Lingua-Graph: A Unified Representation of Cross-Task Common Substructures for Analytic Language Processing

Mingming Sun[1,*] **Runze Jiang**[1,2] **Zhangchenxi Zhu**[1,3]
Minlong Peng[4] **Yunfeng Cai**[5]
[1]AGI Lab, BIMSA [2]School of Mathematical Sciences, Peking University
[3]Institute of Linguistics, CASS [4]Tencent Hunyuan [5]BIMSA
sunmingming@bimsa.cn 2501210087@stu.pku.edu.cn
zhuzhangchenxi@ucass.edu.cn crashpeng@tencent.com caiyunfeng@bimsa.cn
Correspondence: sunmingming@bimsa.cn

Abstract

Structural understanding of natural language requires explicit recovery of internal meaning structures (entities, facts, nested relations), yet current structural-analytic tasks are fragmented by inconsistent task requirements across datasets. We investigate the problem of robust cross-task structural understanding under heterogeneous requirements across structural-analytic tasks and outline a perspective called Analytic NLP in which tasks can be reformulated into a representation-then-decision paradigm. In this paper, we suggest a solution for the representation layer, called Lingua-Graph, which explicitly captures entities, facts, and relations. By representing predictions as explicit graphs with labeled nodes and edges, Lingua-Graph also improves interpretability, enabling transparent inspection and error analysis of intermediate meaning structures. We construct a labeled Lingua-Graph dataset and train a baseline parser. Experiments show that Lingua-Graph provides substantially higher entity-structure hostability than alternative representations on average, and OpenIE systems based on Lingua-Graph achieve superior performance on three benchmarks, demonstrating that better intermediate structures translate into downstream gains. The data, code and the trained model are publicly released at https://github.com/rudaoshi/Lingua.

1 Introduction

Recovering the internal structure of natural language—entities, facts, arguments, and their compositional relations—has been a long-standing goal of natural language understanding. Beyond classical information extraction pipelines, the recent LLM era has renewed this need: memory systems such as GraphRAG (Edge et al., 2025) and AriGraph (Anokhin et al., 2024) explicitly benefit from *structural* memories (Zeng et al., 2024), which in turn require general-purpose structural language understanding to reliably extract entities and facts from text.

Historically, structural understanding is pursued through a collection of isolated structural-analytic tasks—entity detection/typing, coreference resolution, open information extraction, semantic role labeling, relation extraction, and others. In practice, each application scenario instantiates these tasks with its own label set and dataset, and trains a task-specific model (e.g., a BERT-based classifier (Devlin et al., 2019)). This paradigm is effective within a fixed setting, but the acquired structures are typically tied to scenario-specific annotation criteria and transfer poorly across tasks or domains.

Large language models (LLMs), such as GPT (OpenAI, 2026) and DeepSeek (DeepSeek, 2026), attempt to unify extraction via prompting. However, they still struggle to reconcile heterogeneous (and sometimes conflicting) annotation conventions across datasets and tasks (Kim et al., 2024; Xie et al., 2023; Wang et al., 2023). Consequently, practical systems often resort to dataset-specific instruction tuning (Zhou et al., 2024), where prompts explicitly encode task or dataset names, and extraction quality can be sensitive to prompt variants (Zhuo et al., 2024) and chunking/context choices (Zhong et al., 2025). Robust cross-task structural understanding therefore remains an open problem.

We argue that a more principled route is to recover a *shared* global structure latent in language, such that different tasks correspond to selecting and labeling different *local substructures* of the same underlying representation. This motivates a perspective we call **Analytic Natural Language Processing**: build a common intermediate structural layer, then implement downstream tasks as decisions over that layer. Besides improving reuse across tasks, an explicit intermediate layer also enhances *interpretability*: the predicted structure can be inspected directly to reveal which entities, facts, and relations the system has recov-

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 24630–24646

ered, and downstream decisions can be traced back to specific substructures. Many classic parsing formalisms (e.g., constituency/dependency parsing) and semantic representations (e.g., AMR (Banarescu et al., 2013) and Open-Information Annotation (Sun et al., 2020)) follow this representation–decision spirit, but they are rarely designed or validated explicitly as a *cross-task* substrate where multiple tasks are consistently modeled as substructure identification in a shared global structure.

In this work, we study the construction of the representation layer for Analytic Natural Language Processing and propose a solution called **Lingua-Graph**. Lingua-Graph represents linguistic elements—including entities, facts, and relations—as structured subgraphs in a hierarchical predicate–function–argument graph, enabling diverse structural-analytic tasks to be reformulated as learning problems over substructures. To evaluate this idea, we construct a labeled dataset for Lingua-Graph and develop a baseline parsing model. Experiments on representation expressivity across a diverse set of datasets and downstream open information extraction show that Lingua-Graph captures cross-task common internal structures and supports the Analytic NLP paradigm.

2 Analytic NLP

2.1 Perspective

Analytic NLP factorizes a family of structural-analytic tasks into two components: a *representation layer* that constructs a shared structured object, and a *decision layer* that produces task-specific outputs by making decisions over that structure:

$$x \xrightarrow{f_\theta} G \xrightarrow{\mathcal{D}_t} \hat{y}_t. \tag{1}$$

• **Representation layer $\mathcal{R}$.** Given a sentence $x_{1:n}$, the representation layer specifies (i) a parser/constructor $f_\theta : x \to G$ that maps text to a structured representation G (graph, tree, hypergraph, database instance, etc.), and (ii) a *candidate-unit generator* $\Gamma_{\mathcal{R}}(G)$ that enumerates admissible substructures regarded as candidate analytic units:

$$f_\theta : x \to G, \qquad \Gamma_{\mathcal{R}}(G) \subseteq 2^{V(G)}, \tag{2}$$

where $V(G)$ denotes the set of structural elements in G. Intuitively, $\Gamma_{\mathcal{R}}(G)$ defines the *expressive search space* exposed to downstream tasks: the decision layer can only select from what the representation makes available. Note that the representation is irrelevant to the specific task.

• **Decision layer $\mathcal{D}_t$** For each task t, the decision layer maps the representation-induced candidates to final outputs:

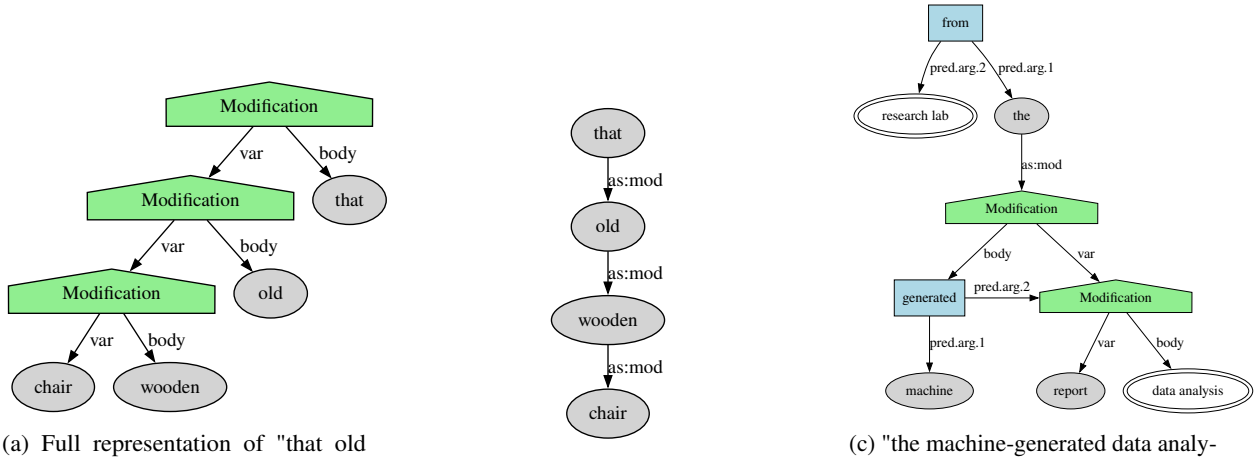
$$\hat{y}_t = \mathcal{D}_t\big(\Gamma_{\mathcal{R}}(G),\, x\big), \tag{3}$$

$\mathcal{D}_t$ is intentionally broad: it may be purely data-driven, purely logic/optimization-based, or based on probabilistic-relational inference that combines learned scores with logical constraints.

Beyond historical roots. The intuition of Analytic NLP—*construct an explicit intermediate representation and make task decisions over it*—has long existed in NLP (see Appendix A). The significance of Analytic NLP is not the decomposition itself, but making the *representation layer* (i) *shared* across tasks and (ii) *explicitly evaluable*. First, the intermediate structure serves as an interpretable task-agnostic interface that induces a common candidate space of analyzable units, enabling modular decision layers to operate on the same substrate and enabling controlled comparisons of representation quality. Second, Analytic NLP advocates a representation-first protocol: before attributing successes or failures to a decision model, we ask whether the representation can *host* the task-induced units (e.g., via oracle expressivity/hostability). Finally, this perspective targets *explicit* generality beyond the token interface: rather than relying on implicit structure inside end-to-end models, it exposes recurrent analytic units as reusable intermediate objects, while remaining compatible with end-to-end learning.

Roadmap and scope. In this paper, we focus on the representation component and instantiate it with Lingua-Graph. Specifically, Section 3 introduces Lingua-Graph and formalizes its candidate substructure space, and we evaluate its representational capacity via *expressivity*: whether task-relevant units can be realized as substructures of the parsed graph.

Developing a *task-independent* decision layer $\mathcal{D}$ is an important long-term direction. Because decision rules are closely tied to task conventions and dataset-specific annotation guidelines, we treat this component as complementary to the representational question studied here. Accordingly, throughout the paper we use task-specific decision modules $\mathcal{D}_t$ when needed (e.g., the OpenIE decision module in our downstream evaluation) to assess how improvements in the representation layer translate into downstream gains.



(a) Full representation of "that old wooden chair"

(b) Simplified representation of (a)

(c) "the machine-generated data analysis report from research lab"

Figure 1: Lingua-Graph for Modifications. Some subgraphs are aggregated as single node with double borderlines.

2.2 Representation Layer

Let an input sentence be a token sequence $x_{1:n}$ and let $I = \{1, \ldots, n\}$ be the token-index set. We use *text-anchored units* to describe the atomic objects that structural-analytic tasks care about.

Task-Induced Unit Set For a task t, let $\mathcal{U}_t(x)$ denote the set of task-induced units on x. In the core case where task units are anchored to text spans/segments, we model them as token sets:

$$\mathcal{U}_t(x) \subseteq 2^I. \tag{4}$$

For example, in NER, $\mathcal{U}_{\text{NER}}(x)$ is the set of gold entity/mention token-sets. More generally, $\mathcal{U}_t(x)$ may represent predicate spans, argument spans, event triggers, or other anchored fragments.

Representation-Induced Candidate Unit Set An Analytic NLP methodology posits an explicit *representation layer* $\mathcal{R}$, consisting of a parser f_θ and an *admissible substructure constructor* $\Gamma_\mathcal{R}$. To connect substructures to text, $\mathcal{R}$ provides a text realization map $\text{tok}_\mathcal{R} : 2^{V(G)} \to 2^I$, which converts a substructure into the token indices it covers. The representation-induced candidate unit set in the text domain is defined as

$$\mathcal{U}_\mathcal{R}(x) := \{ \text{tok}_\mathcal{R}(H) \mid H \in \Gamma_\mathcal{R}(G) \}. \tag{5}$$

Expressivity / Oracle Recoverability We quantify the expressivity of the representation layer for task t by measuring how well $\mathcal{U}_\mathcal{R}(x)$ can cover $\mathcal{U}_t(x)$, that is,

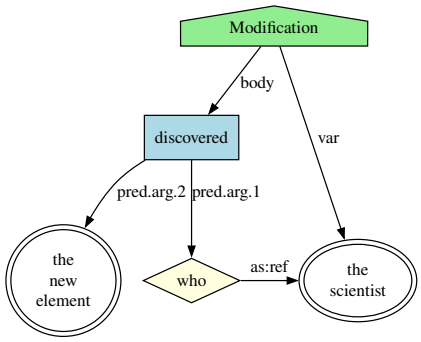
$$\text{Exp}(t; \mathcal{R}, x) := \frac{|\mathcal{U}_{\mathcal{R}(x)} \cap \mathcal{U}_t(x)|}{|\mathcal{U}_t(x)|} \tag{6}$$

For a dataset $\mathcal{D}$, we report $\text{Exp}(\mathcal{D}) = \mathbb{E}_{x \sim \mathcal{D}}[\text{Exp}(t; \mathcal{R}, x)]$. These scores evaluate the representation layer as an *oracle upper bound* (i.e., feasibility of expressing task units), independent of any learned task decision on top of G.

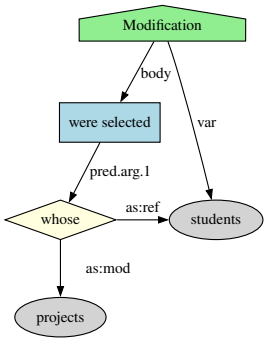
Selectivity / Candidate-Space Compactness While expressivity evaluates whether task-induced units can be realized in the representation, it does not distinguish between a compact candidate space and an overly redundant one. A representation may achieve high expressivity simply by exposing a very large set of candidates, which can make downstream decision making more difficult. We therefore introduce *selectivity* as a complementary metric:

$$\text{Sel}(t; \mathcal{R}, x) := \frac{|\mathcal{U}_{\mathcal{R}(x)} \cap \mathcal{U}_t(x)|}{|\mathcal{U}_R(x)|} \tag{7}$$

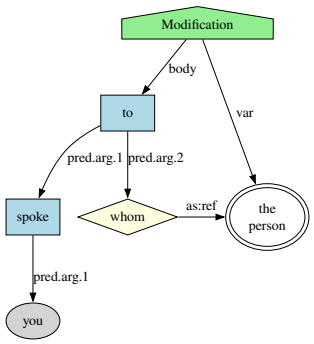
This metric measures the proportion of representation-induced candidate units that correspond to gold task units, and thus captures a precision-like notion of compactness at the representation layer. A higher selectivity score means that the representation exposes fewer irrelevant or redundant candidates relative to the amount of task-relevant structure it makes available. For a dataset D, we report $\text{Sel}(D) = \mathbb{E}_{x \sim D}[\text{Sel}(t; R, x)]$. Together, expressivity and selectivity characterize two complementary aspects of representation quality: coverage of task-induced units and compactness of the induced candidate space. As with expressivity, selectivity is evaluated as an oracle property of the representation layer, independently of any learned decision model built on top of G.



(a) "the scientist who discovered the new element"



(b) "students whose projects were selected"



(c) "the person to whom you spoke"

Figure 2: Lingua-Graph for Relative Clause

3 Lingua-Graph

3.1 Representation Model

Lingua-Graph is a graph-structured implementation of the representation layer of Analytic NLP. Given a sentence $x_{1:n}$ with token-index set $I = \{1, \ldots, n\}$, its Lingua-Graph is a single-rooted DAG

$$G = (V, E, \tau_V, \tau_E, \alpha), \tag{8}$$

where V is a set of nodes, $E \subseteq V \times \Sigma_E \times V$ is a set of labeled directed edges, $\tau_V : V \to \Sigma_V$ and $\tau_E : E \to \Sigma_E$ assign node/edge types from finite inventories Σ_V, Σ_E, and is equipped with a token partition map $\alpha : V \to 2^I$ such that the node token-sets form a partition of I:

$$\cup_{v \in V} \alpha(v) = I, \alpha(u) \cap \alpha(v) = \emptyset, \forall u \neq v. \tag{9}$$

Admissible Substructures: We define two primitive constructors that yield task-candidate substructures in the graph domain: SELF yields a single-node unit and DESCENDANT yields the rooted descendant-closure of a node under the directed graph topology:

$$\textsc{Self}(v) := \{v\}, \textsc{Desc}(v) := \{u \in V \mid v \rightsquigarrow u\}. \tag{10}$$

We define the family of substructures yielded by Lingua-Graph as

$$\Gamma_{\text{LG}}(G) = \cup_{v \in V} \{\textsc{Self}(v), \textsc{Desc}(v)\}. \tag{11}$$

Each $H \in \Gamma_{\text{LG}}(G)$ is a set of nodes (a substructure in the graph domain) and serves as a candidate unit for downstream analytic tasks.

Candidate Unit Set: The representation-induced candidate unit set in the text domain is defined as

$$\mathcal{U}_{\text{LG}}(x) := \{ \text{tok}_{\text{LG}}(H) \mid H \in \Gamma_{\text{LG}}(G)\}, \tag{12}$$

where $\text{tok}_{\text{LG}(H)} := \cup_{v \in H} \alpha(v) \subseteq I$.

3.2 Lingua-Graph Implementation

Based on our representation model, Lingua-Graph is a generalized phrase graph (Sun et al., 2020), in which nodes correspond to phrases in a sentence or to symbolic elements. We build Lingua-Graph on the Predicate–Function–Argument backbone introduced in OIA-Graph (Sun et al., 2020), while extending it with the following key improvements over OIA-Graph:

Hierarchical architecture. Lingua-Graph adopts a hierarchical design so that each node, together with its descendant subgraph, corresponds to a coherent internal structure in the sentence. This better aligns the representation with the compositional and hierarchical nature of language.

New structures. We introduce additional structures to model relationships that are difficult to capture with the original schema. For example, *Expression* represents non-modificational composition relations, and *Attribute* captures $\langle entity, attribute\_name, value \rangle$ tuples.

Finer-grained units. To better capture internal structure within phrases, Lingua-Graph decomposes phrases into finer-grained units and explicitly represents relations among them, especially within noun phrases (which are often treated as single nodes in OIA-Graph).

Figures 1 and 2 illustrate Lingua-Graph representations of basic linguistic structures, which differ substantially from those of OIA-Graph (see Figure 3 in (Sun et al., 2020) for comparison). With these refinements, Lingua-Graph can represent natural substructures in language more neatly, precisely, and compositionally. The full schema and detailed modeling choices of Lingua-Graph are pro-

vided in Appendix B.

3.3 Comparing to Other Semantic Meaning Representations

Deep semantic meaning representations (SMRs) such as AMR (Banarescu et al., 2013) and UCCA (Abend and Rappoport, 2013) instead target abstract or conceptual semantics, and Lingua-Graph can be categorized as a deep SMR as well (Saddedine et al., 2024). The key difference is that Lingua-Graph is designed as a *representation interface* whose primary goal is to expose reusable task-induced substructures.

• **Interface goal vs. holistic semantics.** Dependency/constituency parsing mainly captures surface syntactic relations; AMR maps text to a normalized symbolic meaning graph (Figure 3a); UCCA organizes meaning around cognitively motivated scenes (Figure 3b). In contrast, Lingua-Graph targets a shared interface of analyzable units (entities, facts, nested relations) that can be reused across structural–analytic tasks.

• **Subgraph locality for practical sub-facts.** Many downstream decisions operate on partial structures (extraction, alignment, evidence selection), requiring local entities/facts to correspond to clean subgraphs. This is not a primary optimization target of AMR/UCCA/OIA. For example, the subfact *"we saw the film"* (excluding an adjunct such as "yesterday") is not naturally isolated as a simple descendant subgraph in AMR, UCCA and OIA (Figures 3a, 3b and 3c), whereas it forms a simple descendant subgraph in Lingua-Graph (Figure 3d).

• **Granularity and observability.** Compared to AMR, Lingua-Graph stays closer to surface-observable structure by explicitly modeling the syntax–semantics interface, which facilitates span anchoring, compositional reuse of substructures, and error diagnosis.

• **Hierarchical composition beyond OIA.** OIA (Sun et al., 2020) is the most directly related line to Lingua-Graph, but its graph is closer to dependency-style surface analysis and is relatively flat, with limited hierarchical composition. Lingua-Graph extends this design by explicitly representing nested facts and their composition, making task-relevant substructures more precise and reusable.

4 Task Reformulation with Lingua-Graph

Given a Lingua-Graph parsed from a sentence, the entities and facts targeted by many analytic NLP tasks appear as localized subgraphs. Therefore, a broad range of tasks can be reframed as identifying/classifying (pairs of) subgraphs within a shared graph container (Figure 4). In this paper we emphasize the Lingua-Graph design and its capacity to *host* task-relevant substructures, and leave subgraph (pair) classification models to future work.

4.1 Entity Detection

Entities may be represented either by a single node or by a node's descendant subgraph, which naturally supports nested entities. We annotate each node with two booleans: `SelfIsEntity` (the node itself is an entity) and `DescendantIsEntity` (its descendant subgraph forms an entity). Domain-specific detection becomes node-attribute classification on the graph.

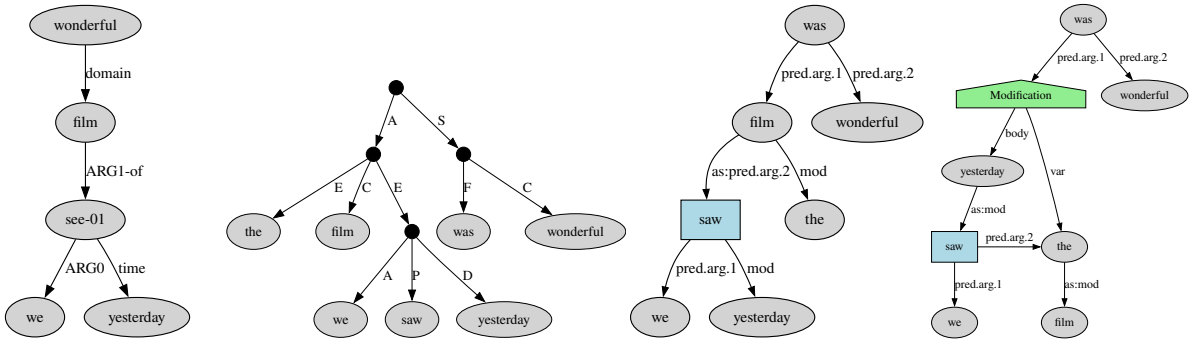
Example: in "2024 Paris Olympic Games", nodes "2024", "Paris", and "Olympic Games" are standalone entities, so `SelfIsEntity=True`; "2024" and "Paris" also modify a larger entity, so `DescendantIsEntity=True`. In "an advanced nutrition program for elite athletes", "for" is not an entity (`SelfIsEntity=False`) but its descendant subgraph realizes a full nominal phrase, hence `DescendantIsEntity=True`. The same pattern applies to structural nodes like "in", `Apposition`, and `Modification`.

4.2 Entity Typing

Typing can be posed as node classification (domain-specific) or tagging/generation (open-domain). Mirroring detection, each entity has `SelfType` (type expressed by the node) and `DescendantType` (type expressed by its descendant subgraph). In Figure 4, "2024" has `SelfType=Year` and `DescendantType=SportsEvent`; "Paris" has `SelfType=Location` and the same `DescendantType`. Detection and typing can be jointly modeled.

4.3 Semantic Role Labeling

Given a `FactualPredicator` node, SRL can be formulated as the task of identifying its argument nodes and assigning them semantic roles (e.g., `Agent`, `Theme`, and `Instrument`). Concretely, this can be implemented as an edge-prediction decision layer over Lingua-Graph, in which semantic roles are encoded as labeled edges from the predicate node to its argument nodes.



(a) AMR (b) UCCA (c) OIA (d) Lingua

Figure 3: Semantic Meaning Representations for sentence "the film we saw yesterday is wonderful"

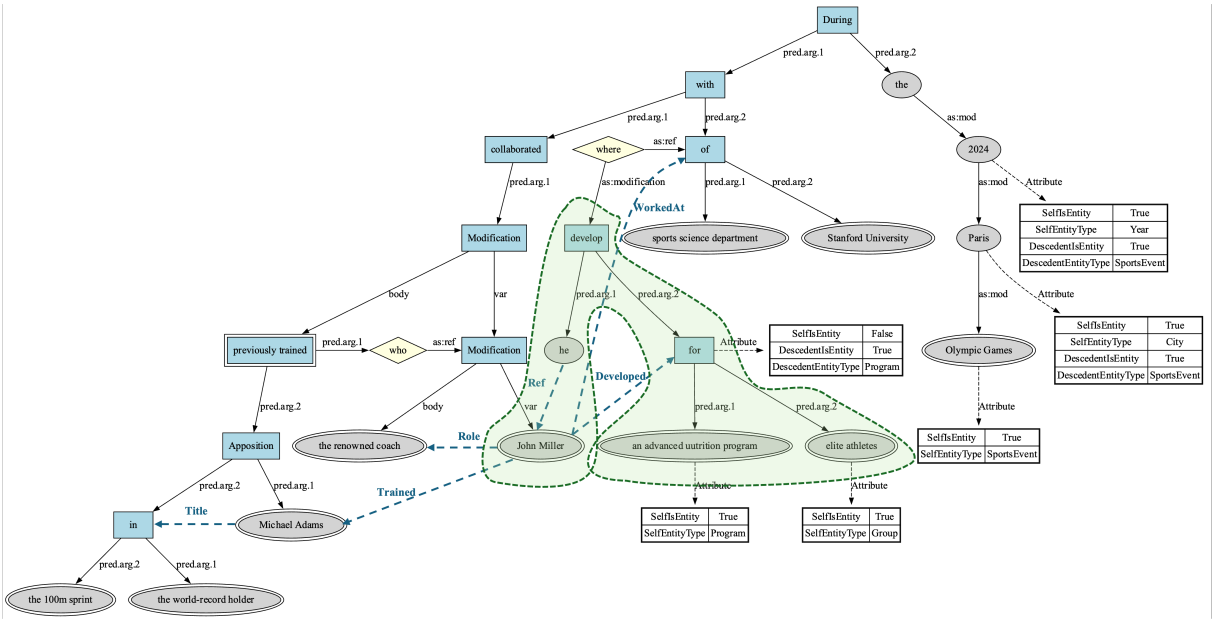


Figure 4: Lingua-Graph as the container of NLP tasks.

4.4 Relation Extraction and Coreference Resolution

Both tasks can be implemented as link prediction over Lingua-Graph. For relation extraction, edges between entity nodes encode relational semantics (e.g., links from “John Miller” / “Michael Adams” to associated entities). For coreference, a dedicated `Ref` edge connects an anaphor (e.g., “he”) to its antecedent (e.g., “John Miller”), explicitly representing referential identity.

4.5 Open Information Extraction

Open facts can be represented by identifying clause-level predicates (nodes marked as `FactualPredicator`); their argument nodes form a complete fact. In Figure 4, the highlighted region yields <*“he”/“John Miller”, “developed”, “an advanced nutrition program for elite athletes”*>.

5 Learning the Lingua-Graph Parser

5.1 Dataset

We construct the Lingua-Graph dataset from the following two resources:

• **OIA-to-Lingua.** This pipeline takes the labeled OIA-Graph dataset[1], introduces Lingua-Graph-specific structures, decomposes complex noun-phrase nodes, enforces a hierarchical organization, and converts the resulting graphs into the Lingua-Graph format.

• **LLM parsing of NLP task samples.** To increase dataset diversity, we additionally sample sentences from the training sets of several structural/-analytic NLP datasets, including ACE05 (Walker et al., 2006), CoNLL03 (Tjong Sang and De Meulder, 2003), GENIA (Kim et al.,

[1] https://github.com/baidu-research/oix

Dataset	LLM	Dependency	Constituency	AMR	Lingua
ACE05	30.98	79.27	84.19	55.42	**92.99**
CoNLL03	86.19	85.56	83.29	92.39	**95.74**
GENIA	73.55	77.49	64.49	40.19	**89.65**
MultiNERD	90.87	88.93	79.21	**95.07**	94.51
OntoNotes	58.74	78.00	**92.39**	77.92	92.32
PolyglotNER	74.82	88.24	81.19	90.40	**91.62**
WikiANN	78.57	73.31	82.66	76.48	**92.90**
Cross NER	**87.02**	78.13	72.10	81.13	82.59
MIT-Restaurant	43.83	44.70	**82.13**	52.30	77.90
MIT-Movie	50.97	67.17	82.15	71.63	**87.97**
Avg	67.55	76.08	80.41	73.29	**89.82**

Table 1: Structure expressivity on various NLP datasets.

2003), MultiNERD (Tedeschi and Navigli, 2022), OntoNotes (Weischedel et al., 2010), PolyglotNER (Al-Rfou et al., 2015), and WikiANN (Pan et al., 2017). We parse these sentences with a large language model (LLM) to extract semantic relations, and then convert the extracted relations into Lingua-Graph using a rule-based conversion system.

We then apply an active-learning procedure with human annotation to improve dataset quality. Specifically, we identify outliers and low-confidence samples (measured via out-of-fold evaluation), select representative instances for manual review and annotation by a trained linguistics student, and discard the rest. Approximately 20% of the samples were manually audited and annotated.

Using this process, we built a dataset with approximately 11K training samples, 1K development samples, and 1K test samples. We assess dataset quality by asking two independent evaluators to annotate a randomly selected set of 100 samples. Under a conservative estimate that counts half of disputed cases as correct, the estimated accuracy is 84%. Inter-annotator agreement, measured by Cohen's kappa, is approximately 0.63, indicating substantial agreement. The main sources of disagreement are: 1) Linguistic ambiguity, especially ambiguity in modification scope / hierarchical attachment; 2) Guideline interpretation differences for certain schema elements (e.g., some Expression-related cases).

5.2 Learning the Parser

Lingua-Graph is a challenging learning target: it includes both multi-word phrase nodes and additional symbol nodes, making direct end-to-end parsing nontrivial. We therefore adopt the learning paradigm used for OIA-Graph in (Wang et al., 2022) to train the parser. Specifically, we first design a *word-level* Lingua-Graph representation that can be converted to and from the original Lingua-Graph losslessly, where each node corresponds to a single word in the input sentence. We then convert gold Lingua-Graphs into word-level graphs and train a BERT-based biaffine relation parser (Dozat and Manning, 2017) to predict the relations between words in this word-level representation.

5.3 Inference

We decode the predicted word-level Lingua-Graph using a greedy search strategy, following the decoding procedure used for OIA-Graph (Wang et al., 2022). During decoding, edges with the highest scores are incrementally added to the graph, while both schema constraints and structural constraints are enforced to ensure that the resulting word-level graph is valid. The decoded word-level Lingua-Graph is then converted back to the original Lingua-Graph. Upon publication, we will release the trained model and the inference code.

6 Experimental Results

6.1 Learning the Lingua-Graph

We fine-tune BERT-base as the backbone model under supervised learning on the Lingua-Graph dataset described in Section 5.1. We set the embedding dimensions of the arc-bilinear and label-bilinear components to 512 and 256, respectively. The baseline parser is trained for 50 epochs with a learning rate of 1×10^{-5}, selecting the checkpoint with the best validation performance.

We evaluate the trained parser on the test set

Dataset	Dependency	Constituency	AMR	Lingua
ACE05	11.63	9.89	9.96	10.92
CoNLL03	8.24	7.32	11.38	8.76
GENIA	4.84	4.53	4.35	7.20
MultiNERD	3.75	3.36	7.35	4.23
OntoNotes	6.77	5.84	6.29	6.24
PolyglotNER	4.69	4.12	4.38	4.53
WikiANN	10.16	9.56	13.95	10.00
Cross NER	7.35	7.07	14.04	8.33
MIT-Restaurant	10.50	10.73	14.32	11.10
MIT-Movie	9.76	10.30	12.00	11.53
Avg	7.77	7.27	9.80	8.28

Table 2: Structure selectivity on various NLP datasets.

at two levels: (i) the predicted word-level Lingua-Graph, and (ii) the recovered Lingua-Graph. We report two metrics: 1) *Node matching:* precision/recall/F1 of the predicted node set (including node label and type) under exact match; 2) *Edge matching:* precision/recall/F1 of the predicted edge set, where each edge is represented as a triplet ⟨*start-node-words*, *edge-label*, *end-node-words*⟩. To emphasize structural evaluation, we use approximate matching for node-word sets: two sets are considered matched if they share at least one word.

Results are shown in Table 3. Further investigation suggests that most errors can be attributed to annotation inconsistencies in the dataset.

Metric	word-level Lingua	Lingua
Node	96.1/96.1/96.1	93.4/93.2/93.4
Edge	84.8/85.0/84.9	83.0/82.8/82.9

Table 3: Performance of the Lingua-Graph Parser

6.2 Structure Expressivity

We evaluate the expressivity of Lingua-Graph by testing whether it can *host* the task-induced substructures required by diverse datasets. As discussed in Section 4, many analytic NLP tasks (e.g., NER, relation extraction, SRL, coreference, and OpenIE) are structurally anchored by entities: relations connect entity mentions, arguments are often entity mentions, and references resolve to entity mentions. Therefore, the ability to represent complex (including nested) entity spans as neat and concise substructures provides a natural proxy for the expressivity of Lingua-Graph as a task-agnostic intermediate layer. We accordingly select datasets with rich entity annotations spanning a broad range of task objectives and conventions (including multi-task datasets such as OntoNotes).

Our evaluation focuses on *Expressivity*: whether a gold entity span corresponds to either (i) a node, or (ii) a descendant subgraph of a node in a given representation. This evaluation isolates representational capacity from the separate decision problem of identifying entities among candidate substructures, which we leave for future work.

We compare Lingua-Graph with major parsing and meaning-representation formalisms, including constituency parsing, dependency parsing, and AMR, as baselines.[2] For each representation and dataset, we report expressivity $\mathrm{Exp}(\mathcal{D})$, defined as the proportion of gold entity spans that can be realized by some node or a node's descendant subgraph. As an additional reference point, we also report the recall of a strong LLM-based entity extractor (Appendix C).

Results are shown in Table 1. CrossNER (Liu et al., 2021) and two MIT datasets (MIT-Restaurant (CSAIL, MIT, 2006b) and MIT-Movie (CSAIL, MIT, 2006a)) serve as out-of-domain evaluations, as their data are not included in our Lingua-Graph training set. Overall, Lingua-Graph achieves substantially higher entity-span hostability than the other representations on average, supporting our goal of capturing reusable substructures across tasks. For the remaining $\sim$10% of spans that are not hostable—largely involving complex phenomena not covered by our current supervision—we expect further gains from broader training data and stronger learning/decoding algorithms, which we leave for future work.

[2]We do not include UCCA because its pipeline cannot be executed due to outdated dependencies.

Model	OIE2016		Re-OIE2016		BenchIE	
	AUC	F1	AUC	F1	AUC	F1
ClauseIE (Corro and Gemulla, 2013)	36.4	58.1	46.4	64.2	13.9	33.9
OpenIE (Christensen et al., 2011)	40.8	58.8	50.9	68.3	10.7	25.4
BIO (Zhan and Zhao, 2020)	46.2	68.6	71.9	80.3	–	–
SpanOIE (Zhan and Zhao, 2020)	48.9	68.7	65.8	77.0	–	–
BiLSTM + BERT (Ro et al., 2020)	–	–	72.1	81.3	–	–
Multi2OIE (Ro et al., 2020)	–	–	74.6	83.9	8.1	22.8
OIE@OIA (Wang et al., 2022)	54.3	71.6	76.9	85.3	–	–
OIE@Lingua	**67.2**	**78.3**	**82.7**	**87.4**	**15.6**	**35.5**

Table 4: Performance Comparison on Three OIE Benchmarks

6.3 Structure Selectivity

Expressivity is the primary metric in our main comparison because we consider a relatively homogeneous family of structured sentence representations, including dependency parsing, constituency parsing, AMR, and Lingua-Graph. For these representations, candidate units are induced from explicit structures via the same local constructors (e.g., a node itself or a node's descendant subgraph), so the induced candidate space preserves structural locality and linguistic meaning rather than trivially enumerating arbitrary token subsets to obtain artificial coverage. Moreover, for this family of representations, the size of the induced candidate space is of the same order—roughly linear in sentence length—making expressivity the more informative first-order criterion for comparing representational hostability.

Nevertheless, for completeness, we report *selectivity*, which measures the proportion of representation-induced candidate units that correspond to gold task units. In this evaluation, we use the same datasets and the same candidate construction as in Section 6.2. For each structured representation, we compute dataset-level selectivity $\mathrm{Sel}(\mathcal{D})$ with respect to the gold entity spans. We restrict this comparison to representations with explicit intermediate candidate spaces; accordingly, the LLM baseline in Table 1 is not included here, since it directly outputs final entity predictions rather than exposing a candidate set induced from a structured representation. Results are shown in Table 2. Overall, Lingua-Graph maintains a selectivity level comparable to other structured representations while achieving substantially higher expressivity, suggesting that its gains are not obtained simply by inflating the candidate space, but by organizing task-relevant structures more effectively.

6.4 Downstream Task Evaluation

Since this paper primarily studies the representation layer, we choose a downstream task where a clear representation–decision decomposition is already established, so that we can isolate and verify the effect of representation quality on end-task performance. We adopt open information extraction (OIE), for which OIE@OIA (Wang et al., 2022) provides a mature paradigm: OIA-Graph serves as the representation layer, and a rule-based system implements the decision layer. We replace OIA-Graph with Lingua-Graph and adapt the decision rules accordingly, yielding our OIE@Lingua system. As shown in Table 4, OIE@Lingua achieves better performance, demonstrating that Lingua-Graph more effectively captures the clause-level structure of open facts and that improvements in the intermediate representation translate into downstream gains.

7 Conclusion

In this work, we introduced the concept of Analytic NLP and proposed Lingua-Graph as a concrete implementation of its unified intermediate representation layer. We focused on modeling the common inner structures of language to provide a consistent structural foundation for a wide range of analytic NLP tasks. The reasonableness and effectiveness of this approach are supported by our experiments: Lingua-Graph attains higher expressivity of task-induced entity structures than alternative representations on average, and replacing the intermediate structure with Lingua-Graph yields consistent gains in our OpenIE case study. Future work includes improving parsing robustness, scaling annotation, extending coverage to more linguistic phenomena and task settings, and investigating the construction of task-independent/generalized decision layers.

Limitations

• **Scaling to broader settings.** The framework depends on accurate graph parsing; complex linguistic phenomena can still be difficult to parse. Extending the parser and the annotation pipeline to broader conditions (e.g., noisy, informal, or spoken text) is an important direction to further strengthen robustness in real-world applications.

• **Task conventions and decision layer.** Our current study focuses on learning a shared representation layer and evaluating its representational capacity (hostability). Fully leveraging the representation layer across tasks still requires a principled decision layer that copes with task-specific conventions.

References

Omri Abend and Ari Rappoport. 2013. Universal Conceptual Cognitive Annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–238, Sofia, Bulgaria. Association for Computational Linguistics.

Rami Al-Rfou, Vivek Kulkarni, Bryan Perozzi, and Steven Skiena. 2015. POLYGLOT-NER: Massive multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM)*, pages 586–594.

Petr Anokhin, Nikita Semenov, Artyom Sorokin, Dmitry Evseev, Mikhail Burtsev, and Evgeny Burnaev. 2024. Arigraph: Learning knowledge graph world models with episodic memory for llm agents. *Preprint*, arXiv:2407.04363.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2670–2676.

Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. 2023. Binding language models in symbolic languages. In *The Eleventh International Conference on Learning Representations*.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton, The Hague/Paris.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2011. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the 6th International Conference on Knowledge Capture (K-CAP)*, pages 113–120, Banff, Alberta, Canada.

Luciano Del Corro and Rainer Gemulla. 2013. ClausIE: clause-based open information extraction. In *Proceedings of the 22nd International World Wide Web Conference (WWW)*, pages 355–366, Rio de Janeiro, Brazil.

CSAIL, MIT. 2006a. Mit movie corpus. https://groups.csail.mit.edu/sls/downloads/movie/. Semantically tagged training and test corpus in BIO format.

CSAIL, MIT. 2006b. Mit restaurant corpus. https://groups.csail.mit.edu/sls/downloads/restaurant/. Semantically tagged training and test corpus in BIO format.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

DeepSeek. 2026. DeepSeek Chat. https://chat.deepseek.com/. DeepSeek.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2025. From local to global: A graph rag approach to query-focused summarization. *Preprint*, arXiv:2404.16130.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, volume 42 of *Studies in Linguistics and Philosophy*. Springer Dordrecht, Dordrecht.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2017. Answering complex questions using open information extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 311–316,

Vancouver, Canada. Association for Computational Linguistics.

Hongjin Kim, Jai-Eun Kim, and Harksoo Kim. 2024. Exploring Nested Named Entity Recognition with Large Language Models: Methods, Challenges, and Insights. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8653–8670, Miami, Florida, USA. Association for Computational Linguistics.

J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. 2003. GENIA corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics (Oxford, England)*, 19(suppl_1):i180–i182.

Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2021. CrossNER: Evaluating cross-domain named entity recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13452–13460.

OpenAI. 2026. ChatGPT. `https://chat.openai.com/`. OpenAI.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Matthew Richardson and Pedro M. Domingos. 2006. Markov logic networks. *Machine Learning*, 62(1–2):107–136.

Youngbin Ro, Yukyung Lee, and Pilsung Kang. 2020. Multi^{2}OIE: Multilingual open information extraction based on multi-head attention with BERT. In *Findings of the Association for Computational Linguistics (EMNLP Findings)*, pages 1107–1117, Online Event.

Zacchary Sadeddine, Juri Opitz, and Fabian Suchanek. 2024. A survey of meaning representations – from theory to practical utility. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2877–2892, Mexico City, Mexico. Association for Computational Linguistics.

Mingming Sun, Wenyue Hua, Zoey Liu, Xin Wang, Kangjie Zheng, and Ping Li. 2020. A predicate-function-argument annotation of natural language for open-domain information expression. In *Conference on Empirical Methods in Natural Language Processing*.

Simone Tedeschi and Roberto Navigli. 2022. MultiNERD: A multilingual, multi-genre and fine-grained dataset for named entity recognition (and disambiguation). In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 801–812, Seattle, United States. Association for Computational Linguistics.

Erik F. Tjong Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. Technical Report 57:45, Linguistic Data Consortium, Philadelphia, PA.

Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuansen Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, Jihua Kang, Jingsheng Yang, Siyuan Li, and Chunsai Du. 2023. Instructuie: Multi-task instruction tuning for unified information extraction. *Preprint*, arXiv:2304.08085.

Xin Wang, Minlong Peng, Mingming Sun, and Ping Li. 2022. OIE@OIA: an adaptable and efficient open information extraction framework. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6213–6226, Dublin, Ireland. Association for Computational Linguistics.

Ralph Weischedel, Mitch Marcus, Martha Palmer, Eduard Hovy, Robert Belvin, Sameer Pradhan, and Lance Ramshaw. 2010. Ontonotes: A large training corpus for enhanced processing. In *Handbook of Natural Language Processing and Machine Translation*. Springer, New York, NY.

Terry Winograd. 1970. Procedures as a representation for data in a computer program for understanding natural language. AI Technical Report 235, MIT Artificial Intelligence Laboratory.

Tingyu Xie, Qi Li, Jian Zhang, Yan Zhang, Zuozhu Liu, and Hongwei Wang. 2023. Empirical Study of Zero-Shot NER with ChatGPT. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7935–7956, Singapore. Association for Computational Linguistics.

Alexander Yates, Michele Banko, Matthew Broadhead, Michael J Cafarella, Oren Etzioni, and Stephen Soderland. 2007. Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 25–26.

Dongran Yu, Bo Yang, Dayou Liu, Hui Wang, and Shirui Pan. 2023. A survey on neural-symbolic learning systems. *Neural Networks*, 166:105–126.

Ruihong Zeng, Jinyuan Fang, Siwei Liu, and Zaiqiao Meng. 2024. On the structural memory of llm agents. *Preprint*, arXiv:2412.15266.

Junlang Zhan and Hai Zhao. 2020. Span model for open information extraction on accurate corpus. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, pages 9523–9530, New York, NY.

Zijie Zhong, Hanwen Liu, Xiaoya Cui, Xiaofan Zhang, and Zengchang Qin. 2025. Mix-of-granularity: Optimize the chunking granularity for retrieval-augmented generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5756–5774, Abu Dhabi, UAE. Association for Computational Linguistics.

Wenxuan Zhou, Sheng Zhang, Yu Gu, Muhao Chen, and Hoifung Poon. 2024. UniversalNER: Targeted Distillation from Large Language Models for Open Named Entity Recognition. *Preprint*, arXiv:2308.03279.

Jingming Zhuo, Songyang Zhang, Xinyu Fang, Haodong Duan, Dahua Lin, and Kai Chen. 2024. ProSA: Assessing and understanding the prompt sensitivity of LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1950–1976, Miami, Florida, USA. Association for Computational Linguistics.

A Historical Roots

NLP traditions that first construct an explicit intermediate representation and then make task decisions over it include:

• *Grounded symbolic NLU.* Early systems such as SHRDLU (Winograd, 1970) built structured world/meaning representations and then performed planning/reasoning for question answering and command execution.
• *Formal semantics:* Semantic interpretation is often described as a two-stage pipeline: construct an explicit representation (syntactic (Chomsky, 1957; de Marneffe et al., 2021), semantic (Banarescu et al., 2013; Abend and Rappoport, 2013; Wang et al., 2022; Sadeddine et al., 2024), or logical (Kamp and Reyle, 1993)) and then perform downstream inference/decisions over it.
• *Neural-symbolic learning:* Recent neural-symbolic approaches revisit the classical "representation + decision" decomposition by coupling neural models with executable symbolic languages. Rather than making predictions in a purely end-to-end manner, these methods construct explicit programs, logical forms, or other structured symbolic objects, and then perform execution or decision making over them. For example, Binder (Cheng et al., 2023) maps task inputs into executable programs in symbolic languages such as SQL or Python, while allowing language-model-based API calls to handle open-ended semantic subproblems. This line of work can be viewed as a modern continuation of explicit intermediate-representation pipelines, where neural components serve as parsers, semantic operators, or module selectors, while symbolic structures retain interpretability, compositionality, and executability (Yu et al., 2023).
• *Open Information Extraction (OpenIE):* OpenIE advocates domain-independent relational tuples extracted at scale (Banko et al., 2007; Yates et al., 2007), enabling downstream optimization/inference on these tuples (e.g., (Khot et al., 2017)).
• *Probabilistic logic:* Frameworks such as MLN (Richardson and Domingos, 2006) define a probabilistic-relational intermediate layer and study decision making (inference/learning) over it across tasks.

B Lingua-Graph

B.1 Schema

The set of auxiliary symbol node labels are listed in Table 5. The set of node types in Lingua-Graph are listed in Table 6. The set of edge types in Lingua-Graph are listed in Table 7. The schema elements marked with * are introduced by Lingua-Graph.

B.2 Modeling Language Structures

B.2.1 Entities

Lingua-Graph places careful emphasis on the internal structure of entities. Among these, modification structures are the most common and are modeled explicitly using a Modification structure. Other forms of internal composition are represented using the more general Expression structure.

B.2.2 Modifications in Entities

Entities are often modified by adjectives, adverbs, prepositional phrases, or even verb/noun phrases, forming more complex expressions. In representations like the OIA-Graph and Dependency Graph, such modifiers are typically connected to the head noun or verb via direct edges. However, these flat structures fail to capture the compositional nature of modification, where each modifier incrementally constrains the meaning of the modified phrase.

Lingua-Graph addresses this limitation by modeling modification hierarchically: each modifier corresponds to a descendant subgraph of the modified node. At the core is the Modification functor, which takes two arguments: *var* (the phrase being modified) and *body* (the modifying phrase). The expressions of modificational structures in entities are illustrated in Figure 1 and 2.

The phrase *"that old wooden chair"* is represented as a nested sequence of Modification structures (Figure 1a).

This approach naturally handles more complex modifiers, including prepositional, verb, and noun phrases. For instance, *"machine-generated data analysis report"* is modeled as a hierarchical subgraph rooted at the node *"the"* (Figure 1c).

Modifications via relative clauses are also supported. The relative clause is first represented independently, with the *wh*-word marked as a special functor and linked to the modified phrase via an *as:ref* edge. The entire construction is then wrapped in a Modification functor. Figure 2 illustrates several examples.

While the hierarchical representation is expressive, it may be simplified when no ambiguity arises. For instance, the structure in Figure 1a can be reduced (Figure 1b) by replacing the nested functor with a direct *as:modification* edge. Similarly, in some prepositional modifications (e.g., Figure 1c), the Modification functor can be omitted and the preposition treated as the root, provided that the edge *pred.arg.1* clearly identifies the head.

However, not all cases permit simplification. In complex phrases like the one rooted at *"the"* in Figure 1c, the full hierarchy must be preserved to avoid ambiguity in head assignment and structural interpretation.

B.2.3 Expression

Lingua-Graph also introduces a distinct structure type—Expression functor—to handle composite entities like *"48 to 49"* or *"Ca 2+"*. In *"48 to 49"*, the node *"to"* functions as the expression functor, with *"48"* and *"49"* as arguments. In *"Ca 2+"*, a special auxiliary symbol `Expression` serves as the functor node, with *"Ca"* and *"2+"* as its arguments.

B.2.4 Simple Fact

Clause-Level Representation. Lingua-Graph represents a simple fact—typically conveyed by a simple clause—using a predicate–argument structure. The predicate phrase of the clause is represented by a predicate node, while its subject, direct object, indirect object, and object complement are represented as argument nodes connected to the predicate. As illustrated in Figures 5a and 5b, the descendant subgraph rooted at the predicate node thus captures the internal structure of the clause.

Attributes. A special class of simple facts involves attributes, commonly expressed by noun phrases such as *Tesla Model S, battery capacity, 100 kWh*. In such cases, the attribute word (e.g., *battery capacity*) is treated as a predicate node of type `AttributePredicator`, with the associated entity and value as its arguments. For cases involving implicit attribute expressions, such as *François Simon (actor)*, an auxiliary node labeled `Attribute` is used as the predicate node. Figure 5c illustrates both types of attribute representations.

Modification of Facts. Simple facts can be modified by adverbs or prepositional phrases. These modifications are modeled using the same Modification structure introduced in the previous section, where the fact serves as the *var* argument and the modifier as the *body*. As before, this structure can be simplified when doing so introduces no ambiguity.

B.2.5 Relations between Entities and Facts

Logical Relations. Connective words such as "therefore," "so," "if," and "because" indicate logical or temporal relationships between facts. These relationships are represented using logical predicates. When a fact (typically expressed in an independent clause) is linked to multiple logical connections from dependent clauses, the hierarchy among these connections should be explicitly modeled, with each layer represented by a corresponding logical predicate.

Conjunction and Disjunction. Conjunctions and disjunctions such as "and" and "or" combine a list of parallel components, allowing for the expression of shared attributes or behaviors. These coordinated structures are captured using the parataxis function.

Lists of Heterogeneous Elements. In some contexts, texts present lists of heterogeneous elements without emphasizing shared properties. These collections are represented using the list function, which reflects enumeration without assuming structural or semantic similarity.

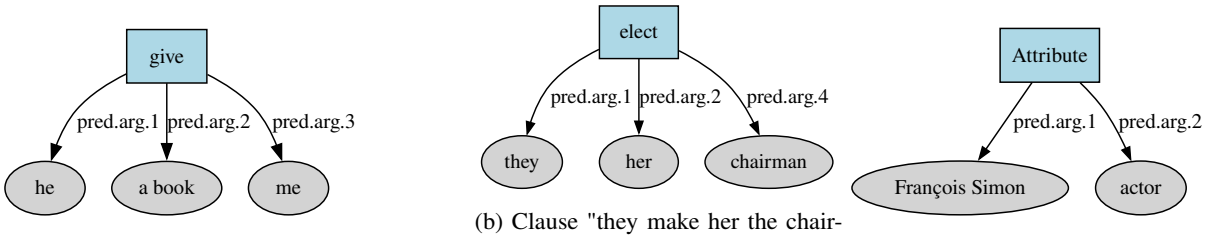
C Entity Detection by LLM

In the LLM-based approach, we use the following prompt template to extract all possible entities from a sentence. The language model employed in this experiment is DeepSeek V3.

Prompt

```
Extract all entities from the following
sentence, and output the entities in
json format {{"entities": [entity1,
entity2, ...]}}.

{sentence}
```



(a) Clause "he gave me a book"

(b) Clause "they make her the chairman"

(c) Phrase "François Simon (actor)"

Figure 5: Lingua-Graph for Simple Facts

Symbol label	**Explanation**
Apposition	Marks an appositive/aliasing relation that re-describes the same entity (e.g., *John, a doctor*).
Attribute*	Marks an attribute–value structure (property assignment) associated with an entity or a nominal/event.
Copula	Marks copular predication (typically *be*-type linking), connecting a subject to a predicate/attribute.
Discourse	Marks discourse-level linkage (e.g., interjections, discourse markers, or parenthetical discourse relations).
Expression*	Marks a composite expression (e.g., ranges, formulas, or multi-token symbolic expressions).
List*	Marks an enumeration/list structure grouping multiple items without enforcing a shared predicate.
Missing	Placeholder symbol when an implicit/elliptic element is required by the structure but not overtly realized.
Modification	Marks modifier–head composition, packaging a modified unit with its modifier phrase/clause.
Parataxis	Marks paratactic coordination/juxtaposition between parallel units (often coordination-like without an explicit head).
Ref	Marks reference linking (e.g., anaphora/relative reference) from a referring form to its target.
Vocative	Marks direct address to an addressee (e.g., *John, ...*), separating it from core propositional content.
Whether	Marks embedded polar/yes-no interrogative subordination (e.g., *whether* clauses) as a structural node.

Table 5: Inventory of Lingua-Graph symbol node labels with brief explanations.

Category	Explanation
AttributePredicator	Predicate node for attribute–value facts (e.g., *battery capacity* with entity/value as arguments).
AppositionalPredicator	Predicate node for apposition/aliasing constructions (e.g., *John Miller, the coach*), linking two co-referring nominals.
FactualPredicator	Predicate node for clause-level facts/events; takes core arguments (subject/object/etc.) in a predicate–argument structure.
LogicalPredicator	Predicate node for logical/temporal relations between facts signaled by connectives (e.g., *if*, *because*, *therefore*).
ReferentialPredicator	Structure for reference phenomena (e.g., anaphora/relative references), linking a referring expression to its antecedent/target.
ConjunctionalFunctor	Functor capturing coordination (*and*/*or*) over parallel components, enabling shared properties/behaviors.
ExpressionFunctor*	Functor for composite expressions (ranges, formulas, etc.), e.g., *48 to 49* or *Ca 2+*.
GeneralFunctor*	Generic functor represented by wh-words in clause.
ListFunctor*	Functor for enumerations of heterogeneous elements (a list without assuming shared structure/semantics).
ModificationalFunctor	Functor for hierarchical modification: *var* (modified phrase) + *body* (modifier phrase).
DeterminerConstant*	Determiner tokens anchoring nominal phrases (e.g., *the*, *a*, *this*).
InterjectionConstant	Interjections/exclamations (e.g., *oh*, *wow*) treated as standalone constants.
ModificationalConstant	Lexical modifiers used inside modification structures (e.g., adjectives/adverbs, pre-nominal modifiers).
NominalConstant	Content nominals (nouns/proper names/numerals) serving as entity-like arguments/heads.
OtherConstant	Fallback constant type for tokens not covered by the above constant categories.
PunctuationalConstant*	Punctuation marks (comma, period, parentheses, etc.) represented as constants.
SymbolConstant*	Symbols/units/special notations (e.g., +, %, scientific symbols) represented as constants.

Table 6: Node category inventory with brief explanations.

Edge label	Explanation
body*	The *body* component in a variable–body (modifier or operator) structure; typically the core content being modified.
variable*	The *variable* (or modified target) component in a variable–body structure; the unit whose meaning is refined by the body.
pred.arg.1	Predicate argument slot 1 (core argument position for a predicator; e.g., subject-like role, depending on predicate type).
pred.arg.2	Predicate argument slot 2 (core argument position; often object-like when applicable).
pred.arg.3	Predicate argument slot 3 (additional argument position for indirect object, etc.).
pred.arg.4	Predicate argument slot 4 (additional argument position for ditransitives, complements, etc.).
func.arg	Functor argument edge: links a functor node to the structured unit(s) it composes (e.g., list/expression/parataxis).
appositive	Apposition edge linking an entity/nominal to its appositive redescription (alias, title, parenthetical name).
attribute*	Attribute edge linking an entity/event to its attribute/value substructure.
discourse	Discourse edge linking discourse markers/interjections/parentheticals to the main clause or discourse unit.
index	Indexing edge for alignment/anchoring (e.g., linking repeated mentions, list indices, or structure bookkeeping).
modification	Modification edge connecting a modifier structure to the unit it modifies (distinct from variable/body decomposition).
punctuation	Punctuation attachment edge anchoring punctuation constants to their local structural host.
ref	Reference edge linking a referring expression (pronoun/relative form/etc.) to its antecedent/target.
repeat	Repeat edge indicating a repeated element/token span used for structural normalization or shared attachment.
vocative	Vocative edge attaching an addressee phrase to the utterance while keeping it outside core propositional content.

Table 7: Inventory of edge labels with brief explanations. The optional "as:" prefix denotes the reverse direction of an edge (i.e., the inverse relation).