Semi-Automatic Information and Knowledge Systems: Ontology Mapping & Alignment

Monika Lanzenberger

Outline

- Introduction
- Similarity Layers
- General Alignment Process
- Application Scenarios
- Complexity & Evaluation
- Using Mappings

Introduction

- Ontology Mapping: for each ontological entity in the first ontology, we try to locate a corresponding entity in the second ontology, with the same or the closest semantics. It constitutes a fragment of more ambitious tasks such as the alignment of ontologies.

- Ontology Alignment: bringing two ontologies into mutual agreement, making them consistent and coherent with one and another. It may often include a transformation of the source ontologies removing the unnecessary information and integrating missing information.

- Whereas alignment merely identifies the relation between ontologies, mappings focus on the representation and the execution of the relations for a certain task.

Introduction: Similarity of Entities

- Entities are the same, if their features are the same or similar enough.

- Features represent a certain meaning

- Low similarity may not give evidence for alignments

- High similarity may give strong evidence for alignments

- Not every similarity is of equal importance
### Introduction: Relations Among Concepts

<table>
<thead>
<tr>
<th>OWL Ontology Construct</th>
<th>Comparison Relationship</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept</td>
<td>Equal</td>
<td>URI's equal. Class member instances equal.</td>
</tr>
<tr>
<td></td>
<td>Syntactically equal</td>
<td>Labels are the same.</td>
</tr>
<tr>
<td></td>
<td>Similar</td>
<td>Superclasses are the same. Subclasses are the same.</td>
</tr>
<tr>
<td></td>
<td>Data properties</td>
<td>Data properties are the same. Object properties are the same.</td>
</tr>
<tr>
<td></td>
<td>Similar low/high fraction of instances</td>
<td></td>
</tr>
<tr>
<td>Broader than</td>
<td>Subclass superclass comparison.</td>
<td></td>
</tr>
<tr>
<td>Narrower than</td>
<td>Subclass subclass comparison.</td>
<td></td>
</tr>
<tr>
<td>Different</td>
<td>Class is different from all classes of the second ontology.</td>
<td></td>
</tr>
</tbody>
</table>

### Introduction: Relations Among Instances

<table>
<thead>
<tr>
<th>Instances</th>
<th>Comparison Relationship</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Equal</td>
<td>URI's equal. Labels are the same.</td>
</tr>
<tr>
<td></td>
<td>Syntactically equal</td>
<td>Instances of the same concept. Property members are the same.</td>
</tr>
<tr>
<td></td>
<td>Similar</td>
<td>Two instances linked via the same property to another instance.</td>
</tr>
<tr>
<td>Different</td>
<td></td>
<td>Instance is different from all instances of the second ontology.</td>
</tr>
</tbody>
</table>

### Introduction: Relations Among Data Properties

<table>
<thead>
<tr>
<th>OWL Ontology Construct</th>
<th>Comparison Relationship</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Properties</td>
<td>Equal</td>
<td>URI's equal. Labels are the same. Data property domain are the same. Data super properties are the same. Data sub properties are the same. Data properties members are the same.</td>
</tr>
<tr>
<td></td>
<td>Syntactically equal</td>
<td>Object property domains are the same. Object super properties are the same. Object sub properties are the same. Object properties members are the same.</td>
</tr>
<tr>
<td></td>
<td>Similar</td>
<td></td>
</tr>
<tr>
<td>Different</td>
<td></td>
<td>Object property is different from all object properties of the second ontology.</td>
</tr>
</tbody>
</table>
Outline

• Introduction
• Similarity Layers
• General Alignment Process
• Application Scenarios
• Complexity & Evaluation
• Using Mappings

Framework for Similarity Computation

Ontology Similarity

\[
sim : \mathcal{P}(E) \times \mathcal{P}(E) \times O \times O \rightarrow [0, 1]
\]

Positive

\[\forall e, f \in \mathcal{P}(E), O_1, O_2 \in O, sim(e, f, O_1, O_2) \geq 0\]

Maximality

\[\forall e, f, g \in \mathcal{P}(E), O_1, O_2 \in O, sim(e, e, O_1, O_2) \geq sim(f, g, O_1, O_2)\]

Symmetry

\[\forall e, f \in \mathcal{P}(E), O_1, O_2 \in O, sim(e, f, O_1, O_2) = sim(f, e, O_2, O_1)\]

Ontology Similarity (2)

Two entity sets are identical

\[\forall e, f \in \mathcal{P}(E), O_1, O_2 \in O, sim(e, f, O_1, O_2) = 1 \iff e = f\]

Two entity sets are similar / different to a certain degree

\[\forall e, f \in \mathcal{P}(E), O_1, O_2 \in O, 0 < sim(e, f, O_1, O_2) < 1\]

Two entity sets are different and have no common characteristics

\[\forall e, f \in \mathcal{P}(E), O_1, O_2 \in O, sim(e, f, O_1, O_2) = 0 \iff e \neq f\]
Data Layer

Data types such as integers, strings etc. are compared by operations such as relative distance and edit distance.

Equal values:

\[ \text{sim}_{\text{equality}}(v_1, v_2) := \begin{cases} 1 & \text{if } v_1 = v_2, \\ 0 & \text{otherwise}. \end{cases} \]

String similarity:

\[ \text{sim}_{\text{string}}(v_1, v_2) := \max(0, \frac{\min(|v_1|, |v_2|) - \text{ed}(v_1, v_2)}{\min(|v_1|, |v_2|)}) \]

Relative distance:

\[ \text{sim}_{\text{diff}}(v_1, v_2) := 1 - \frac{|v_1 - v_2|}{\max\text{diff}} \]

Ontology Layer

Similarity between sets of entities:

- Single Linkage
  \[ \text{sim}_{\text{single}}(E, F) := \max_{(e, f) \in E \times F} \text{sim}(e, f) \]

- Average Linkage
  \[ \text{sim}_{\text{complete}}(E, F) := \frac{\sum_{(e, f) \in E \times F} \text{sim}(e, f)}{|E| \cdot |F|} \]

- Multi Similarity
  \[ \text{sim}_{\text{set}}(E, F) := \frac{\sum_{e \in E} \sum_{f \in F} \text{sim}(e, f)}{|E| \cdot |F|} \]

with \( e = (\text{sim}(e, e_1), \text{sim}(e, e_2), \ldots, \text{sim}(e, f_1), \text{sim}(e, f_2), \ldots) \), \( f \) analogously.

Object similarity:

- Object Equality
  \[ \text{sim}_{\text{object}}(e, f) := \begin{cases} 1 & \text{if } \text{align}(e) = f, \\ 0 & \text{otherwise}. \end{cases} \]

- Explicit Equality
  \[ \text{sim}_{\text{explicit}}(e, f) := \begin{cases} 1 & \text{if } \text{statement}(e, \text{"sameAs"}, f), \\ 0 & \text{otherwise}. \end{cases} \]

Similarity between sets of entities:

- Dice Coefficient
  \[ \text{sim}_{\text{dice}}(E, F) := \frac{2 \cdot |E \cap F|}{|E| + |F|} \]

- Jacquard Coefficient
  \[ \text{sim}_{\text{jacquard}}(E, F) := \frac{|E \cap F|}{|E| \cup |F|} \]

Label similarity:

\[ \text{sim}_{\text{label}}(e, f) := \text{sim}_{\text{string}}(\text{label}(e), \text{label}(f)) \]

Taxonomic Similarity for Concepts: Extensional

\[ \text{sim}_{\text{taxonomic}}(c_1, c_2) := \begin{cases} e^{-al} \cdot e^{\frac{\gamma h}{e^{\gamma h} + e^{-\gamma h}}} & \text{if } c_1 \neq c_2, \\ 1 & \text{otherwise}. \end{cases} \]
Ontology Layer

Extensional Concept Similarity:
\[ \text{sim}_{\text{extension}}(c_1, c_2) := \text{sim}_{\text{set}}(\iota C(c_1), \iota C(c_2)) \]

Domain and Range Similarity:
\[ \text{sim}_{\text{domran}}(r_1, r_2) := 0.5 \cdot ( \text{sim}_{\text{object}}(\text{ran}(r_1), \text{ran}(r_2)) + \text{sim}_{\text{object}}(\text{dom}(r_1), \text{dom}(r_2)) ) \]

Concept Similarity of Instances:
\[ \text{sim}_{\text{parent}}(i_1, i_2) := \text{sim}_{\text{object}}(c_1, c_2) \]
where \( i_1 \in \iota C(c_1), i_2 \in \iota C(c_2) \)

Outline

- Introduction
- Similarity Layers
- General Alignment Process
- Application Scenarios
- Complexity & Evaluation
- Using Mappings

Context Layer

Frequency of usage of an entity or its characteristics in a certain context
\[ \text{sim}_{\text{use}}(e, f) := \text{sim}_{\text{diff}}(\text{Usage}(e, \text{con}), \text{Usage}(f, \text{con})) \]

Example: Two books may be similar if their authors have many coauthored publications.

General Alignment Process

\[ \text{align} : E \times O \times O \rightarrow E, \]
\[ \forall e \in E_{O_1} \exists f \in E_{O_2}, O_1 \in O, O_2 \in O : \text{align}(e, O_1, O_2) = f \]
\[ \forall \text{align}(e, O_1, O_2) = \perp \]
General Alignment Process: Step 1

\[
\text{feat} : O \times O \rightarrow \mathfrak{P}(F)
\]

Determine a list of features F:
Extract characteristics of both ontologies, i.e. the features of their ontological entities (concepts C, relations R, and instances I) from intensional and extensional ontology definitions

Ontology Alignment Example: Step 1

The Car concept of ontology 1 is described by its label: Car, its superclass (subClassOf Vehicle), its concept sibling: boat, the direct property: hasSpeed, and its instance Porsche KA-123

The relation hasSpeed is described by the domain: Car and the range: Speed

The instance Porsche KA-123 is characterized by the instantiated property instance: belongsTo: Marc and property instance: hasSpeed: 300 km/h

Ontology Alignment Example

Possible features:
- Identifiers: i.e. strings with dedicated formats, such as unified resource identifiers (URIs) or RDF labels.
- RDF/S Primitives: such as properties or subclass relations
- OWL Primitives: such as an entity being the sameAs another entity
- Derived Features: which constrain or extend simple RDFS primitives (e.g. most-specific-class-of-instance)
- Aggregated Features: i.e. aggregating more than one simple RDFS primitive, e.g. a sibling is every instance-of the parent-concept of an instance
- Domain Specific Features for instance, in an application where files are represented as instances and the relation hashcode-of-file is defined, we use this feature to compare representations of concrete files
- Ontology External Features: Any kind of information not directly encoded in the ontology, such as a bag-of-words from a document describing an instance
Features and Similarity Measures

<table>
<thead>
<tr>
<th>Comparing</th>
<th>No.</th>
<th>Feature</th>
<th>Similarity Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concepts</td>
<td>1</td>
<td>label(X₁)</td>
<td>string(X₁, X₂)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>(identifier, X₁)</td>
<td>explicit(X₁, X₂)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>(X₁, sameAs, X₂) relation</td>
<td>object(X₁, X₂)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>(direct relations, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>all (inherited relations, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>all (superconcepts, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>all (subconcepts, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>(superconce, Y₁) / (subconce, Y₂)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>(concept siblings, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>(instances, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td>Relations</td>
<td>1</td>
<td>(identifier, X₁)</td>
<td>explicit(X₁, X₂)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>(X₁, sameAs, X₂) relation</td>
<td>object(X₁, X₂)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>(domain, X₁) and (range, X₁)</td>
<td>object(X₁, X₂)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>all (subrelations, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>all (subrelations, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>(relation siblings, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>(relation instances, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td>Instances</td>
<td>1</td>
<td>(identifier, X₁)</td>
<td>explicit(X₁, X₂)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>(X₁, sameAs, X₂) relation</td>
<td>object(X₁, X₂)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>all (parent-concepts, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>all (parent-concepts, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>(relation instances, Y₁)</td>
<td>set(Y₁, Y₂)</td>
</tr>
</tbody>
</table>

| Relations-Instances | 1 | (domain, D₁) and (range, R₁) | object(D₁, D₂, R₁, R₂) |
|                    | 2 | (parent relation, T₁)         | set(T₁, T₂)            |

Ontology Alignment Example: Step 2

Compare all entities of the same type:

55 candidate alignments:
- 42 concept pairs (6x7)
- 4 relation pairs (2x2)
- 9 instance pairs (3x3)

E.g., comparing
o₁:belongsTo with
o₂:hasProperty and with
o₂:hasMotor

General Alignment Process: Step 2

Most common methods:
- compare all entities of O₁ with all entities of O₂:
  \( e, f \in E₁ \times E₂ \)
  any pair is treated as a candidate mapping
- or only compare entities of the same type
  \( e, f \in (C₁ \times C₂) \cup (R₁ \times R₂) \cup (I₁ \times I₂) \)
- or use heuristics to lower the number of candidate mappings (e.g., applied in QOM) using strategies such as random or label, or change propagation

General Alignment Process: Step 3

Based on the features F of the ontological entities we do the similarity computation for all pairs of candidates

Additional similarity measures exist

An example is described on the next slide
The individual similarity measures are weighted and combined. Some approaches for aggregation:

Averaging:

\[ \text{adj}_k(x) = \text{id}(x) \]

Linear Summation:

\[ w_k \text{ learned or manually assigned, } \text{adj}_k(x) = \text{id}(x) \]

Linear Summation with negative evidence: \( w_k \) can have a negative value (e.g., superconcepts of the first entity have a high similarity with subconcepts of the second entity)

Sigmoid Function: emphasize high similarity and de-emphasize low similarity:

\[ \text{adj}_k(x) = \text{sig}_k(x - 0.5), \]
\[ \text{sig}_k(x) = \frac{1}{1 + e^{-a_k x}} \]

### General Alignment Process: Step 4

\[ \text{agg} : [0, 1]^k \rightarrow [0, 1] \]

The individual similarity measures are weighted and combined

\[ \text{sim}_{\text{agg}}(e, f) := \sum_{k=1, \ldots, n} w_k \cdot \text{adj}_k(\text{sim}_k(e, f)) \]

Some approaches for aggregation:

Averaging:

\[ w_k = 1, \text{adj}_k(x) = \text{id}(x) \]

### General Alignment Process: Step 5

\[ \text{inter} : [0, 1] \rightarrow \{\text{alignment}\} \]

Aggregated similarity is compared to a threshold:

every value above indicates an alignment

Determine the threshold:

- constant,
- \( \max(\text{sim}_{\text{agg}}(e, f)) - \text{constant} \),
- \( \max(\text{sim}_{\text{agg}}(e, f))(1 - p) \)

Computing the similarity of the candidate alignment: \( o_1:\text{Car} \) and \( o_2:\text{Automobile} \)

For every feature we compute a similarity. E.g.,

\[ \text{sim}_{\text{label}}(o_1:\text{Car}, o_2:\text{Automobile}) = \text{sim}_{\text{string}}(\text{Car}, \text{'Automobile'}) = 0.0 \]

\[ \text{sim}_{\text{superconcept}}(o_1:\text{Car}, o_2:\text{Automobile}) = \text{sim}_{\text{set}}(\{o_1:\text{Vehicle}\}, \{o_2:\text{Vehicle}\}) = 1.0 \]
General Alignment Process: Step 6

- Entities are similar if their position in the structure is similar
- Structure similarity is expressed through the similarity of the other entities in the structure
- Calculating the similarity for one entity pair, the similarity of the neighboring entity pairs are needed
- In a first round only basic comparison methods (e.g., string similarity) is applied (or pre-given alignments are used)
- In further rounds already computed pairs and use more sophisticated structural similarity measures
- When to stop the iteration:
  1. fixed number of iterations
  2. fixed time constraint
  3. changes of alignments compared to a threshold

Outline

- Introduction
- Similarity Layers
- General Alignment Process
- Application Scenarios
- Complexity & Evaluation
- Using Mappings

Ontology Merging

- Two or more ontologies are combined into one target ontology
- By establishing alignments among entities we identify equal entities which we can merge
- Time resources are less critical
- Human post-processing is required
- Finally high quality requirements

Web Service Composition

- Agents or web service often use different representations of their domains resulting in different expressions on their goals, and their input or output
- Collaborate despite the heterogeneous representations
- Standard upper-level ontologies or ontology alignment
- Alignment needs to be fast, reliable, and correct
- Wrong results can lead to unjustified costs
- Sometimes user checks are possible
- Example: combine a booking service of an air carrier and a hotel reservation network
Query and Answer Mapping

• Users formulate a query in a specific query language based on a specific ontology
• Query is sent to a query engine
• To access heterogeneous information sources the query needs to be re-written for the target ontology(s)
• For the presentation of the answers the results have to be transformed back again

• Rewriting / Mapping should be fast and fully automatic
• Users may tolerate wrong results as long as the correct results are returned as well

Reasoning

• New information is inferred from distributed and heterogeneous ontologies
• Time constraints are not critical (for both, alignment and inference tools)
• Quality of the alignments is very important
• Alignment needs to be done automatically
• Wrong results may trigger additional wrong results in a cascading manner
• Detection of conflicting inconsistencies is required
• Many unsolved research issues

Evaluation: Recall & Precision

Standard information retrieval metrics

Precision \( p = \frac{\#correct\_found\_mapping}{\#found\_mappings} \)
Recall \( r = \frac{\#correct\_found\_mappings}{\#existing\_mappings} \)
F-Measure \( f_1 = \frac{2pr}{p+r} \)
Evaluation

• Compliance measure
  quality of identified alignments

• Performance measure
  quality of algorithm in terms of computational resources

• User-related measure
  overall subjective user satisfaction, e.g., how much user effort is needed

• Task-related measure
  quality of alignment for a certain use case or application scenario

[ML]

Video Satisficing Ontology Mapping

[ML]

Yearly Contest

Steffen Staab Video Slide - Reduction of Comparisons

[ML]
• $c = (\text{feat} + \text{select} + \text{comp} \cdot (\Sigma_k \text{sim}_k + \text{agg}) + \text{inter}) \cdot \text{iter}$

• NOM
  $c = O((n + n^2 + n^2 \cdot (\log^2(n) + 1) + n) \cdot 1)$
  $= O(n^2 \cdot \log^2(n))$

• PROMPT
  $c = O((n + n^2 + n^2 \cdot (1 + 0) + n) \cdot 1)$
  $= O(n^2)$

• QOM
  $c = O((n + n \cdot \log(n) + n \cdot (1 + 1) + n) \cdot 1)$
  $= O(n \cdot \log(n))$
Outline

- Introduction
- Similarity Layers
- General Alignment Process
- Application Scenarios
- Complexity & Evaluation
- Using Mappings

Video Ontology Mapping and Alignment

Using Mappings

- Data transformation
- Query answering
- Reasoning with mappings
  - mapping composition (covered earlier)
- Generation of ontology extensions
# Data Transformation

- **Mapping Interpreter (Stanford SMI)**
  - Uses an instantiated ontology of mappings
    - mapping structure
    - Python rules

- **OntoMerge**
  - Treats source ontologies with data and mapping axioms as a single ontology
  - Uses a theorem prover to create new data

---

# Query Answering

- **Two settings**
  - one-to-one mappings
  - global ontology

---

# Query Answering (II)

- **Piazza (UW)**
  - Peer-to-peer architecture for query answering
  - Query reformulation using mappings between adjacent peers
Query Answering

- **OIS (Calvanese, et. al.)**
  - Global ontology mapped to local ontologies
  - Mappings defined as views
  - Using a Description Logic Reasoner to answer queries

**References**


