined in the model. These serialized texts are then fed into our embedding pipeline, for the purpose of alignment with their corresponding natural language term $a$ and multimodal instance $m(a)$ within a shared vector space. This example of serialization grounds the term $a'$ with modeling semantics, explicitly enriching its embedding relative to the plain term $a$, and similarly, producing $b'$ as the semantically contextualized version of $b$.

*b) Aligning Embeddings in the Shared Space:* Natural language terms ($a$, $b$), their conceptually enriched forms ($a'$, $b'$), and multimodal instances ($m(a)$, $m(b)$) are jointly embedded as illustrated in Fig. 2. Conceptual enrichment pulls each $a'$ and $b'$ closer to their corresponding natural term ($a$, $b$) via attractive forces, while simultaneously pushing them away from unrelated terms (e.g., $a' \rightarrow \text{repel}(b')$).

*c) Semantic Shifts:* Each event ($a$, $b$) is paired with a collection of multimodal data entries $m(a)$ and $m(b)$, composed of image sequences sourced from real-world DNA test kit usage. These images span multiple domains (e.g., 23andMe, AncestryDNA), highlighting visual variance across kits while maintaining semantic coherence. The multimodal embedding model projects each $m^{(i)}(a)$ and $m^{(j)}(b)$ into the same space as their symbolic counterparts, and the figure shows a clustering behavior around $a'$ and $b'$. Semantic transitions are

geometrically encoded via vector shifts. These shifts—$s(a \rightarrow a')$, $s(m(a) \rightarrow a')$—are drawn as arrows within the joint embedding space. By computing and visualizing angles $\varphi$ between $a$ and $a'$, or distances between $m(a)$ and $a'$, we gain semantic similarity insight.

## III. Proof of Concept: Aligned Embeddings of Knowledge and Symbols (ALeKS)

In this section, we present a proof-of-concept implementation of the **Aligned Embeddings of Knowledge and Symbols (ALeKS)** framework. ALeKS is designed to bridge symbolic conceptual models and machine-readable embeddings, enabling downstream tasks such as similarity search, model comparison, and intelligent querying over conceptual structures. Our implementation specifically targets the OntoUML modeling language, known for its ontologically grounded constructs and formal semantics.

We begin by demonstrating the serialization procedure, which transforms OntoUML models—comprised of modeling terms and properties—into structured natural language sentences. This step is critical to ensuring the symbolic data can be meaningfully embedded. Following serialization, we describe the alignment mechanism, which trains a joint vector space where symbolic and statistical representations are semantically aligned.

The serialization phase handles two major model constructs: *terms* (e.g., classes or entities) and *properties* (e.g., attributes or associations). Each element is decomposed into a set of human-readable, formally structured sentences that preserve the semantics encoded in the modeling language. This transformation enables the use of language models and embedding techniques to learn aligned representations.

### A. Conceptual Model Serialization: Terms and Properties

The serialization procedure converts raw modeling elements into readable and semantically meaningful sentences. This structured textual representation acts as an intermediate form between symbolic models and their embedding. Below, we detail each step of the serialization process for OntoUML *terms* and *properties*, illustrating them with sentence patterns.

*Parent Term Linkage:* Each property $p$ is linked to its parent modeling element $n$, typically a class or entity:

PROPERTY $n$ (id: ID_N) BELONGS TO $m$ (id: ID_M).

This sentence formalizes the containment relationship between a property and its owning element, preserving model hierarchy.

*a) Type Declaration:* If a property has an explicitly declared type, this is rendered into:

PROPERTY $n$ (id: ID_N) IS OF TYPE $T$.

This identifies the kind of value the property holds, such as a QUANTITY, IDENTIFIER, or other domain concept.

*b) Boolean Flags:* Boolean modeling characteristics—such as derivation, immutability, or ordering—are recorded individually:

PROPERTY $n$ (id: ID_N) IS DERIVED. PROPERTY $n$ (id: ID_N) IS READ-ONLY. PROPERTY $n$ (id: ID_N) IS ORDERED.

These aspects encode essential behavioral semantics for the property.

*c) Cardinality:* Cardinality constraints, if defined, are serialized as:

PROPERTY $n$ (id: ID_N) HAS CARDINALITY [MIN..MAX].

This supports reasoning about optionality and multiplicity.

*d) Property Type (Nested Dictionary):* If the property includes a nested dictionary specifying a TYPE, we render it as:

PROPERTY $n$ (id: ID_N) HAS PROPERTY TYPE $T$.

This captures additional typing information not reflected in the main declaration.

*e) Aggregation Kind:* When an AGGREGATIONKIND is specified:

PROPERTY $n$ (id: ID_N) HAS AGGREGATION KIND $K$.

This indicates whether the property is shared, composite, or neither.

*f) Property Assignments:* Metadata or configuration values grouped as property assignments are preserved:

PROPERTY $n$ (id: ID_N) HAS PROPERTY ASSIGNMENTS $\{$k:v, ...$\}$.

This ensures auxiliary data is not lost in transformation.

*g) Stereotype:* If a stereotype is assigned to a property, it is recorded as:

PROPERTY $n$ (id: ID_N) HAS STEREOTYPE $S$.

Stereotypes such as FUNCTIONALPROPERTY or QUALIFIER add semantic precision.

*h) Subsetted Properties:* When a property specializes other properties, the subsetted relationship is captured as:

PROPERTY $n$ (id: ID_N) HAS SUBSETTED PROPERTIES $\{p_1, p_2, \ldots\}$.

*i) Redefined Properties:* Redefinitions are encoded similarly:

PROPERTY $n$ (id: ID_N) HAS REDEFINED PROPERTIES $\{r_1, r_2, \ldots\}$.

*Serialization of Terms:* For each modeling term $n$, we begin with its identification:

TERM $n$ (id: ID_N).

*j) Parent-Term Relationship:* If a term belongs to another model element:

TERM $n$ (id: ID_N) BELONGS TO $m$ (id: ID_M).

*k) Type Declaration:* The type of the term is declared explicitly:

TERM $n$ (id: ID_N) IS OF TYPE $T$.

*l) Boolean Flags:* Boolean attributes on terms, such as abstraction or derivation, are expressed as:

TERM $n$ (id: ID_N) IS ABSTRACT. TERM $n$ (id: ID_N) IS DERIVED.

*m) Stereotype and Property Assignments:* Stereotypes and auxiliary properties are rendered as:

TERM $n$ (id: ID_N) HAS STEREOTYPE $S$. TERM $n$ (id: ID_N) HAS PROPERTY ASSIGNMENTS $\{$k:v, ...$\}$.

### B. Embedding Alignment and Dimensionality Reduction

Once conceptual modeling elements are serialized into CMT and paired with their corresponding NLT, we align their representations in a shared vector space. The alignment process begins with reducing the dimensionality of high-dimensional embeddings to enable qualitative visual inspection and statistical analysis.

*a) Dimensionality Reduction via t-SNE:* To visualize how NLT and CMT embeddings relate in latent space, we apply t-distributed stochastic neighbor embedding (t-SNE). First, we take two sets of aligned vectors: NLT_EMBEDDING and CMT_EMBEDDING, and vertically stack them into a single matrix for joint processing.

*b) Pretrained Embeddings Models:* To populate the aligned space with semantic vectors, we employ an embedding model via the `ollama` interface. Specifically, we use the ALL-MINILM model to embed natural language descriptions of OntoUML models.

### C. Implementation

To operationalize the semantic alignment methodology introduced above, we implemented a supervised learning pipeline that learns to map NLT embeddings to their corresponding CMT embeddings in a two-dimensional semantic space.

*Data Preparation and Preprocessing:* We begin by loading two-dimensional embeddings extracted from a heterogeneous collection of conceptual models, specifically [16]. The input features $\mathbf{X} \in \mathbb{R}^{n\times 2}$ represent NLT coordinates, while the target outputs $\mathbf{Y} \in \mathbb{R}^{n\times 2}$ correspond to the semantically enriched CMT coordinates. Both sets are standardized using `StandardScaler` from scikit-learn to ensure numerical stability and uniform learning behavior across dimensions.

*Model Architecture:* To model the alignment function $f : \mathbb{R}^2 \to \mathbb{R}^2$, we implemented a deep multilayer perceptron (MLP) using PyTorch. The architecture consists of five fully connected layers with ReLU activation functions and gradually decreasing dimensionality. The model consists of a sequence of fully connected layers and ReLU activations. It begins with a `Linear(2, 128)` layer, followed by a `ReLU` activation. This is followed by another `Linear(128, 128)` layer with a `ReLU`, then a `Linear(128, 64)` layer and another `ReLU`. Next, it includes a `Linear(64, 32)` layer followed by a `ReLU`, and finally a `Linear(32, 2)` layer that outputs the final result.

*Alignment Loss Function:* We use loss function which combines Mean Squared Error (MSE) with Cosine Embedding Loss to encourage both positional accuracy and semantic directional alignment:

$$\mathcal{L}_{\text{align}} = \alpha \cdot \mathcal{L}_{\text{MSE}} + (1 - \alpha) \cdot \mathcal{L}_{\text{cosine}},$$

where $\alpha = 0.7$ balances the geometric and angular components of the loss. This formulation takes into account that predictions are not only numerically close to the target embeddings but also semantically aligned in direction within the shared vector space.

*Training Procedure:* For robust generalization, we employ 5-fold cross-validation using the `KFold` strategy with shuffling. In each fold, the training set is passed through a batch-wise `DataLoader` with a batch size of 32. The model is trained for 200 epochs per fold using the Adam optimizer with a learning rate of 0.001. During training, we monitor the running loss every 20 epochs to confirm convergence behavior.

*Model Selection and Evaluation:* After each fold, the trained model is evaluated on its respective validation set using MSE. The model achieving the lowest validation MSE across all folds is retained as the best-performing instance. The average MSE across folds is also reported as a measure of overall stability. Once selected, the best model is saved to disk using PyTorch's serialization utility.

*Post-Training Inference and Visualization:* We deploy the best model on the entire dataset to generate predictions for all input NLT embeddings. These predictions are then inverse-transformed back into the original coordinate space. To qualitatively assess alignment performance, we visualize the predicted embeddings against their actual CMT counterparts using directional arrows. Each arrow originates at the predicted point and terminates at the ground truth, thereby visualizing the semantic shift vector $\vec{s}(a \to a')$ described in Section II-A.

### D. Visual Tool for Alignment Inspection and Verification

To support the manual inspection and interpretation of alignment results, we developed a visual comparison tool (previewed in Fig. 3). The tool enables users to explore semantic alignment between NLT, CMT, and large language model (LLM) representations within a unified interface. It allows both developers and domain experts to verify the quality of the semantic shift functions and gain insights into embedding behavior.

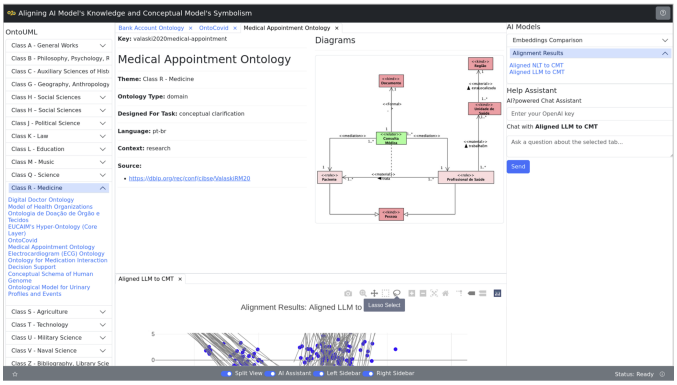


Fig. 3. Screenshot of the implemented visual inspection tool for alignment analysis. The user can browse conceptual model ontologies (left), view diagram semantics (top center), and explore alignment results (bottom). The AI comparison module (right) provides tools to inspect embeddings, check alignment quality, and interactively verify model serializations across NLT, CMT, and AI representations. The example shows a model from [17].

The interface is composed of three main panels: (i) a structured browser for conceptual models categorized by domain from [16] (e.g., medicine, law, science), (ii) a visualizer for individual conceptual model diagrams, and (iii) an AI

model comparison module. This last module supports the selection of embedding alignment types, including *NLT* $\rightarrow$ *CMT* and *LLM* $\rightarrow$ *CMT*, and provides interactive plots to inspect embedding proximity and vector shifts. We use this tool in two key ways: first, to conduct manual verification of model-generated alignments by inspecting whether predicted CMT vectors correspond to their expected symbolic roles and locations; second, to enable exploratory analysis of how symbolic terms are recontextualized across different ontologies and embedding strategies. The interface also includes a conversational assistant that helps explain or inspect selected terms, further facilitating the alignment audit process. To support users in navigating the alignment process and deriving meaningful insights, we developed a custom *GPT*-based assistant integrated into the tool.

## IV. Evaluation

To validate our proposed methodology, we conduct a multi-perspective evaluation covering alignment quality, generalizability across modeling languages, human-centered assessments via the *Technology Acceptance Model* (TAM) [18], and system-level performance and scalability.

### A. Evaluation of the Alignment Model – AI Knowledge and Symbolism

To test if *ALeKS* leads toward reconstruction of symbolic knowledge from enriched conceptual model terms (CMT), we compare raw AI-generated summaries of OntoUML models to their predicted counterparts generated from serialized CMT variants. Specifically, we apply the following prompt using `llama3.3` to each **non-serialized** model in the OntoUML/UFO Catalogue:

```
prompt_template =    """  OntoUML model:
serialization
Task: Analyze the OntoUML model
provided in the serialization and
generate a concise description that
outlines how elements are connected.
Instructions:
Go through all internal element
identifiers (IDs) and identify the
relationships between them.
For each connection, preserve and
use the natural-language names of the
involved elements.
Summarize the connections between
classifiers (e.g., classes, relators,
roles) and describe their nature
(e.g., mediations, generalizations,
associations).
Output:
A structured description of the
model's connectivity, focusing only
on inter-element relationships.
Use OntoUML-relevant terminology (e.g.,
"relator mediates between", "role
played by", "generalization from X
to Y").
Avoid listing raw IDs, but ensure every
named element mentioned is traceable
through its name.
The output should serve as a compact
summary of the model's structure
through its relationships. """
```

We then compute cosine similarity between the emebeddings of these generated descriptions and the embeddedings of ground-truth CMT serializations. Our evaluation reveals a significant increase in semantic alignment when using CMT-enriched prompts, showing higher cosine similarity with ground-truth model annotations than the raw NLT terms alone. This supports the hypothesis that CMT serves as an effective bridge between AI knowledge representations and symbolic model structures.

*a) Alignment Accuracy.:* The average cosine similarity between the predicted embeddings and the true CMT representations was notably high at *0.94*, indicating that *ALeKS* effectively captures the structural semantics encoded in formal modeling templates.

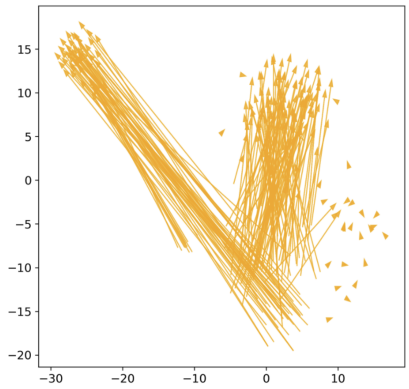


Fig. 4. **2D projection of semantic shifts in OntoUML/UFO Catalgoue [9].** Each arrow represents the vector shift $s(a \rightarrow a')$ from a natural language term (NLT) to its corresponding conceptual model term (CMT), based on OntoUML data. Embeddings are projected to 2D using.

Fig. 4 visualizes the semantic transformation from NLT to enriched conceptual modeling terms (CMT) for the OntoUML/UFO Catalogue. Each arrow represents a shift vector $s(a \rightarrow a')$ projected into a 2D embedding space using PCA. The left cluster (NLT) contains more linguistically raw representations, while the right cluster (CMT) shows tightly grouped structurally enriched terms. The uniform directional flow from left to right reveals recontextualization: as terms are enriched with OntoUML semantics (e.g., kinds and roles), they move into a different semantical region of the space.

### B. Cross-Conceptual Modeling Language Observations

To assess generalizability across modeling languages, we evaluate the semantic shift from NLT to their enriched conceptual modeling terms (CMT) across five conceptual models repositories: *ontouml* [9], *modelset* [11], *eamodelset* [12], *hdbpmn* [14], and *SysML PhS* [15]. These shifts are visualized in Fig. 5, showing consistent directional movement in the embedding space for each model collection.

In each case, NLT points (blue circles) shift toward CMT points (orange squares), indicating that conceptual enrichment

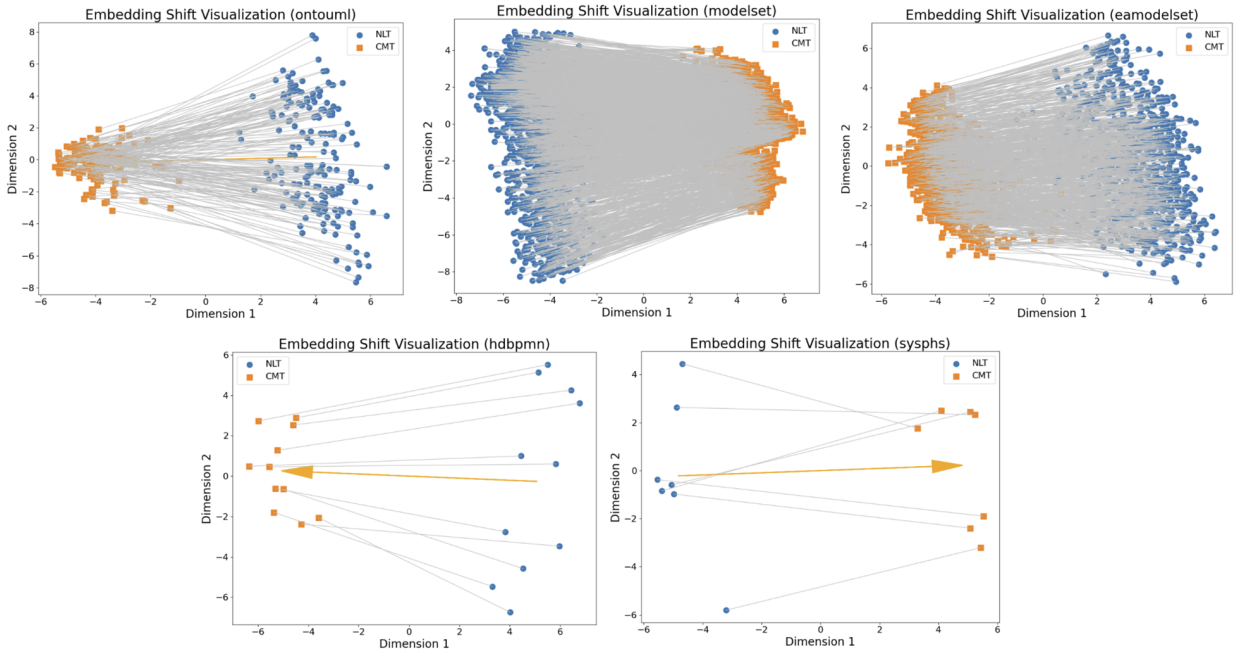


Fig. 5. **Embedding shift visualization across modeling languages.** Semantic shifts from natural language terms (NLT) to conceptual model terms (CMT) are shown across six repositories: *ontouml* [9], *modelset* [11], *eamodelset* [12], *hdbpmn* [14], and *SysML PhS* [15]. Each arrow represents a semantic shift vector $s(NLT \rightarrow CMT)$ in the shared embedding space.

introduces a measurable semantic transformation. For high-density model repositories (top row), shift vectors exhibit clustering, whereas for sparse or abstract models (bottom row), the shift vectors are fewer but semantically distinct. This provides evidence that the method may apply across languages with different granularity and modeling purposes.

### C. Technology Acceptance Model Evaluation

We further evaluated the alignment tool and the AI support system through a qualitative *Technology Acceptance Model* (TAM) [18] study. This involved interviews and surveys with a cohort of 10 participants: 2 PhD holders in Conceptual Modeling, 3 PhD students, and 5 advanced Bachelor students from the same domain. Each participant interacted with the tool and evaluated its usefulness and ease of use.

TAM defines three key constructs: *Perceived Usefulness* (U), *Perceived Ease of Use* (E), and *Behavioral Intention to Use* (BI). To assess these, we asked questions such as:

- *"Does the system help you better understand model semantics through AI-generated summaries?"* (U)
- *"How difficult was it to interact with the embedding comparison interface?"* (E)
- *"Would you use this tool to explain models in your teaching or research?"* (BI)

Results showed high agreement on perceived usefulness from experts, who noted the value of recontextualized model descriptions. Bachelor students scored slightly lower on ease of use, suggesting that onboarding might benefit from improved interface design or guidance. The overall behavioral intention was positive, especially when participants understood the value of CMT-based enrichment and its alignment with perceptual data.

### D. Discussion

*a) Scalability and Performance:* The embedding pipeline, including multimodal alignment and CMT enrichment, was executed on a single NVIDIA A40 GPU with 48GB VRAM. This setup supports processing of thousands of terms across multiple modeling languages in parallel. Runtime performance scales linearly with the number of model elements, making it feasible to apply at enterprise or institutional scale.

*b) Cost of Operation:* As the architecture relies solely on locally hosted models—embedding via Ollama and multimodal alignment through ImageBind—there are no recurring cloud API costs. This significantly reduces operational overhead and allows researchers and educators to use the tool offline.

*c) Privacy and Data Control:* All computation is executed locally, ensuring full control over sensitive model artifacts, especially in regulated industries like healthcare and defense. This eliminates concerns associated with sending confidential data to external AI APIs.

*d) Evaluation on Multimodal Data:* To assess the multimodal alignment, we integrated ImageBind [1] embeddings, which support six modalities—image, text, audio, depth, thermal, and inertial measurement unit (IMU). Preliminary results on the DNA home kit scenario from [10] yielded a retrieval accuracy of 0.56 for matching modalities. While promising, performance remains limited due to the absence of a dedicated benchmark set. Future work will focus on expanding multimodal datasets and refining evaluation protocols to better capture alignment quality across sensory channels.

*e) Limitations and Threats to Validity:* One limitation is the dependency on high-quality modeling artifacts; poor

naming or inconsistent stereotypes in conceptual models can reduce alignment accuracy. Furthermore, while cosine similarity offers a proxy for alignment, it may not fully capture conceptual equivalence. Threats to validity include limited domain diversity and potential annotation bias in manually curated model collections.

*f) Response to Research Questions.:* Our evaluation addresses all four research questions. For **RQ1**, we introduced formal definitions of embedding spaces, semantic shifts, and alignment functions, and implemented a shift-based mapping between conceptual model elements and AI embeddings. Regarding **RQ2**, experiments showed that enriched conceptual model terms (CMT) yield significantly higher cosine similarity scores (average 0.94) compared to raw natural language terms (NLT), demonstrating that semantic enrichment improves alignment. For **RQ3**, we validated our approach across six modeling languages and repositories (OntoUML, UML, BPMN, ArchiMate, SysML), confirming consistent semantic shifts and robust performance across heterogeneous domains. In response to **RQ4**, a Technology Acceptance Model study involving researchers and students revealed strong agreement on the tool's usefulness and usability, with participants expressing high intention to adopt the alignment interface in both research and teaching contexts.

## V. Related Work

Our work—grounding AI foundation model embeddings in conceptual modeling semantics—intersects with developments in (1) AI and conceptual modeling alignment, and (2) conceptual model-oriented alignment and simulation.

### A. AI and Conceptual Modeling Alignment

*1) AI-Augmented Conceptual Modeling:* Efforts to connect AI with conceptual models often focus on encoding model structures into machine-readable representations. [3] offers a taxonomy of such encoding strategies, complemented by mappings from [19] on integrating AI into modeling workflows. Cognitive framing is emphasized by [20] and [21], who envision conceptual models as interpretable layers for AI-enhanced enterprises. Relatedly, [2], [22] stress AI-assisted model complexity management. Prompt-based strategies for model completion and extraction are gaining traction: [4] applies few-shot learning for auto-completion, while [5], [6] use LLMs for extracting and reasoning over process models. Unlike these, our *ALeKS* framework introduces a semantic shift mechanism aligning multimodal AI embeddings with symbolic CMT structures in a shared vector space.

*2) Multimodal Foundation Models:* Multimodal foundation models like *ImageBind* [1] embed text, images, and more into a unified space, enabling cross-modal reasoning. Extensions to video generation and reasoning are seen in *Sora* [23], [24] and in domain-specific applications like neurosurgery [25]. From a language model perspective, [26] discusses reasoning incentivization in multimodal agents. While these models capture perceptual semantics, they often lack structural grounding. *ALeKS* enhances multimodal embeddings with conceptual modeling constraints—e.g., roles, kinds, relators—enabling structured AI understanding rooted in formal semantics.

### B. Conceptual Model-Oriented Alignment and Simulation

AI integration in process modeling includes conformance checking [27], simulation [28], and predictive modeling. Knowledge graphs [29], simulation trust metrics [30], and contextualization [31] all enhance process analysis. Tools like ProM [32] and frameworks like COBIT [33] underscore structured alignment. Recent multimodal work explores bridging sensor and video data, or multimodal data in general with models, enabling materialization of the conceptual models (like UML) as image or audio [34], and learning the stakeholder specific representation of models [35]. Our method complements these by explicitly projecting both symbolic and perceptual data into a shared semantic space consistent with languages like BPMN, SysML, and UML.

*1) Modeling Tools, Usability, and Human Factors:* Human-centric modeling approaches focus on accessibility and visual intuitiveness. [36] and [37] discuss usability in domain-specific modeling. Visualization taxonomies [38] demonstrate the value of interactive support. Our dashboard for semantic shift visualization contributes to this thread by making AI–model alignment interpretable. In line with [18], we aim to increase perceived usefulness and ease of use for practitioners engaging with AI-augmented modeling tools.

### C. Positioning Our Approach

*ALeKS* advances the field by unifying AI embeddings, multimodal perception, and conceptual model semantics into a joint representation. It differs from prior work by treating modeling semantics as first-class citizens when creating in the embedding process.

## VI. Conclusion

We presented *ALeKS*—Aligned Embeddings of Knowledge and Symbols—a novel framework for aligning the internal representation spaces of AI foundation models with the symbolic structures of conceptual models. By formalizing semantic shifts between natural language terms, multimodal data, and enriched conceptual model constructs, we demonstrated how to create a joint embedding space that enables meaningful comparisons and reasoning across modalities.

Our implementation integrates ontology-aware serialization, multimodal embedding models, and a neural alignment function trained on real-world modeling repositories. Visual and interactive tools were developed to inspect alignment behavior, and empirical evaluation showed strong alignment accuracy (average cosine similarity of 0.94) and generalization across modeling languages.

Through a human-centered TAM study, we also validated the approach's perceived usefulness and ease of adoption in educational and research contexts. Future work will explore dynamic alignment in evolving models, use-case-specific fine-tuning, and extending the pipeline to real-time semantic grounding modeling environments.

## References


[1] R. Girdhar, A. El-Nouby, Z. Liu, M. Singh, K. V. Alwala, A. Joulin, and I. Misra, "Imagebind: One embedding space to bind them all," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 15 180–15 190.

[2] H. A. Proper, B. van Gils, and K. Haki, *Final Conclusions and Outlook*. Cham: Springer Nature Switzerland, 2023, pp. 311–314.

[3] S. J. Ali, A. Gavric, H. Proper, and D. Bork, "Encoding conceptual models for machine learning: A systematic review," pp. 562–570, 2023.

[4] M. B. Chaaben, L. Burgueño, and H. Sahraoui, "Towards using few-shot prompt learning for automating model completion," in *2023 IEEE/ACM 45th International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, 2023, pp. 7–12.

[5] J. Neuberger, L. Ackermann, H. van der Aa, and S. Jablonski, "A universal prompting strategy for extracting process model information from natural language text using large language models," in *International Conference on Conceptual Modeling*. Springer, 2024, pp. 38–55.

[6] A. Rebmann, F. D. Schmidt, G. Glavaš, and H. van Der Aa, "Evaluating the ability of llms to solve semantics-aware process mining tasks," in *2024 6th International Conference on Process Mining (ICPM)*. IEEE, 2024, pp. 9–16.

[7] "Regulation (eu) 2016/679 of the european parliament and of the council," https://eur-lex.europa.eu/eli/reg/2016/679/oj, April 2016, official Journal of the European Union, L 119, 4.5.2016, p. 1–88.

[8] "Ollama: Run large language models locally," https://ollama.com, accessed: 2025-04-03.

[9] P. P. F. Barcelos, T. P. Sales, M. Fumagalli, C. M. Fonseca, I. V. Sousa, E. Romanenko, J. Kritz, and G. Guizzardi, "A fair model catalog for ontology-driven conceptual modeling research," in *International Conference on Conceptual Modeling*. Springer, 2022, pp. 3–17.

[10] A. Gavric, D. Bork, and H. Proper, "Enriching business process event logs with multimodal evidence," in *The 17th IFIP WG 8.1 Working Conference on the Practice of Enterpris Modeling (PoEM)*, 2024.

[11] J. A. H. López, J. L. Cánovas Izquierdo, and J. S. Cuadrado, "ModelSet: a dataset for machine learning in model-driven engineering," *Softw. Syst. Model.*, vol. 21, no. 3, pp. 967–986, 2022. [Online]. Available: https://doi.org/10.1007/s10270-021-00929-3

[12] P.-L. Glaser, E. Sallinger, and D. Bork, "Ea modelset–a fair dataset for machine learning in enterprise modeling," in *IFIP Working Conference on The Practice of Enterprise Modeling*. Springer, 2023, pp. 19–36.

[13] C. Community, "Bpmn for research," https://github.com/camunda/bpmn-for-research, 2020, accessed: 2025-04-03.

[14] B. Schäfer, H. van der Aa, H. Leopold, and H. Stuckenschmidt, "Sketch2bpmn: Automatic recognition of hand-drawn bpmn models," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2021.

[15] C. A. Manion, C. A. Manion, C. Bock, and R. Barbau, *Physical Component Libraries for SysPhS Modeling and Simulation in Manufacturing*. US Department of Commerce, National Institute of Standards and Technology, 2023.

[16] S. Haridy, R. M. Ismail, N. Badr, and M. Hashem, "Ontouml conceptual model for egyptian e-government: Student and traveler case studies," in *2021 Tenth International Conference on Intelligent Computing and Information Systems (ICICIS)*. IEEE, 2021, pp. 506–511.

[17] J. Valaski, S. S. Reinehr, and A. Malucelli, "Avaliação de abordagem centrada em modelos conceituais em ontouml para apoiar a derivação de requisitos funcionais." in *CIbSE*, 2020, pp. 264–277.

[18] F. D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS quarterly*, pp. 319–340, 1989.

[19] D. Bork, S. J. Ali, and B. Roelens, "Conceptual modeling and artificial intelligence: A systematic mapping study," 2023.

[20] V. Kulkarni, S. Reddy, T. Clark, and H. Proper, *The AI-Enabled Enterprise*. Cham: Springer International Publishing, 2023, pp. 1–12.

[21] B. v. Gils and H. A. Proper, *Next-Generation Enterprise Modeling*. Cham: Springer Nature Switzerland, 2023, pp. 279–305.

[22] H.-G. Fill, F. Härer, I. Vasic, D. Borcard, B. Reitemeyer, F. Muff, S. Curty, and M. Bühlmann, "Cmag: A framework for conceptual model augmented generative artificial intelligence," 12 2024.

[23] B. Lin, Y. Ge, X. Cheng, Z. Li, B. Zhu, S. Wang, X. He, Y. Ye, S. Yuan, L. Chen *et al.*, "Open-sora plan: Open-source large video generation model," *arXiv preprint arXiv:2412.00131*, 2024.

[24] Y. Liu, K. Zhang, Y. Li, Z. Yan, C. Gao, R. Chen, Z. Yuan, Y. Huang, H. Sun, J. Gao *et al.*, "Sora: A review on background, technology, limitations, and opportunities of large vision models," *arXiv preprint arXiv:2402.17177*, 2024.

[25] A. A. Mohamed and B. Lucke-Wold, "Text-to-video generative artificial intelligence: sora in neurosurgery," *Neurosurgical Review*, vol. 47, no. 1, p. 272, 2024.

[26] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[27] J. Mendling, H. A. Reijers, M. La Rosa, and M. Dumas, "Fundamentals of business process management." in *GI-Jahrestagung*. Springer, 2013, p. 157.

[28] W. M. Van Der Aalst, "Business process simulation survival guide," in *Handbook on business process management 1: Introduction, methods, and information systems*. Springer, 2014, pp. 337–370.

[29] K. Amiri Elyasi, D. Sola, C. Meilicke, H. van der Aa, and H. Stuckenschmidt, "Knowledge graph completion for activity recommendation in business process modeling," *KI-Künstliche Intelligenz*, pp. 1–15, 2024.

[30] D. Chapela-Campa, I. Benchekroun, O. Baron, M. Dumas, D. Krass, and A. Senderovich, "Can i trust my simulation model? measuring the quality of business process simulation models," in *International Conference on Business Process Management*. Springer, 2023, pp. 20–37.

[31] T. Grisold, H. van der Aa, S. Franzoi, S. Hartl, J. Mendling, and J. Vom Brocke, "A context framework for sense-making of process mining results," in *2024 6th International Conference on Process Mining (ICPM)*. IEEE, 2024, pp. 57–64.

[32] A. Rozinat, R. S. Mans, M. Song, and W. M. van der Aalst, "Discovering simulation models," *Information systems*, vol. 34, no. 3, pp. 305–327, 2009.

[33] Information Systems Audit and Control Association, *COBIT 2019 Framework: Governance and Management Objectives*. ISACA, 2018. [Online]. Available: https://www.isaca.org/resources/cobit

[34] A. Gavric, D. Bork, and H. Proper, "How does uml look and sound? using ai to interpret uml diagrams through multimodal evidence," in *43rd International Conference on Conceptual Modeling (ER)*, 2024.

[35] ——, "Stakeholder-specific jargon-based representation of multimodal data within business process," in *Companion Proceedings of the 17th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling (PoEM Forum 2024)*, 2024.

[36] D. Bork, C. Schrüffer, and D. Karagiannis, "Intuitive understanding of domain-specific modeling languages: Proposition and application of an evaluation technique," in *Conceptual Modeling - 38th International Conference, ER 2019, Salvador, Brazil, November 4-7, 2019, Proceedings*, ser. Lecture Notes in Computer Science, A. H. F. Laender, B. Pernici, E. Lim, and J. P. M. de Oliveira, Eds., vol. 11788. Springer, 2019, pp. 311–319.

[37] A. Sarioglu, H. Metin, and D. Bork, "Accessibility in conceptual modeling - A systematic literature review, a keyboard-only UML modeling tool, and a research roadmap," *Data Knowl. Eng.*, vol. 158, p. 102423, 2025.

[38] D. Bork and G. De Carlo, "An extended taxonomy of advanced information visualization and interaction in conceptual modeling," *Data & Knowledge Engineering*, vol. 147, p. 102209, 2023.