Ontologies

Manas Gaur

Sharing Semantics

Metadata

- Data describing the content and meaning of resources and services.
- But everyone must speak the same language…

Terminologies

- Shared and common vocabularies
- For search engines, agents, curators, authors and users
- But everyone must mean the same thing…

Ontologies

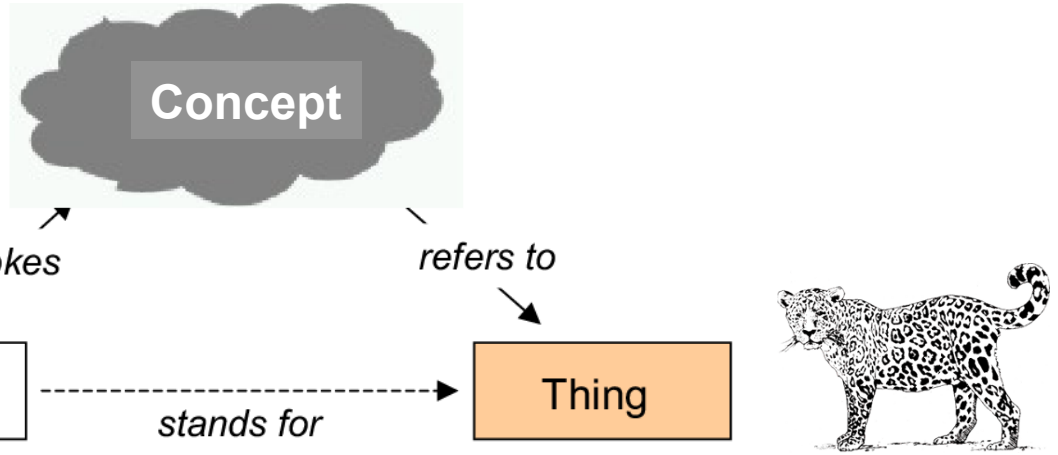
- Shared and common understanding of a domain
- Essential for search, exchange and discovery

➔ Ontologies aim at sharing meaning

[Carole Goble, Nigel Shadbolt, Ontologies and the Grid Tutorial]

The Meaning Triangle

- Humans require words (or at least symbols) to communicate efficiently. The mapping of words to things is indirect. We do it by creating *concepts* that refer to things.
- The relation between symbols and things has been described in the form of the *meaning triangle*:



"Jaguar"

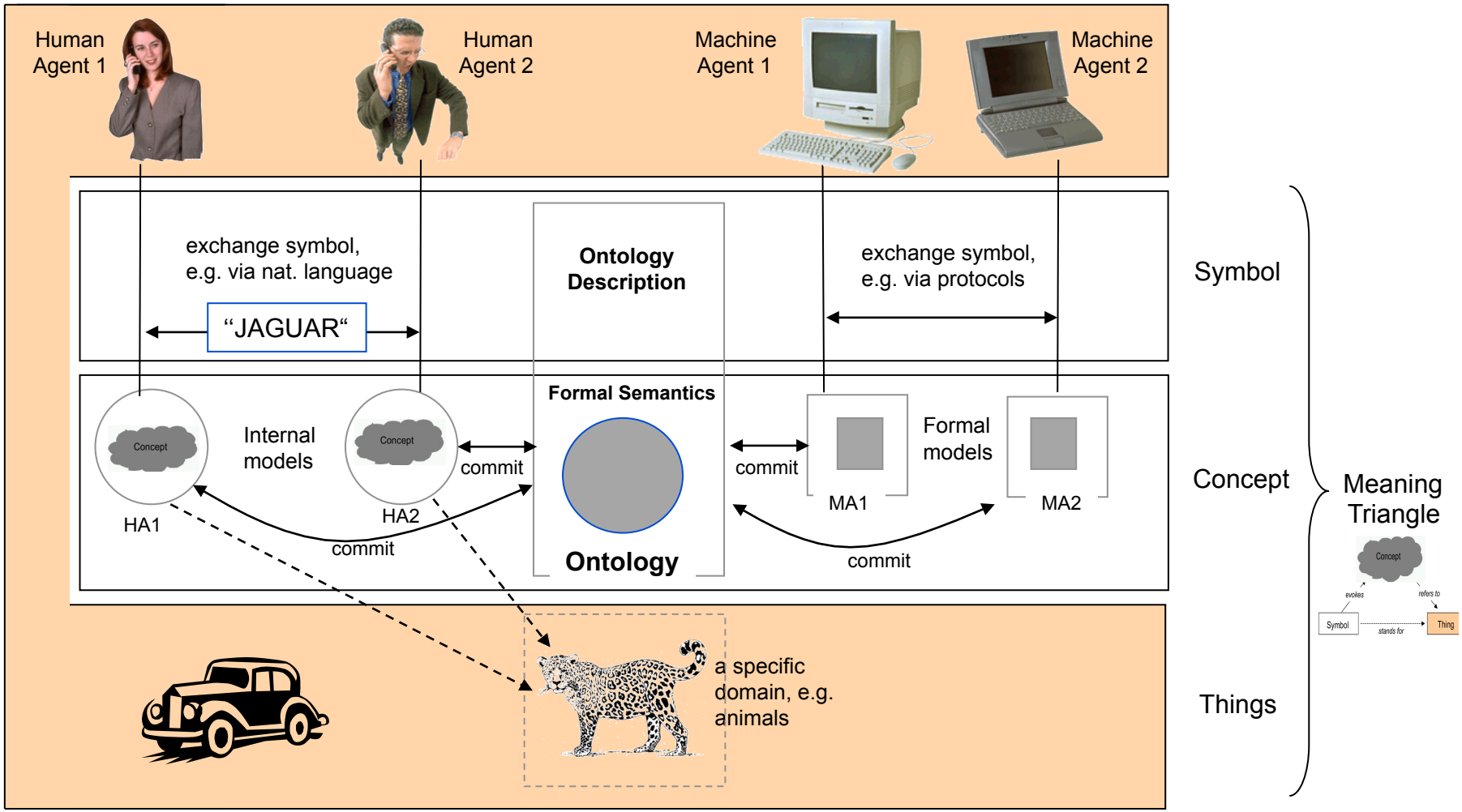


Ogden, C. K. & Richards, I. A. 1923. "The Meaning of Meaning." 8th Ed. New York, Harcourt, Brace & World, Inc

before: Frege, Peirce; see [Sowa 2000]

[Carole Goble, Nigel Shadbolt, Ontologies and the Grid Tutorial]

Human and machine communication



[Maedche et al., 2002]

Ontology: A definition

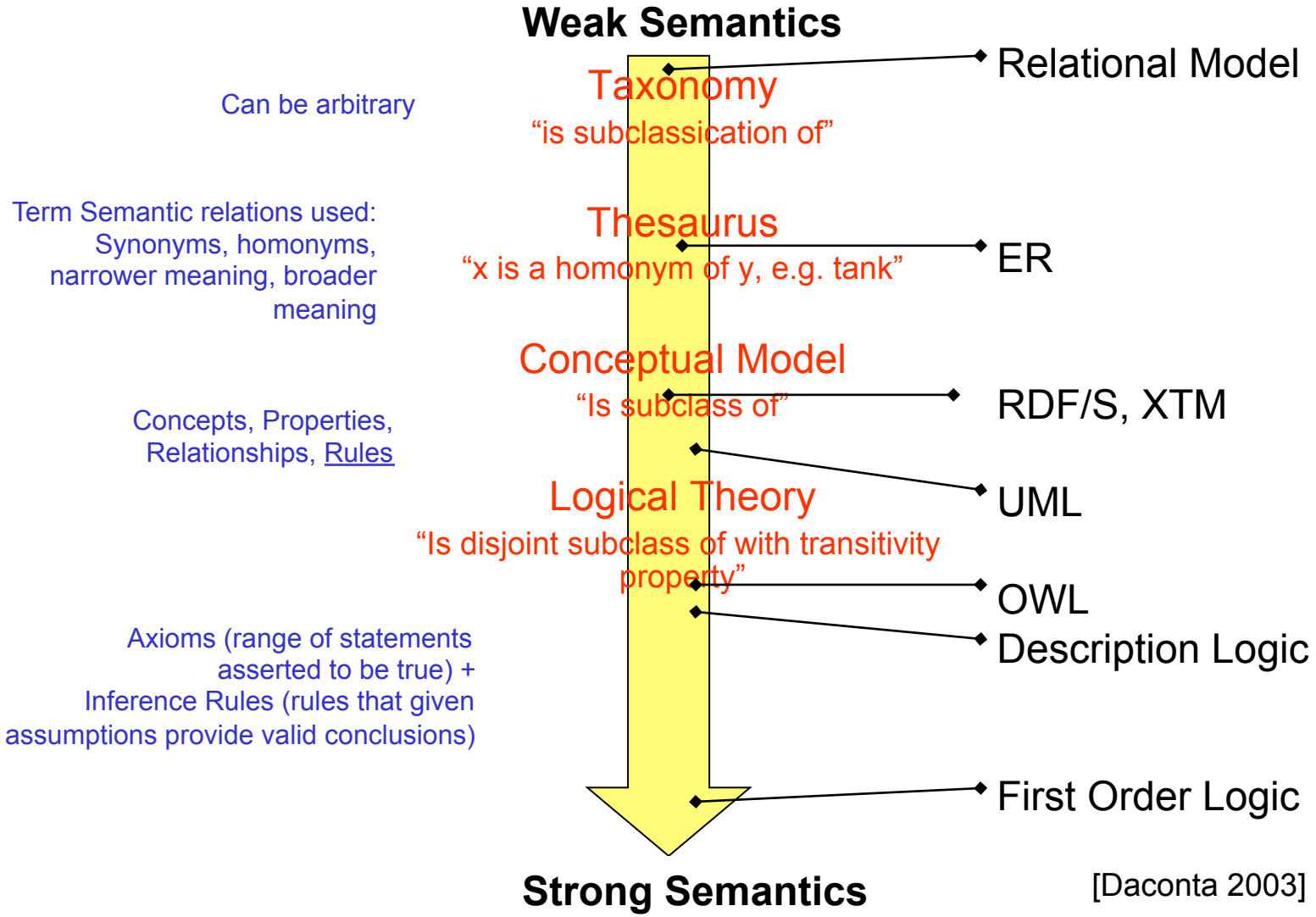
An "ontology" describes the common words, concepts and relationships between concepts used to describe and represent an area of knowledge.

An ontology can range from a

- Taxonomy (knowledge with minimal hierarchy or a parent/child structure) to a …
- Thesaurus (words and synonyms) to a ….
- Conceptual Model (with more complex knowledge) to a…
- Logical Theory (with very rich, complex, consistent and meaningful knowledge).

A well-formed ontology is one that is expressed in a well-defined syntax that has a well-defined machine interpretation consistent with the above ontology definition

Ontology Spectrum



[Daconta 2003]

Difference between Relational Schema and Ontology

Relational Schema primary purpose is to organise data within a database

- As we have seen previously relationships between entities are implicit and interpretation is necessary by program/human who accesses
- If the program/human lacks knowledge of semantics of the data or database then becomes unusable

Ontology

- Relationships defined formally and interpretable by both human and program

Why are ontologies useful?

- Enable semantics of documents to be used by applications and agents
- Standardize metadata terms within a community
- Enable reuse of domain knowledge
- Make domain assumptions explicit

Why are ontologies useful?

- Associations between concepts
- Information extraction
- Information integration
- Automatic reasoning

Reasoning Example

- Parent is a more general relationship than mother
- Mary is the mother of Bob
- Conclusion: Mary is a parent of Bob
- Reasoning allows us to answer query "Who are Bob's parents?"

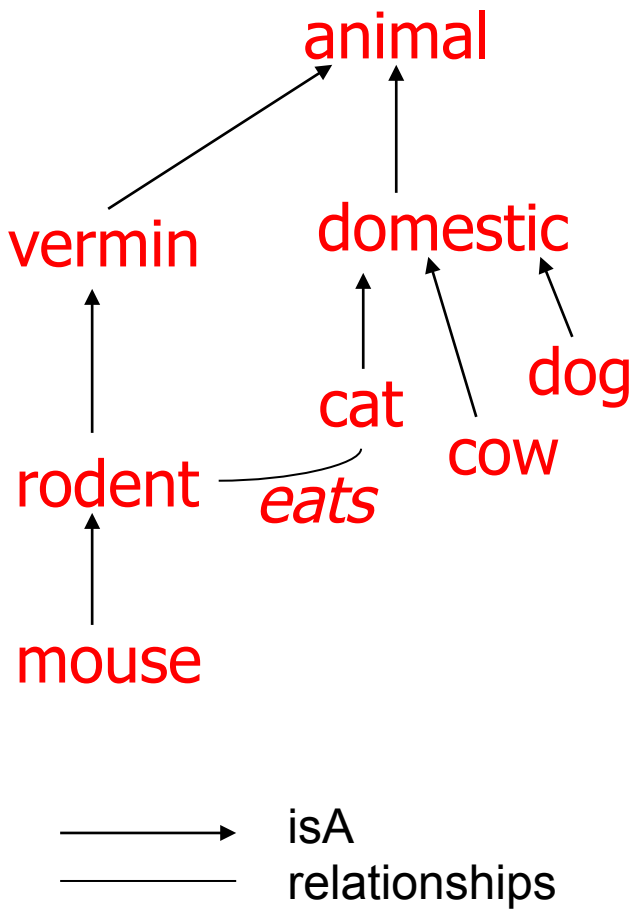
Ontology Modeling
An explicit description of a domain

Concepts (**class**, set, type, predicate)

- event, gene, gammaBurst, atrium, molecule, cat

Properties of concepts and *relationships* between them (**slot**)

- *Taxonomy*: generalisation ordering among concepts *isA*, *partOf*, *subProcess*
- *Relationship*, *Role* or *Attribute*: *functionOf*, *hasActivity* location, *eats*, size



[Carole Goble, Nigel Shadbolt, Ontologies and the Grid Tutorial]

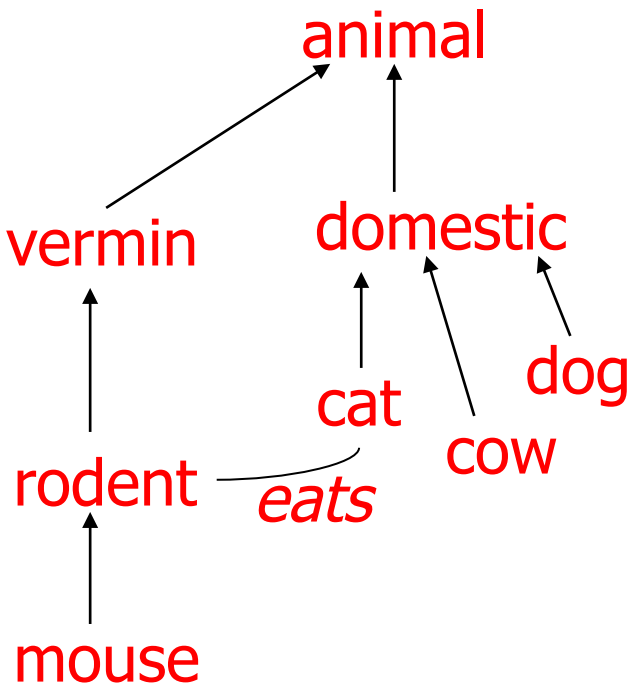
An explicit description of a domain

Constraints or *axioms* on properties and concepts:

- value: integer
- domain: cat
- cardinality: at most 1
- range: 0 <= X <= 100
- oligonucleiotides < 20 base pairs
- cows are larger than dogs
- cats cannot eat only vegetation
- cats and dogs are disjoint

Values or *concrete domains*

- integer, strings
- 20, trypotoplan-synthetase



[Carole Goble, Nigel Shadbolt, Ontologies and the Grid Tutorial]

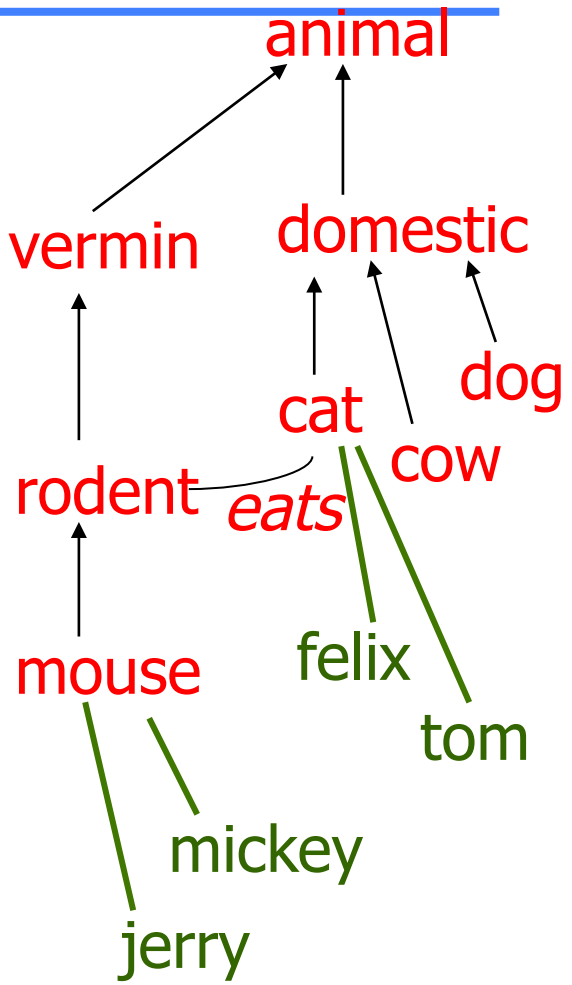
An explicit description of a domain

Individuals or *Instances*

- sulphur, trpA Gene, **felix**

Ontology versus Knowledge Base

- An *ontology* = concepts+properties+axioms +values
- A *knowledge base* = ontology+instances



[Carole Goble, Nigel Shadbolt, Ontologies and the Grid Tutorial]

Ontology terminology that we will use

Class: Description of a **concept** in domain of discourse

- E.g. "Room" in the intelligent space domain

Slot/Property: Describes **attribute** of a concept or **relationship** of a concept

- E.g. longtitude and latitude of a room provides its coordinates
- E.g. isSpatiallySubsumedBy – points to instance of class "Building"

Facet/Restriction: Describes a **constraint** on a slot

- Slot cardinality e.g. one or many values allowed in slot
- Slot value type, e.g. String, Number, Boolean, Enummerated, Instance

More details

- Range from simple taxonomies to complex metadata schemes
- Descriptions for:
  - Classes (general things) in domain of interest
  - Relationships that can exist among things
  - Properties (or attributes) those things may have

OWL Components

- Individuals
  - Objects in a domain
  - Objects with different names *might* be the same
- Properties
  - Binary relations on individuals
  - e.g. hasSibling
  - Can have *inverses* e.g., hasOwner and OwnedBy
  - Can be *transitive*
  - Can be *symmetric*

OWL Components (continued)

- Classes
  - Sets that contain individuals
  - Described using formal descriptions that state requirements for membership
  - May be organized into superclass-subclass hierarchy (taxonomy)
    - Cat is a subclass of animal
  - Superclass-subclass relationships may be computed by a *reasoner*

OWL Schema Features

- *Class* – group of individuals that share some properties
- *rdfs:subClassOf* – create hierarchies
- *rdf:Property* – relationships between individuals
- *rdfs:subPropertyOf:* create property hierarchies
- *rdfs:domain* and *rdfs:range*
- *Individual*

OWL Equality

- *equivalentClass*
- *equivalentProperty*
- *sameAs*
- *differentFrom*
- *AllDifferent*
- *distinctMembers*

OWL Property Characteristics

- *ObjectProperty*
- *DatatypeProperty*
- *inverseOf*
- *TransitiveProperty*
- *SymmetricProperty*
- *FunctionalProperty*
- *InverseFunctionalProperty*

Fundamental Rules in Ontology Design

1) There is no one correct way to model a domain— there are always viable alternatives.
   - The best solution almost always depend on the application that you have in mind and the extensions that you anticipate.

2) Ontology development is necessarily an iterative process.

3) Concepts in the ontology should be close to objects (physical or logical) and relationships in your domain of interest.
   - These are most likely to be nouns (objects) or verbs (relationships) in sentences that describe your domain

Step 1: Determine Scope and Domain of Ontology

What is the domain that the ontology will cover?

- E.g representing an intelligent Universtity space

For what we are going to use the ontology?

- E.g. applications that support roaming students, lecturers and visitors are going to use the ontology

For what types of questions the information in the ontology should provide answers?

- E.g. to help roaming student print to the nearest printer etc.
- E.g. to help the university intelligent space applications to implement energy conservation policies etc.

Who will use and maintain the ontology?

- E.g. if it is maintained by someone in TCD for TCD users but needs also to be used by visiting users from UCD etc., then mappings need to be included

Step 1: Competency Questions

One way to determine scope is to list out the questions the knowledge base based on the ontology should be able to answer

For example

- Which rooms have data projectors?
- Is ORI G.14 a conference room?
- Is there a conference room near restrooms?
- What is the best choice of room for a student tutorial?

Judging from this list of questions, the ontology will include information on

- Various room types and characteristics
- Various event types and characteristics
- Classifications of rooms with respect to appropriateness for events
- Classifications of rooms with respect to location
- Classifications of rooms with respect to furniture/equipment

Step 3: Enumerate important terms in the ontology

List of all terms we would like either to make statements about or to explain to a user.

- What are the terms we would like to talk about?
- What properties do those terms have?
- What would we like to say about those terms?

For example, important a university intelligent space related terms will include

- Building, campus, restroom, room, hallway, stairs, chairs, projectors;
- different types of event, such as lecture, tutorial, seminar, meeting;
- and so on.

Get a comprehensive list of terms without worrying about:

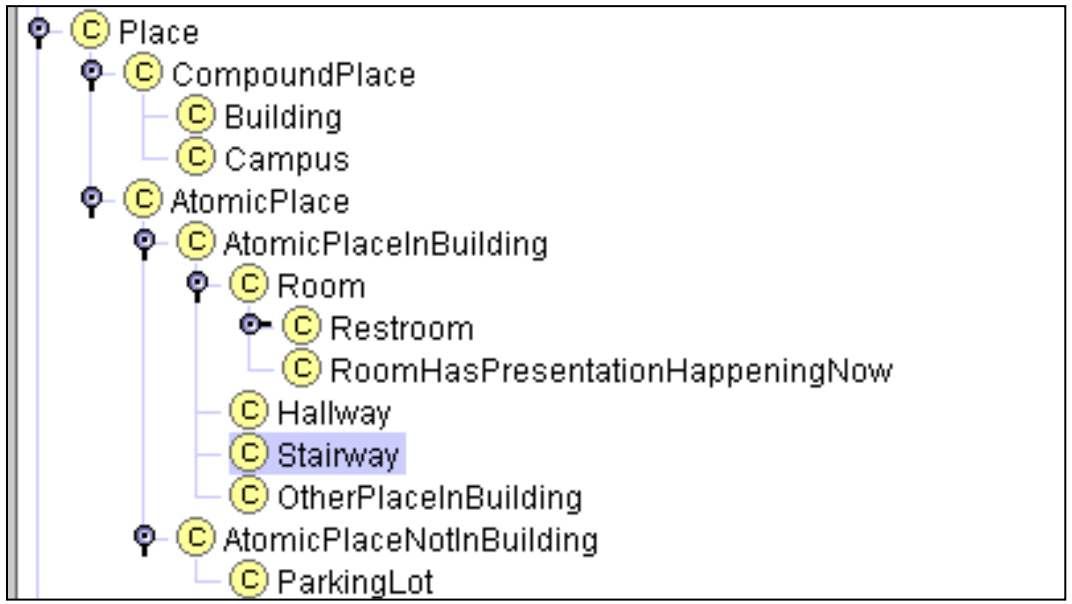
- overlap between concepts they represent,
- relations among the terms,
- or any properties that the concepts may have,
- or whether the concepts are classes or slots.

Step 4: Define the classes and class hierarchy

From the list created in Step 3, we select the terms that describe objects having independent existence rather than terms that describe these objects.

These terms will be classes in the ontology and will become anchors in the class hierarchy

Multiple Inheritance allowed



Step 5: Define the properties of classes - slots

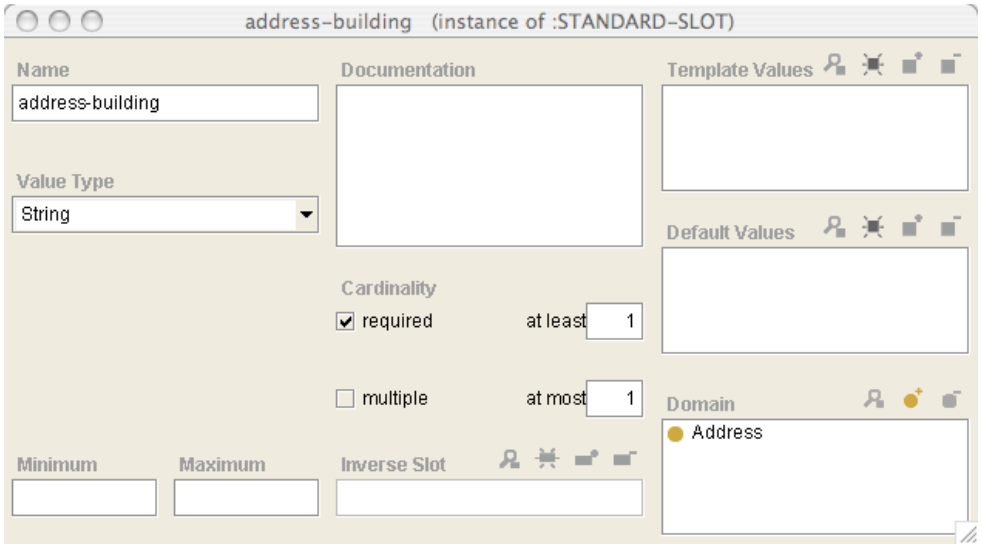
In general, there are several types of object properties that can become slots in an ontology:

- “intrinsic” properties such as the capacity of the room;
- “extrinsic” properties such as a room’s name;
- parts, if the object is structured; these can be both physical and abstract “parts” (e.g., the courses of a meal)
- relationships to other individuals; these are the relationships between individual members of the class and other items (e.g., spatiallySubsumedBy, representing a relationship between a room and a CompoundPlace (subclasses are Building and Campus))

Step 6: Define the Facets of each Slot

Facet: Describes a restriction on a slot

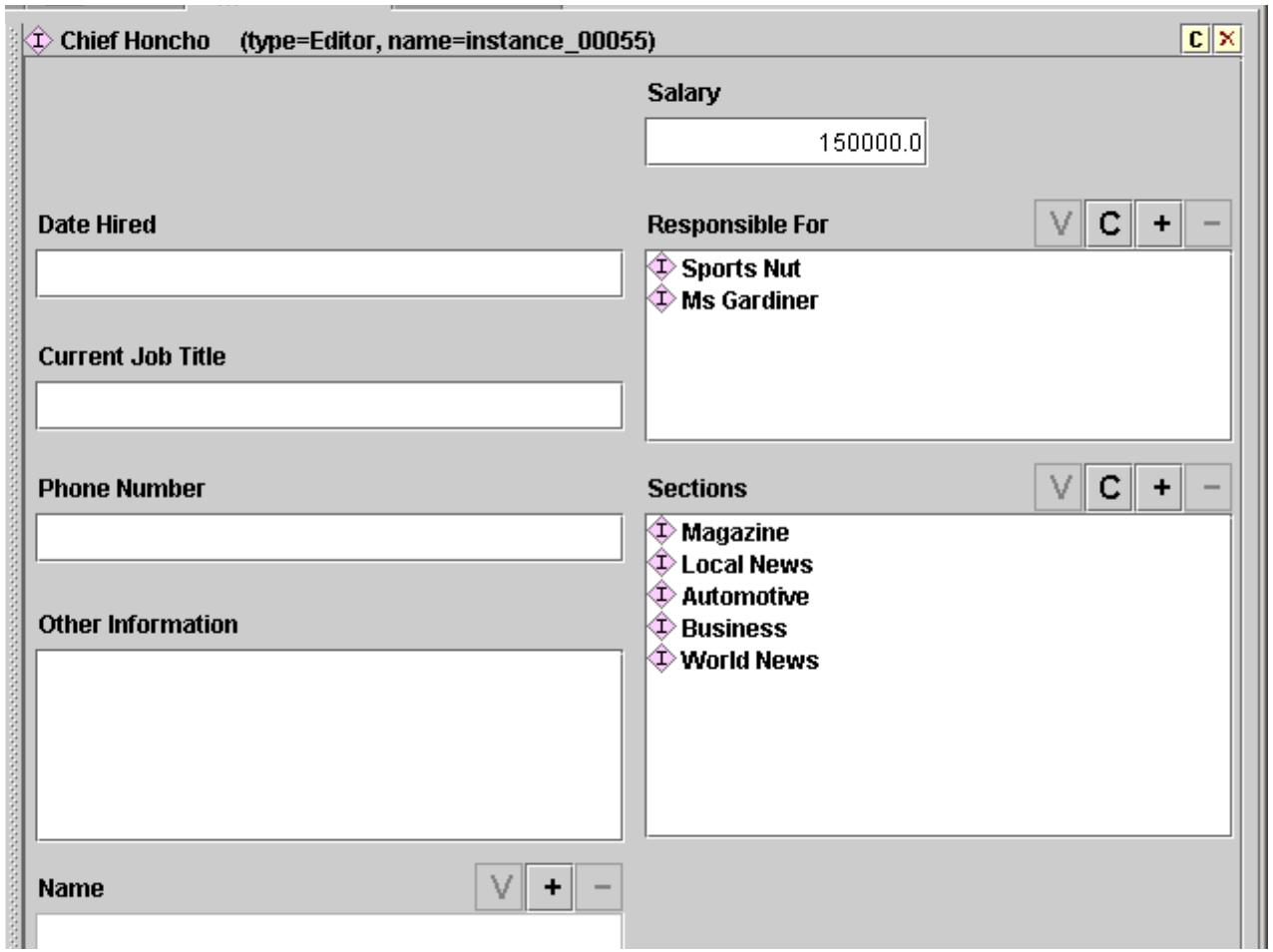
- Slot cardinality e.g. one or many values allowed in slot
- Slot value type, e.g. String, Integer, Boolean, Enummerated, Instance



Step 7: (Optionally) Create Instances

Chief Honcho is an instance of the class Editor

This instance has the following slot values defined



Seven steps to…

1. Determine scope and domain of ontology
2. Consider reusing existing ontologies
3. Ennumerate important terms in the ontology
4. Define the classes and class hierarchy
5. Define the properties of classes – slots
   - Properties, relationships
6. Define the facets of slots
   - Value type (String, Number, Boolean, Enummerated, Instance)
   - Value cardinality
7. Define the instances

Modes of Development

- top-down – define the most general concepts first and then specialize them
- bottom-up – define the most specific concepts and then organize them in more general classes
- combination – define the more salient concepts first and then generalize and specialize them

Slide by N. Noy and S. Tu

Slot and Class Inheritance

- A subclass inherits all the slots from the superclass
  - *If a wine has a name and flavor, a red wine also has a name and flavor*
- If a class has multiple superclasses, it inherits slots from all of them
  - *Port is both a dessert wine and a red wine. It inherits "sugar content: high" from the former and "color:red" from the latter*

Slide by N. Noy and S. Tu

Limiting the Scope

- An ontology should not contain all the possible information about the domain
  - No need to specialize or generalize more than the application requires
  - No need to include all possible properties of a class
    - Only the most salient properties
    - Only the properties that the applications require

Slide by N. Noy and S. Tu

OWL: Web ontology language

- Derived from DAML+OIL
- Adds capabilities compared to RDF:
  - Relations between classes (e.g., disjointness)
  - Cardinality (e.g. exactly one)
  - Equality
  - Rich typing of properties
  - Characteristics of properties
  - Enumerated classes

OWL-DL and OWL-Full Extensions

- oneOf (enumerated classes)
- hasValue (property values)
- disjointWith
- unionOf,complementOf, intersectionOf
- minCardinality, maxCardinality, Cardinality

Example: SWEET Ontologies

- **S**emantic **W**eb for **E**arth and **E**nvironmental **T**erminology
- Many domains including Earth Realm, Physical Phenomena, Sun Realm, Biosphere
- Also include ontologies for units and conversion

OWL Example

```
<owl:Class rdf:ID="OceanRegion">
    <rdfs:subClassOf>
            <owl:Restriction>
                    <owl:onProperty
    rdf:resource="http://sweet.jpl.nasa.gov/ontology/space.owl#isPartOf" />
                    <owl:allValuesFrom rdf:resource="#Ocean" />
            </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
            <owl:Restriction>
                    <owl:onProperty rdf:resource="#hasEcosystemType" />
                    <owl:allValuesFrom rdf:resource="#MarineEcosystem" />
            </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf rdf:resource="#TopographicalRegion" />
</owl:Class>
```

Protégé

- Free, open source ontology editor and knowledge-based framework
- Protégé-frames:
  - Constructing and storing frame-based ontologies (represent info as objects)
  - Entering instance data
- Protégé-OWL:
  - Load and save OWL and RDF ontologies
  - Edit and visualize classes, properties
  - Execute reasoners

BioPortal Ontologies Search Annotator Recommender Mappings Login Support

Search for a class

Enter a class, e.g. Melanoma

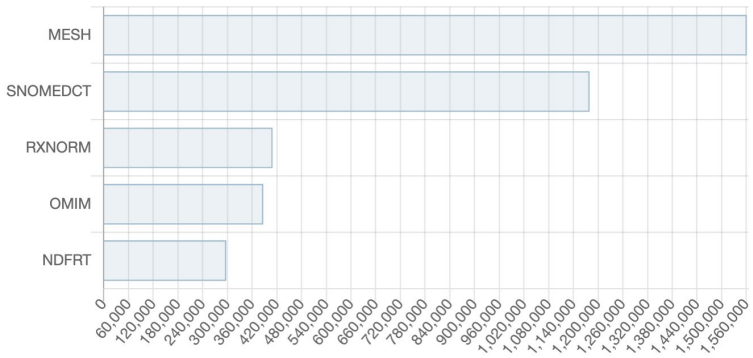
Advanced search

Find an ontology

Start typing ontology name, then choose from list

Browse ontologies

Ontology visits (September 2026)



More

Statistics

Ontologies	1,306
Classes	12,284,787
Properties	169,142
Individuals	2,173,040
Mappings	63,418,549

Ontology Development and Learning using Generative AI

Manas Gaur

What is ontology story

An example ontology story from the semantic web course

During each year a number of theatre festivals are held in cities around Italy. In January 2007 …

Competency questions

When and **Where** did a certain theatre festival take place?

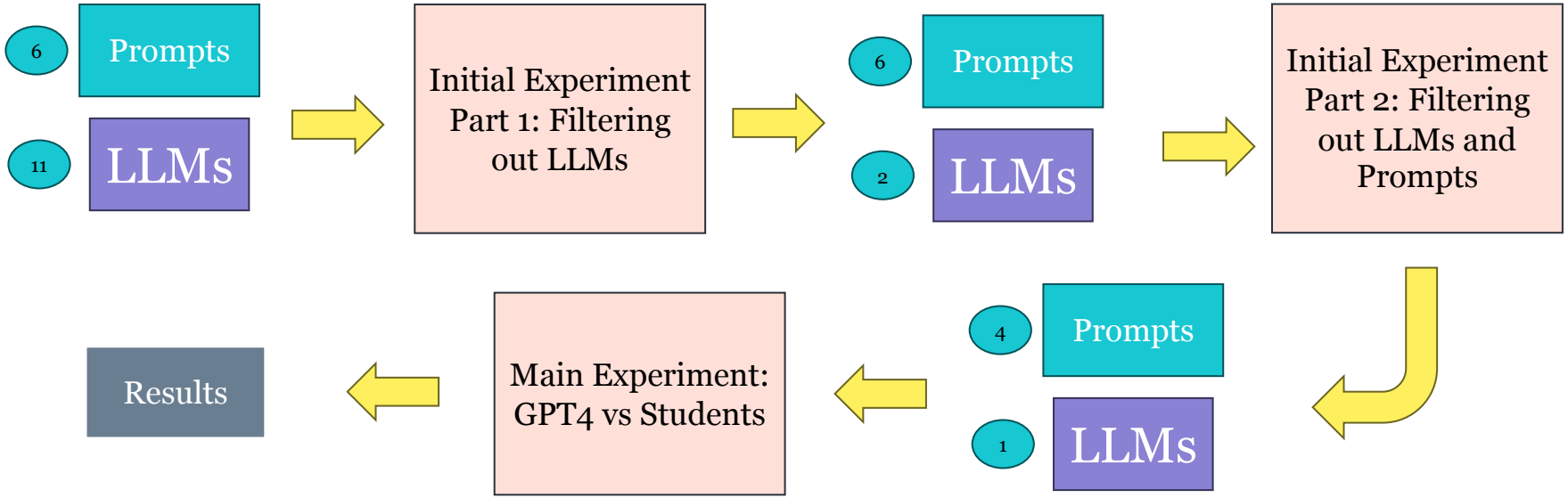
Research Questions

1- To what extent can LLMs create an ontology that meets the Ontology Requirements?

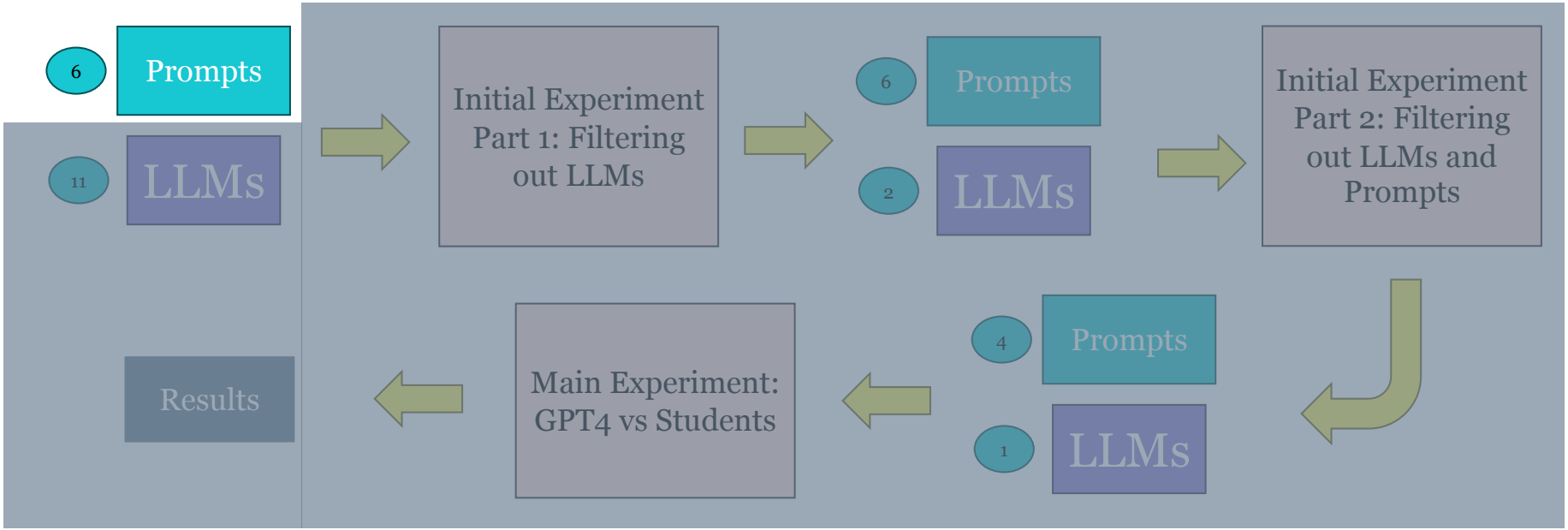
2- Which LLMs are suitable for this task?

3- What prompting techniques are the most effective?

Structure of the presentation



Structure of the presentation



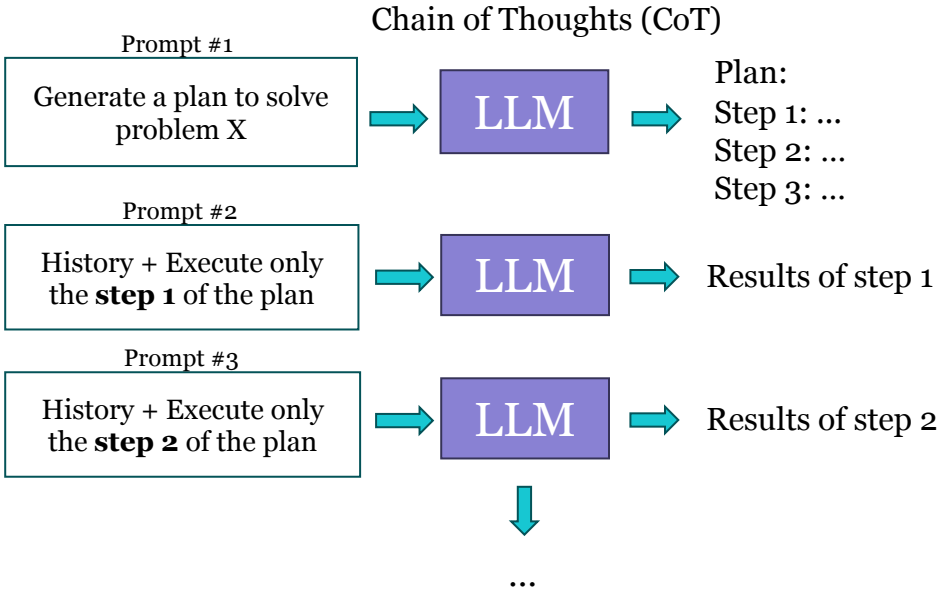
What is a prompt?



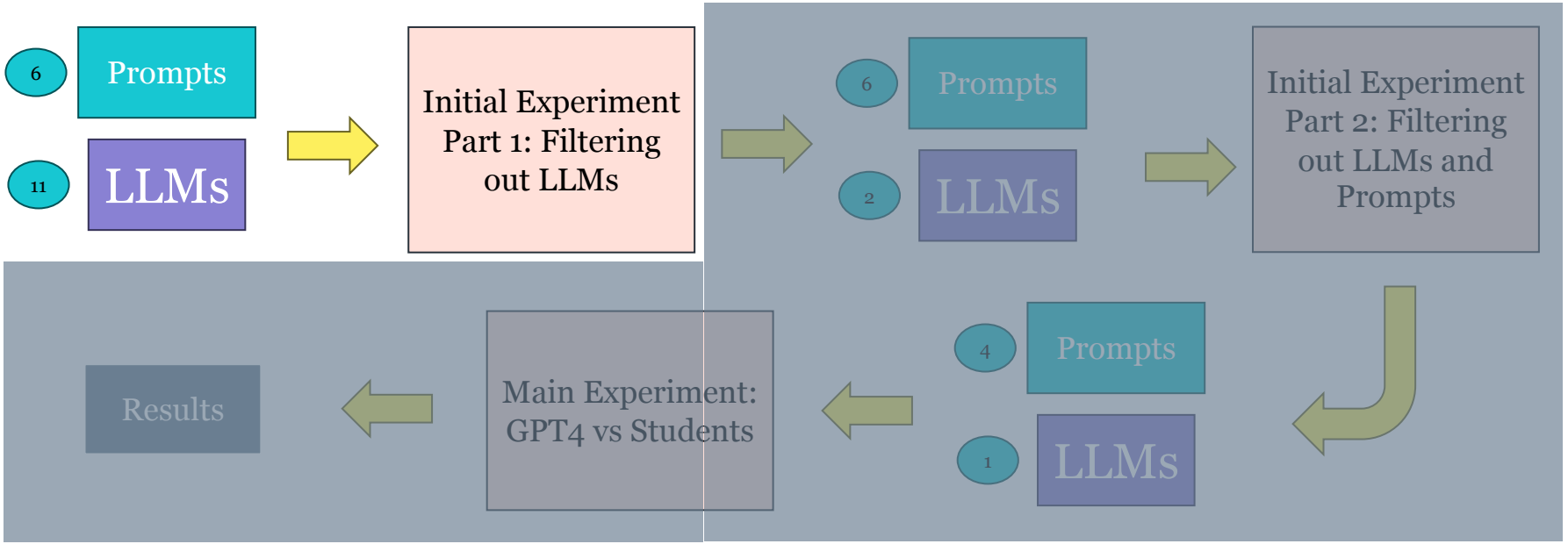
What is a prompting technique?



- Zero-Shot
- Chain of Thoughts (**CoT**)
- Chain of Thoughts Self-Consistency (**CoT-SC**)
- Sub-task Decomposed Prompting - Waterfall approach (**Waterfall**)
- Sub-task Decomposed Prompting - Competency Question by Competency Question (**CQbyCQ**)
- Graph of Thoughts (**GoT**)

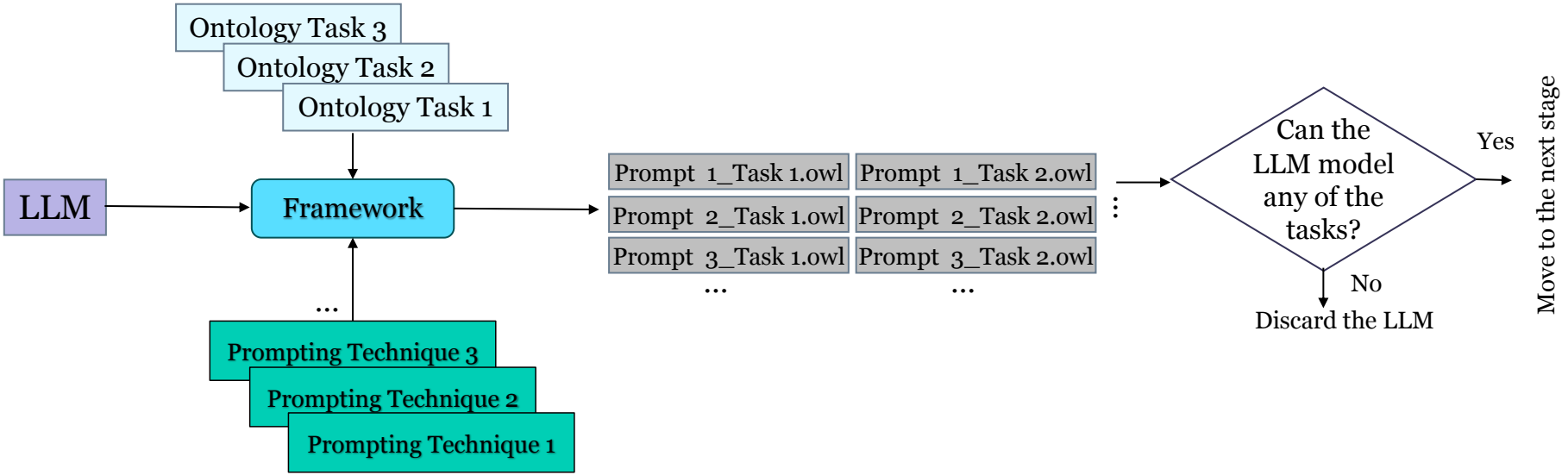


Structure of the presentation



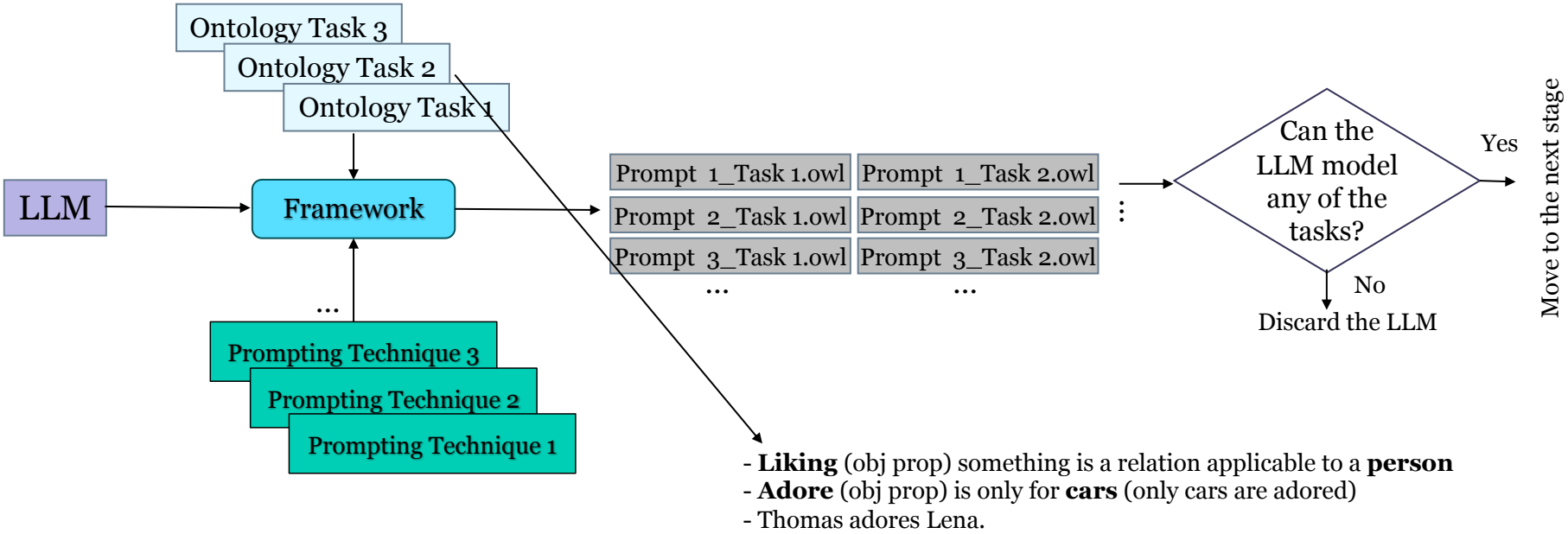
Initial Experiment: Phase 1

Which LLMs are suitable for ontology generation?

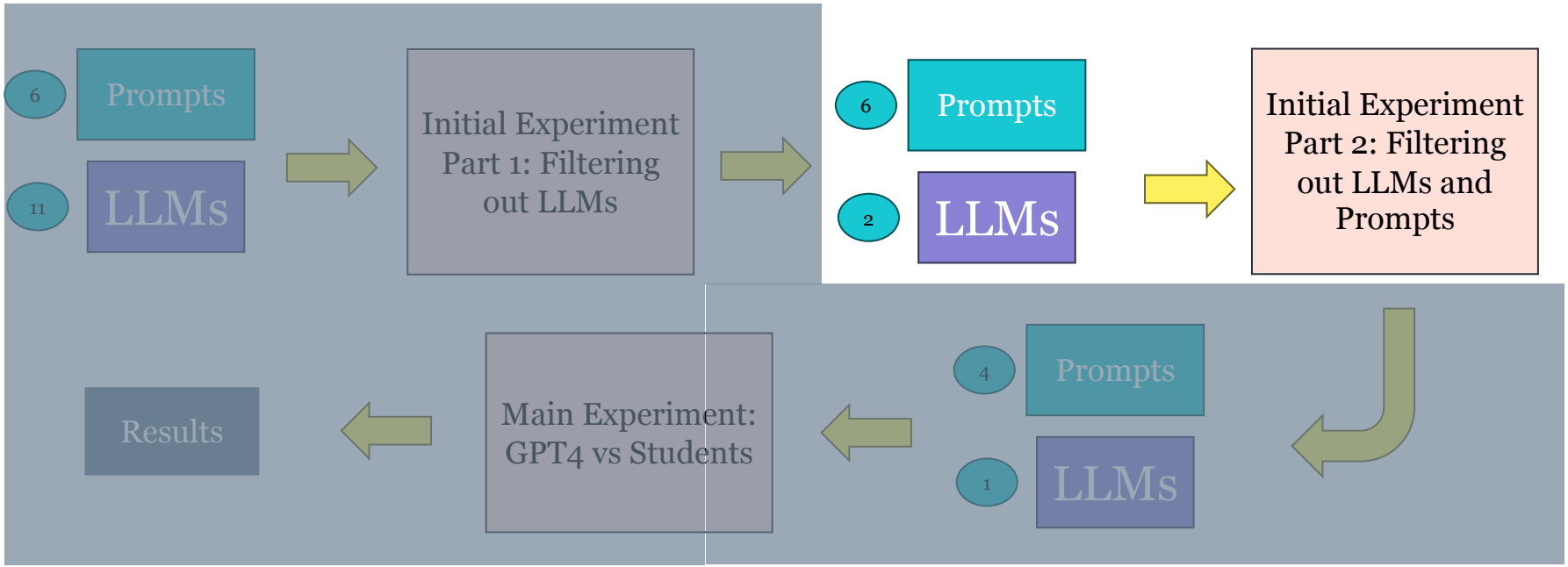


Initial Experiment: Phase 1

Which LLMs are suitable for ontology generation?

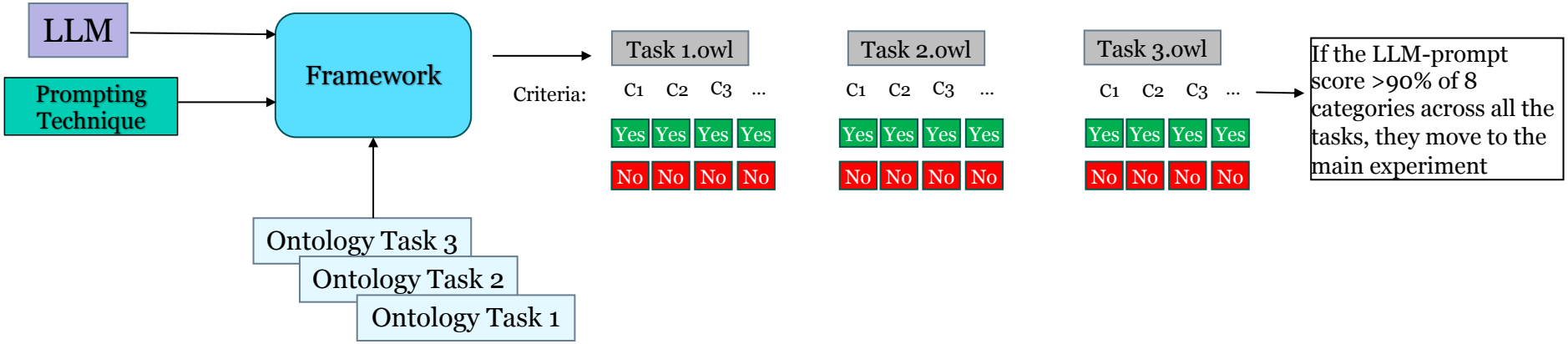


Structure of the presentation



Initial Experiment: Phase 2

Which pair of LLMs and prompts are suitable for ontology generation?



1: Correctness of the Turtle syntax.
2: Usage taxonomy (class hierarchy) in the ontology
3: Usage of Data Properties (if needed)
4: Presence of reification class

5: Presence of instances (if needed)
6: Establishment of domain or range restriction for properties
7: Presence of “EquivalenceClass” restriction
8: Semantic coherence of class hierarchy

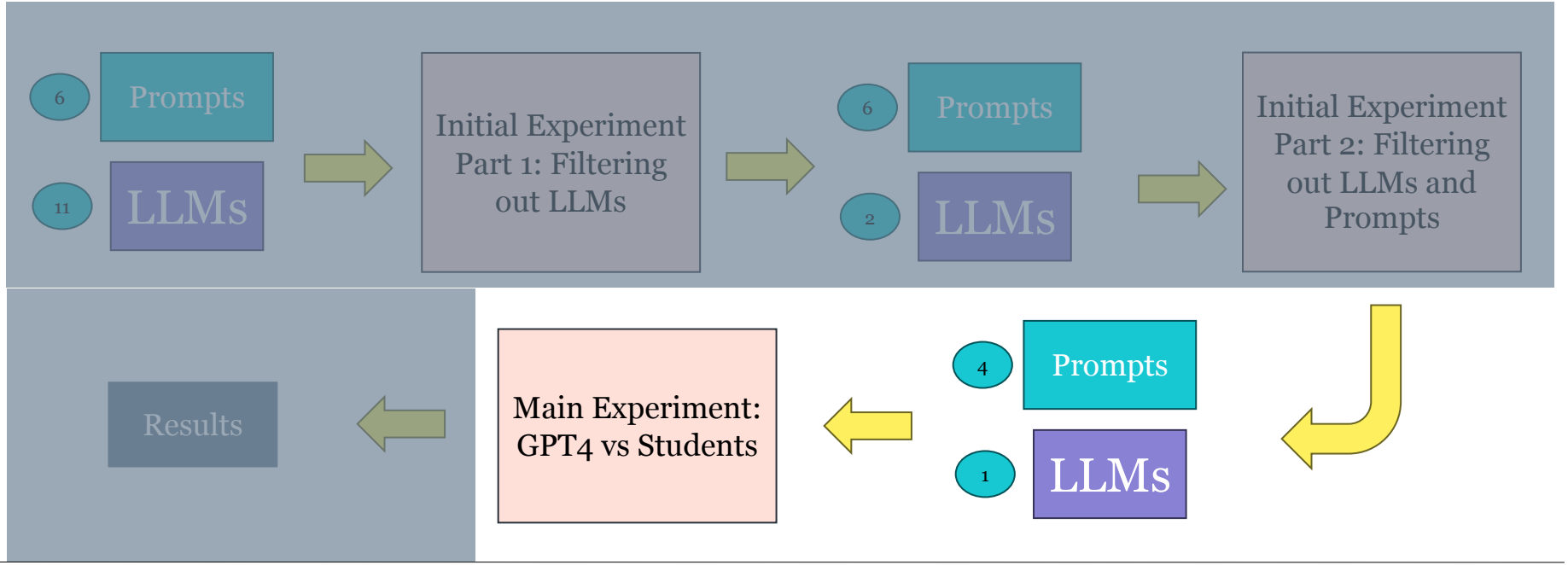
Result of Initial Experiment

LLMs: **GPT-4**

Prompting Techniques:

- **CoT** (chain of thoughts)
- **CoT-SC** (Chain of Thoughts Self-Consistency)
- **CQbyCQ** (Sub-task Decomposed Prompting - Competency Question by Competency Question)
- **GoT** (Graph of Thoughts)

Structure of the presentation



Baseline

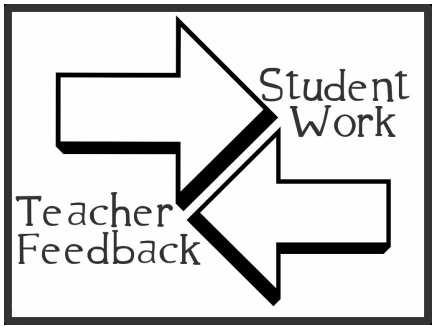
Baseline: Master's students who took a **semantic web course**

Course: **3** ontology tasks/stories, **15 competency** questions each

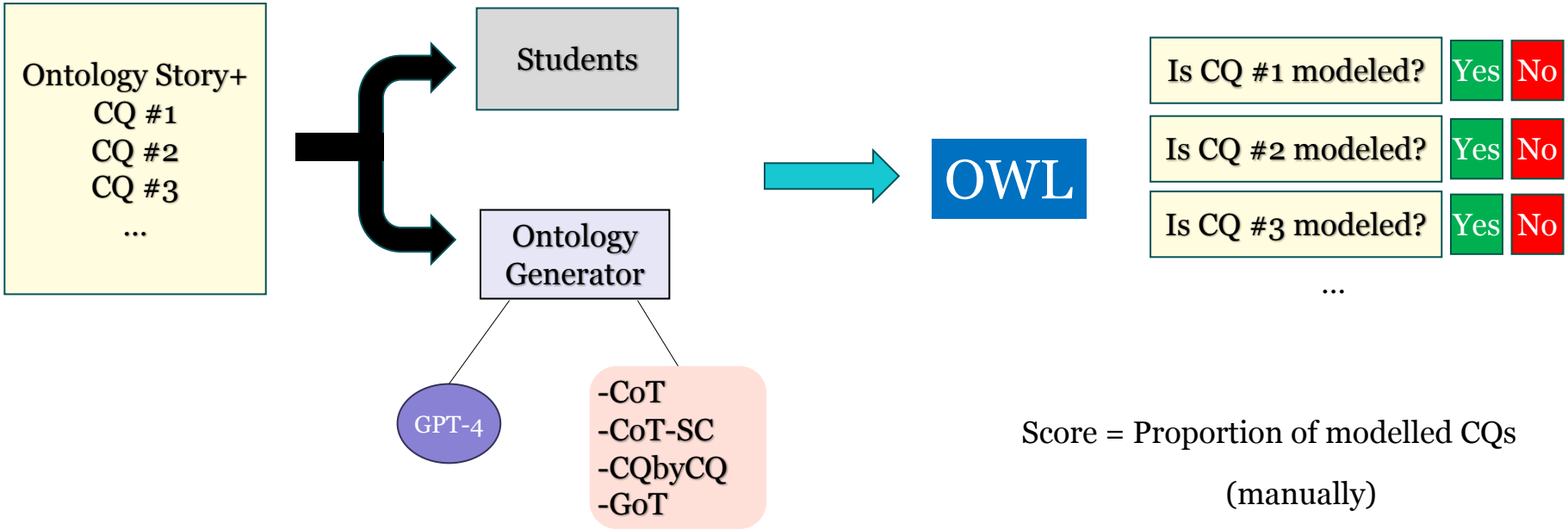
10 groups, and **2 students** on average in each group

Students submitted each modelling on average **3 times** to pass the lab

The baseline is the **first** and the **last student submissions** of the course.



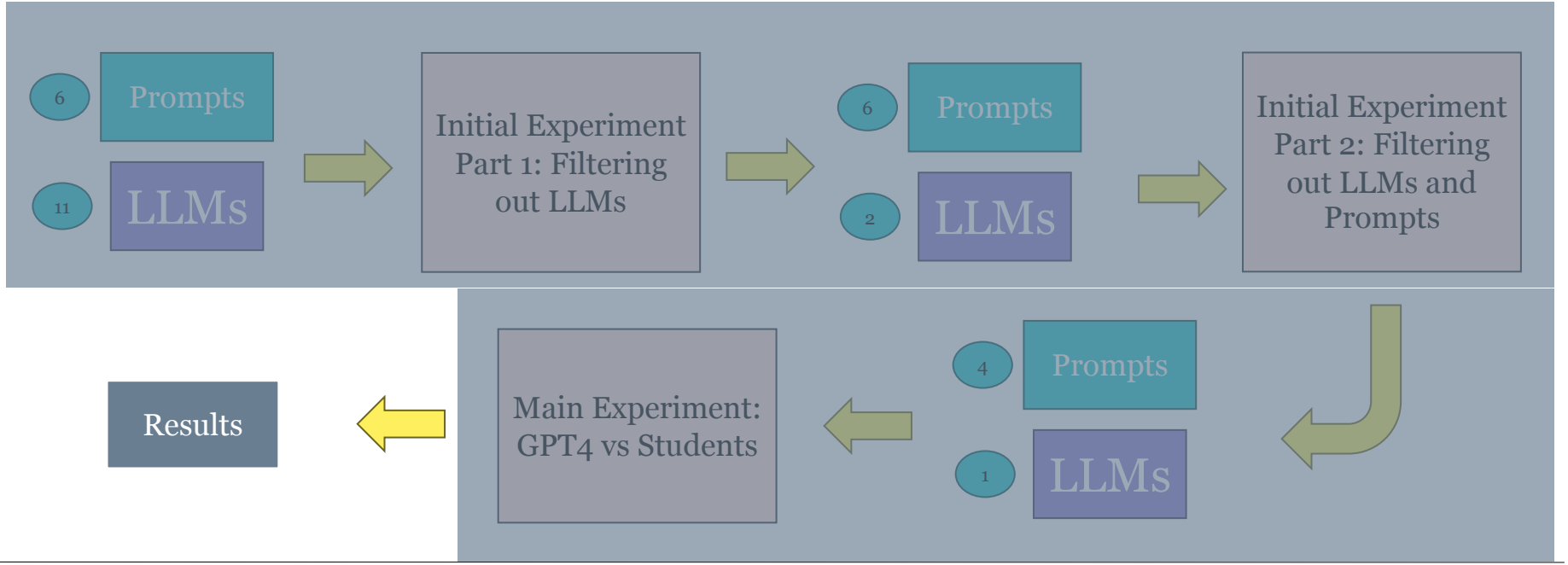
Evaluation Criteria



Score = Proportion of modelled CQs

(manually)

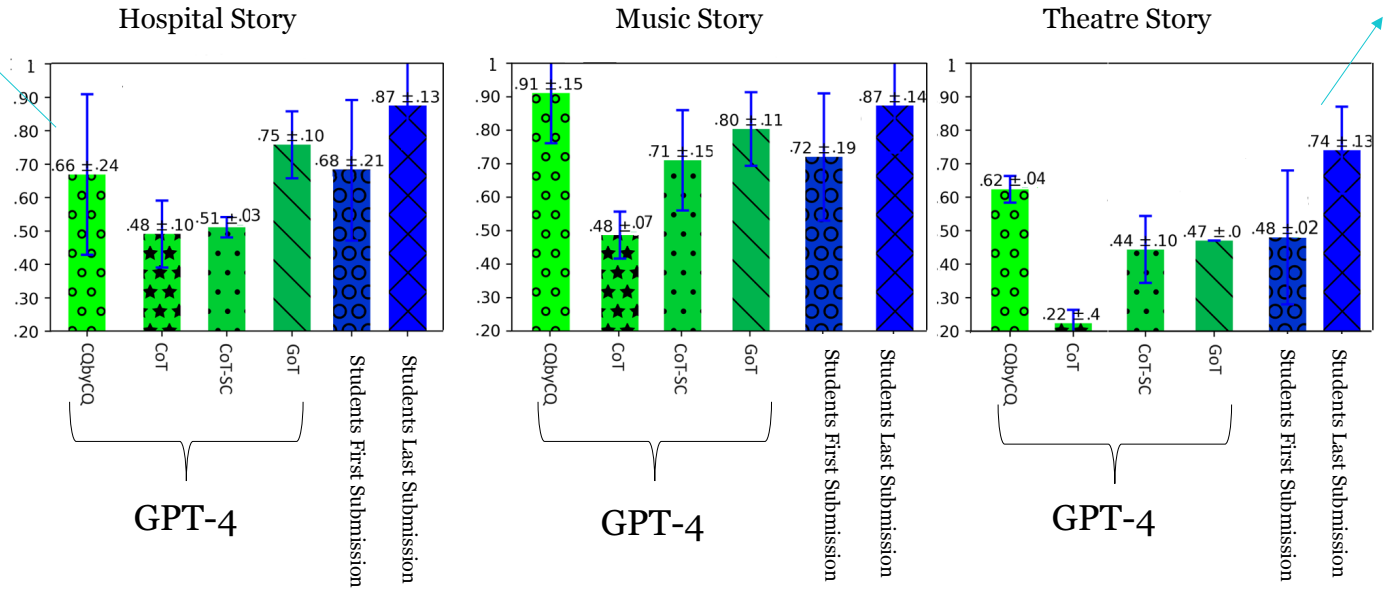
Structure of the presentation



Results

Std: we ran models 3 times

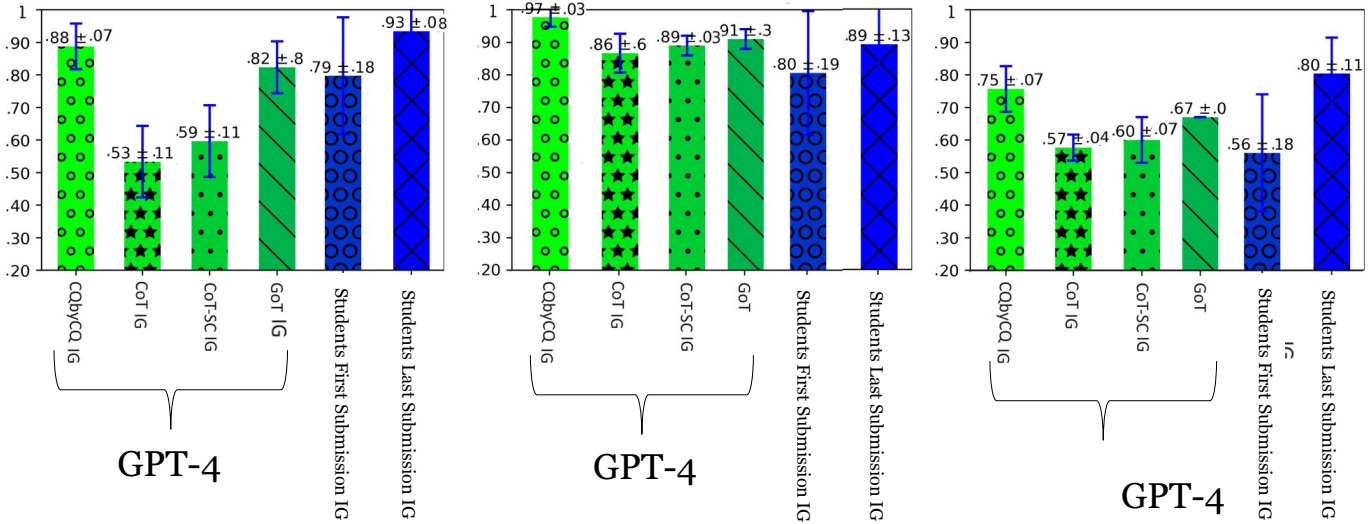
Std of students' submissions



CQbyCQ	GoT	Stu F S	Stu L S
0.73	0.67	0.62	0.82

Results with a relax criteria

missing one single data/object property



CQbyCQ	GoT	Stu F S	Stu L S
0.86	0.80	0.71	0.87

Evaluation: types of CQs

	Theatre Story				Music Story				Hospital Story			
	DP	OP	Reif.	Rest.	DP	OP	Reif.	Rest.	DP	OP	Reif.	Rest.
CQbyCQ	**.93**	**.78**	.55	.33	**1.0**	**.94**	**1.0**	**1.0**	**1.0**	.94	**1.0**	0
CoT	.80	.50	.22	**1.0**	.80	.78	**1.0**	**1.0**	.53	.50	.44	**1.0**
CoT-SC	**.93**	.55	0	**1.0**	.80	.89	**1.0**	**1.0**	.53	.66	.66	.33
GoT	.86	.61	.44	**1.0**	.80	**.94**	**1.0**	**1.0**	.73	**1.0**	.66	.66
Students First Submission	.64	.63	.43	.10	.84	.83	.87	.30	.78	.85	.73	.60
Students Last Submission	.92	**.78**	**.73**	.50	.94	.92	.90	.50	.94	**1.0**	.90	.60

Simpler
- DP: Simple data property modeling
- OP: Simple object property modeling

More Complex
- Reif.: Reifications
- Rest.: Restrictions on classes or properties

Evaluation: types of CQs

	Theatre Story				Music Story				Hospital Story			
	DP	OP	Reif.	Rest.	DP	OP	Reif.	Rest.	DP	OP	Reif.	Rest.
CQbyCQ	**.93**	**.78**	.55	.33	**1.0**	**.94**	**1.0**	**1.0**	**1.0**	.94	**1.0**	0
CoT	.80	.50	.22	**1.0**	.80	.78	**1.0**	**1.0**	.53	.50	.44	**1.0**
CoT-SC	**.93**	.55	0	**1.0**	.80	.89	**1.0**	**1.0**	.53	.66	.66	.33
GoT	.86	.61	.44	**1.0**	.80	**.94**	**1.0**	**1.0**	.73	**1.0**	.66	.66
Students First Submission	.64	.63	.43	.10	.84	.83	.87	.30	.78	.85	.73	.60
Students Last Submission	.92	**.78**	**.73**	.50	.94	.92	.90	.50	.94	**1.0**	.90	.60

Simpler
- DP: Simple data property modeling
- OP: Simple object property modeling

More Complex
- Reif.: Reifications
- Rest.: Restrictions on classes or properties

Ontology Learning

Manas Gaur

Background

- **Ontology Learning (OL)** involves automatically identifying terms, types, relations, and potential axioms from textual information to construct an ontology.
- Traditional OL approaches in the field are based on **lexico-syntactic pattern mining** and **clustering** [1].
- In contrast, recent advances in NLP through LLMs offer a promising alternative to traditional OL methods.

[1]

Hypothesis

- Can LLMs effectively apply their language pattern capturing capability to OL, which involves automatically extracting and structuring knowledge from natural language text?

- Experimental Considerations:
  - Do the characteristics of LLMs justify ontology learning?
  - Which LLMs to test for the LLMs4OL task hypothesis?

Prompting – Prompt Templates

Standard prompting to initial exploration of the tasks:

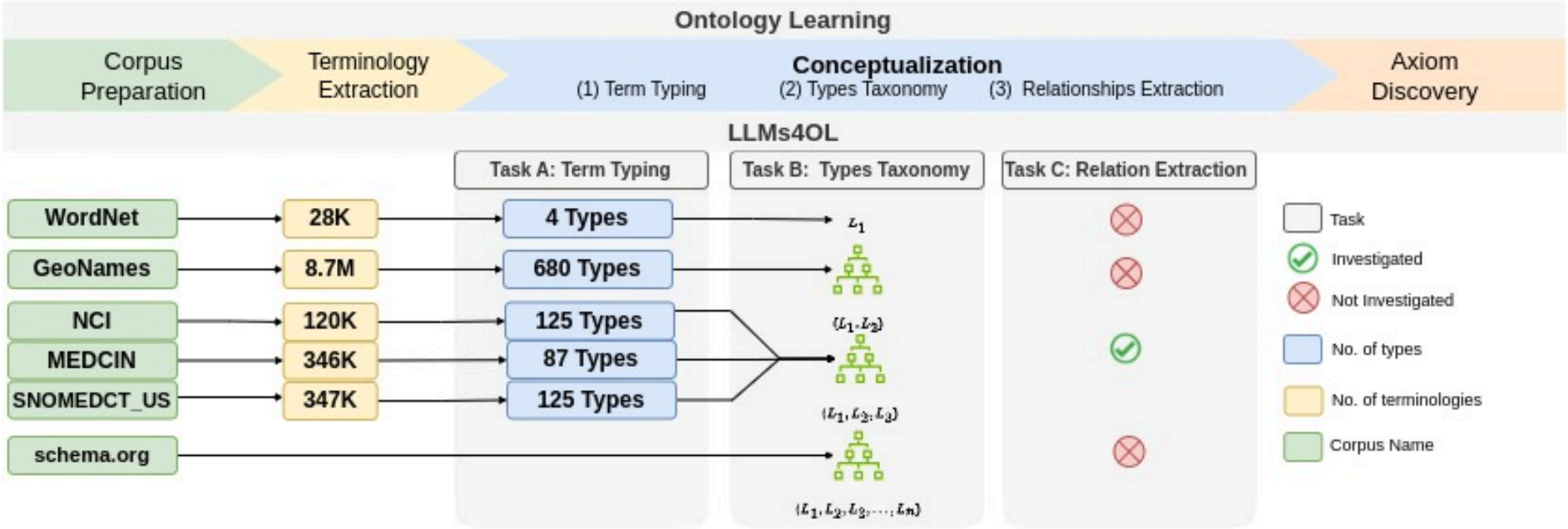
- TASK A: Term Typing – Generalized types are discovered for a lexical term.
  $f^{A}_{cloze-prompt}(L) := [S?].\ [L]\ [P_{domain}]\ is\ a\ \ [MASK].$ [2]
  $f^{A}_{prefix-prompt}(L) = [instruction] + f^{A}_{cloze-prompt}(L)$
- TASK B: Type Taxonomy Discovery – Taxonomic hierarchy between pairs of types are discovered.
  $f^{B}_{cloze-prompt}(a, b) := [a|b]\ is\ [P_{hierarchy}]\ of\ \ [b|a].\ This\ statement\ is\ [MASK].$ [3]
  $f^{B}_{prefix-prompt}(a, b) = [instruction] + f^{B}_{cloze-prompt}(a, b)$
- TASK C: Type Non-Taxonomic Relation Extraction – Non-taxonomic semantic heterarchical relations between types are discovered.
  $f^{C}_{cloze-prompt}(h, r, t) := [h]\ is\ [r]\ [t].\ This\ statement\ is\ [MASK].$ [4]
  $f^{C}_{prefix-prompt}(h, r, t) = [instruction] + f^{C}_{cloze-prompt}(h, r, t)$

[2] L is lexical term, $S?$ is an optional context, and P_{domain} is a domain specification

[3] $P_{hierarchy}$ indicate superclass relation, (a, b) are type pairs

[4] r is the non-taxonomic relation between head type h and target type t

LLMs4OL Paradigm



- Task A: **Term Typing** – How effective are LLMs for automated type discovery to construct an ontology? – RQ1
- Task B: **Type Taxonomy Discovery** – How effective are LLMs to recognize a type taxonomy i.e. the "is-a" hierarchy between types? – RQ2
- Task C: **Type Non-Taxonomic Relation Extraction** – How effective are LLMs to discover non-taxonomic relations between types? – RQ3

Prompting · Prompt Templates

$f_{cloze-prompt}(L) := [S?].\ [L]\ [P_{domain}]$ *is a* $[MASK]$.
$f^{A}_{prefix-prompt}(L) = [instruction] + f^{A}_{cloze-prompt}(L)$

- TASK B: Type Taxonomy Discovery – Taxonomic hierarchy between pairs of types are discovered.
  $f^{B}_{cloze-prompt}(a, b) := [a|b]$ *is* $[P_{hierarchy}]$ *of* $[b|a]$. *This statement is* $[MASK]$.[3]
  $f^{B}_{prefix-prompt}(a, b) = [instruction] + f^{B}_{cloze-prompt}(a, b)$
- TASK C: Type Non-Taxonomic Relation Extraction – Non-taxonomic semantic heterarchical relations between types are discovered.
  $f^{C}_{cloze-prompt}(h, r, t) := [h]$ *is* $[r]$ $[t]$. *This statement is* $[MASK]$. [4]
  $f^{C}_{prefix-prompt}(h, r, t) = [instruction] + f^{C}_{cloze-prompt}(h, r, t)$

[2] L is lexical term, $S?$ is an optional context, and P_{domain} is a domain specification
[3] $P_{hierarchy}$ indicate superclass relation, (a, b) are type pairs

Ontological Sources · Task A Datasets

- 5 sources, 3 domains: Lexicosemantics (WordNet) – Geography (GeoNames) – Biomedicine (UMLS subontologies – NCI, MEDICIN, SNOMEDCT_US)
- Dataset stats:

	WordNet	**GeoNames**	**NCI**	**MEDCIN**	**SNOMEDCT_US**
Train Set Size	40,559	8,078,865	96,177	277,028	278,374
Test Set Size	9,470	702,510	24,045	69,258	69,594
Types	4	680	125	87	125

Ontological Sources · Task B Datasets

- 3 sources, 3 domains: Geography (GeoNames) – Biomedicine (UMLS)–Web content types (schema.org)
- Dataset Construction Method:
  1. Lets consider:
     - aRb as *"a is a superclass of type b"* and $b\neg Ra$ "b is a superclass of type a"
     - T_n as a collection of types at level n of taxonomy, where $n+2 > n+1 > n$, n is the root.
  2. $\forall(a \in T_n, b \in T_{n+1}) \longmapsto aRb \wedge b\neg Ra$
     - aRb (positive label) & $b\neg Ra$ (negative label)
  3. $\forall(a \in T_n, b \in T_{n+1}, c \in T_{n+2}); aRb \wedge bRc \longmapsto aRc$
  4. $\forall(a \in T_n, b \in T_{n+1}, c \in T_{n+2}); c\neg Rb \wedge b\neg Ra \longmapsto c\neg Ra$
- Dataset stats:

	GeoNames	**UMLS**	**schema.org**
Types	689	127	797
Levels	2	3	6
Positive/Negative Samples	680/680	254/254	2,670/2,670
Train/Test split	272/1,088	101/407	1,086/4,727

Ontological Sources · Task C Datasets

- 1 source, 1 domain: Biomedicine (UMLS)
- UMLS reports 53 non-taxonomic relations across its 127 term types.
- Comprised all pairs of types for each relation, where for any given relation some pairs are true while the rest are false candidates.
- Dataset stats:

Task	**UMLS**
Non-Taxonomic Relations	53
Positive/Negative Samples	5,641/1,896
Train/Test Split	1,507/6,030

Moving Forward · Which LLMs?

Evaluating different LLMs supports performance comparison, model improvement, and research advancement.

- **Encoder**: BERT-Large (340M), PubMedBERT (Domain Specific)
- **Encoder-Decoder**: BART-Large (400M), Flan-T5-Large (780M), Flan-T5-XL (3B)
- **Decoder**: BLOOM-1b7 (1.7B), BLOOM-3b (3B), LLaMA (3B), GPT-3 (175B), GPT-3.5 (174B), GPT-4 (>1T)

Zero-Shot Testing · Results

Task	Dataset	BERT-Large	PubMedBERT	BART-Large	Flan-T5-Large	Flan-T5-XL	BLOOM-1b7	BLOOM-3b	GPT-3	GPT-3.5	LLaMA-7B	GPT-4
A	WordNet	27.9	-	2.2	31.3	52.2	79.2	79.1	37.9	**91.7**	81.4	90.1
	GeoNames	38.3	-	23.2	13.2	33.8	28.5	28.8	22.4	35.0	29.5	**43.3**
	NCI	11.1	5.9	9.9	9.0	9.8	12.4	15.6	12.7	14.7	7.7	**16.1**
	SNOMEDCT	21.1	28.5	19.8	24.3	31.6	37.0	**37.7**	24.4	25.0	13.8	27.8
	MEDCIN	8.7	15.6	12.7	13.0	18.5	28.8	**29.8**	25.7	23.9	4.9	23.7
B	GeoNames	54.5	-	55.4	59.6	52.4	36.7	48.3	53.2	**67.8**	33.5	55.4
	UMLS	48.2	33.7	49.9	55.3	64.3	38.3	37.5	51.6	70.4	32.3	**78.1**
	schema.org	44.1	-	52.9	54.8	42.7	48.6	51.3	51.0	**74.4**	33.8	74.3
C	UMLS	40.1	42.7	42.4	46.0	**49.5**	43.1	42.7	38.8	37.5	20.3	41.3

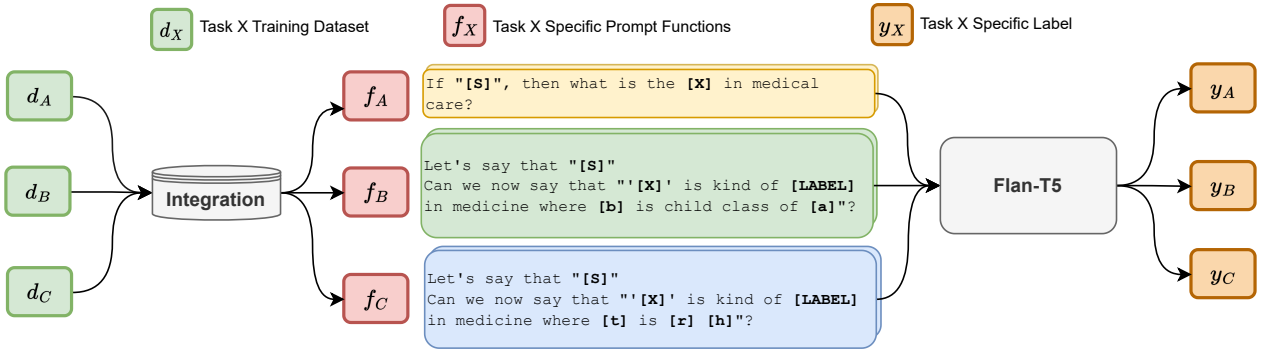
- High performance on WordNet (task A) – simple type space.
- Promising result on GeoNames (task A) - more fine-grained type space.
- Domain-specific PubMedBERT underperforms in UMLS
- Better performance of larger models on Task B than Task C.

Zero-Shot Testing · Findings

- Results indicate that LLMs seem promising for OL.
- They would need task-specific finetuning to be a practically viable solution.

Finetuning LLM · Flan-T5

- Adapted instruction tuning workflow as described in Flan-T5 [5]
- Instructions are instantiated from a small selection of samples using tasks train set for each knowledge source

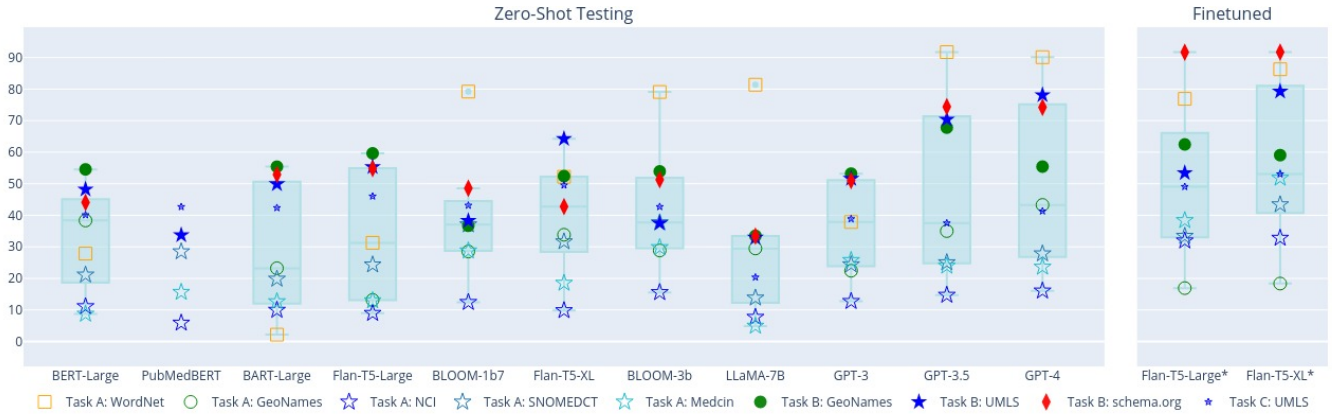


[5]

Finetuning LLM · Results

Task	Dataset	Zero-Shot Test	Flan-T5-Large	Flan-T5-XL
	WordNet	**91.7 (GPT-3.5)**	76.9	86.3
	GeoNames	**43.3 (GPT-4)**	16.9	18.4
A	*NCI*	16.1 (GPT-4)	31.9	**32.8**
	SNOMEDCT_US	37.7 (BLOOM-3b)	33.4	**43.4**
	MEDCIN	29.8 (BLOOM-3b)	38.4	**51.8**
	GeoNames	**67.8 (GPT-3.5)**	62.5	59.1
B	*UMLS*	78.1 (GPT-4)	53.4	**79.3**
	Schema.Org	74.4 (GPT-3.5)	91.7	**91.7**
C	*UMLS*	49.5 (Flan-T5-XL)	49.1	**53.1**

Error Analysis · Limitations



- LLaMA-7B overall low performance – Absence of few-shot testing
- NCI low performance in most of the evaluations – Heuristics-based generated answer set mapping to the gold standard
- Finetunning - low number of training shots in finetuning

Conclusion

- We explore LLMs' potential for Ontology Learning through our introduced conceptual framework, LLMs4OL.
- Our empirical results are promising to pave future work for OL:
  - A needs to enhance LLMs specifically for OL tasks.
  - Expanding evaluations to cover more diverse knowledge domains.
  - A hybrid approach that combines LLMs with traditional OL techniques.
  - Standardizing evaluation metrics, creating benchmark datasets