Automating Ontology Mapping in IT Service Management: A DOLCE and ITSMO Integration

RESEARCH PAPER

ANDREY KHALOV
OLGA ATAEVA
*Author affiliations can be found in the back matter of this article

ABSTRACT

Background: Ontologies and knowledge graphs have become critical for structuring data into machine-interpretable knowledge, especially in dynamic domains like IT service management (ITSM). Traditional ontology engineering relies heavily on domain experts, making it costly and slow. This study investigates whether a domain-specific ontology can be extended from a top-level ontology without expert involvement, using the IT service management ontology (ITSMO) and the descriptive ontology for linguistic and cognitive engineering (DOLCE-lite) as a test case used in this study.

Methodology: We propose an automated mapping approach integrating lexical approaches, embeddings, graph neural networks (GNN), and large language models (LLMs). Two primary mapping methods were developed: (1) embedding-based matching, computing cosine similarity between class embeddings from DOLCE and ITSMO; and (2) LLM-based matching, prompting a language model (GPT-4o) to evaluate class compatibility on a numeric scale. We also experiment with *GraphSAGE GNN* to enrich embeddings with ontology structure. *Z-score clustering* is applied to similarity scores to select top candidate mappings while filtering out outliers from the top cluster. The methodology operates with no annotated data and was validated using three-steps approach: GPT-4o as a surrogate expert for baseline class matching evaluation, expert spot-check, and OWL reasoner (Pellet and HermiT) to prove logical consistency (Glimm *et al.*, 2014; Sirin *et al.*, 2007).

Results: The automated method successfully mapped ITSMO classes under DOLCE, yielding an integrated ontology (80 classes) that extends DOLCE into the ITIL domain with minimal expert intervention (expert consolidated suggestions into a result ontology). The LLM-based approach (GPT-4o) achieved the best performance with 73.5% accuracy for top-1 mappings and 82.4% for top-3 (cluster) inclusion. Transformer-based embeddings (e.g., DeBERTa) also performed well (up to 39.3% top-1, outperform random matching with 27.6% accuracy), but classical graph embeddings (RDF2Vec/Node2Vec) failed due to the small ontology size. Incorporating a GNN provided smoother embedding distributions and increased correct mappings within top-3 clusters, but it slightly reduced top-1 precision in this small-graph setting. These findings underscore the effectiveness of LLMs in zero-shot ontology alignment and the limitations of purely structural methods on limited data.

CORRESPONDING AUTHOR:
Andrey Khalov
Moscow Institute of Physics and Technology, National Research University, Moscow, Russia
khalov.a@phystech.edu

KEYWORDS:
ontology mapping; knowledge graph; DOLCE; embeddings; large language models

TO CITE THIS ARTICLE:
Khalov, A. and Ataeva, O. 2025 Automating Ontology Mapping in IT Service Management: A DOLCE and ITSMO Integration. *Data Science Journal*, 24: 23, pp. 1–19. DOI: https://doi.org/10.5334/dsj-2025-023

Conclusions: This work demonstrates, as a proof-of-concept, that an upper-level ontology can be extended to a domain ontology automatically, with no or minimal expert involvement, by leveraging AI-based mapping techniques. The resulting new ontology integrates ITSMO into DOLCE, providing a consistent semantic foundation for IT domain knowledge graphs. The approach is immediately applicable to ITSM and suggests a generalizable framework for ontology expansion in other domains. Future work will focus on scaling the method to larger ontologies, automatically discovering new classes/relations from text, and evaluating the approach's practical impact on IT service management processes.

INTRODUCTION

The rapid growth of enterprise data has created a pressing need for effective knowledge management techniques and data utilization pipelines. Converting unstructured data into structured knowledge is key to building intelligent IT systems. Semantic technologies—notably ontologies and knowledge graphs—enable this transformation by providing a formal representation of domain knowledge, which can enhance reasoning and improve the performance of large language models (Lewis *et al.*, 2020).

Building comprehensive knowledge graphs for a specific domain traditionally requires significant expert involvement. Domain experts must pre-define classes, relationships, and rules (ontology engineering process), which is traditionally time-consuming and expensive (Fernández-López and Gómez-Pérez, 2002; Noy and McGuinness, 2001). On the other hand, a data-driven approach builds the knowledge graph incrementally as new data arrives: classes and relations are defined alongside ingestion. This reduces expert involvement but does not guarantee global consistency (see Figure 1).

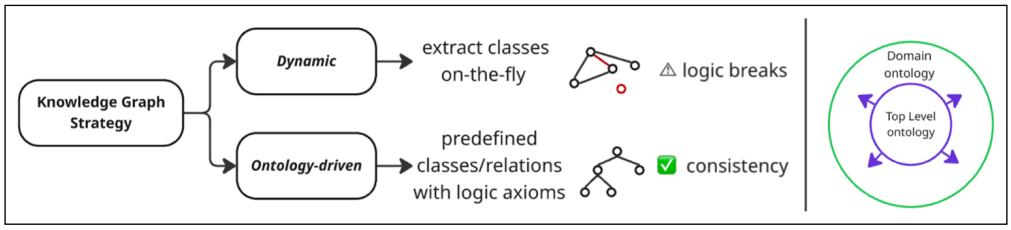

Figure 1 Two approaches to building knowledge graphs and TLO ontology expand to specific domain.

Many domain ontologies already exist (e.g., in bioinformatics: disease ontology, protein ontology, etc.), and there are several widely used top-level ontologies (TLOs) like SUMO, DOLCE, and BFO that provide universal concepts. The task of ontology mapping or ontology alignment arises when linking a TLO with a domain ontology to create a unified knowledge model. Prior studies have shown high success rates in aligning large ontologies (with >1000 classes) when ample training data or manual mappings are available (Euzenat and Shvaiko, 2013; Jiménez-Ruiz *et al.*, 2022). However, mapping in low-data scenarios (few classes, no annotated correspondences) remains challenging and underexplored.

Our study focuses on extending the DOLCE-lite ontology into the IT domain by integrating the IT service management ontology (ITSMO), which is based on ITIL methodology (Borgo *et al.*, 2023; El Yamami *et al.*, 2019). The main hypothesis is that an ontology can be initially created and extended without domain experts by leveraging semantic similarity techniques and AI models. Specifically, we aim to map super classes of the ITSMO to an appropriate leaf-classes in the DOLCE, effectively seeding a new merged ontology that covers both general and domain-specific concepts. If successful, this approach would suggest that ontology engineering efforts can be significantly accelerated with automated methods, reserving human expertise for validation rather than initial development.

To tackle this, we propose an integrated methodology combining embedding-based similarity, graph neural networks (GNNs), and large language model (LLM) prompting. This method does not require any pre-existing mappings or training labels—an important advantage since no annotated dataset is available for DOLCE and ITSMO. The contributions of this work are: (1) a novel LLM-powered mapping approach that treats class alignment as a contextual similarity problem solvable via prompt engineering; (2) incorporation of graph structure through GNN-based embedding enrichment to capture ontological context; (3) an evaluation of multiple

approaches (lexical, embedding, GNN, and LLM) on ontology mapping task, highlighting their strengths and limitations; and (4) the construction of domain specific new ontology, which demonstrates the feasibility of building ontologies with minimal expert involvement. We also discuss how this ontology can be expanded further and its potential value in real-world scenarios.

The remainder of the paper is structured as follows: Section 'Related Work and Background' reviews related work on ontology mapping and the use of DOLCE, ITSMO, and ITIL in semantic systems. Section 'Methodology' details the proposed methodology, including data preparation, embedding generation, LLM prompting, and the mapping algorithm with *Z*-score-based filtering. Section 'Results' presents the results of our experiments, comparing mapping accuracy across methods, and provides a combined results and discussion of the findings. Section 'Discussion' offers a deeper discussion on the practical implications, scalability, and limitations of our approach, and Section 'Conclusion' concludes the paper with final remarks and future research directions.

RELATED WORK AND BACKGROUND

TOP-LEVEL AND DOMAIN ONTOLOGIES IN IT

Top-level ontologies: TLOs such as *SUMO*, *BFO*, and *DOLCE* provide abstract frameworks for representing reality, which can be specialized to any domain (Borgo *et al.*, 2023; Smith, 2008). SUMO offers a broad formal ontology; basic formal ontology (BFO) focuses on fundamental categories for scientific domains; and descriptive ontology for linguistic and cognitive engineering (DOLCE) emphasizes cognitive distinctions and has been applied across various fields, including IT. Notably, DOLCE's design makes it flexible for conceptual modeling, which is a reason for its adoption in our work. In the IT context, both BFO and DOLCE could, in principle, represent ITIL concepts (such as configuration and incident), but DOLCE's cognitively oriented categories are better suited for describing processes and artifacts in IT service management. Indeed, ITIL's core concepts (like Configuration Item and Incident) structurally map to key classes in upper ontologies (e.g., endurant/perdurant in DOLCE or continuant/occurrents in BFO). This study uses the DOLCE-lite version as the foundational ontology (TLO) to be extended into the IT domain (DOLCE overview, ISTC-CNR).

Domain ontologies for ITSM: Several ontologies have been developed for IT and related domains. Examples include the software ontology (SWO) for software tools, common core ontologies for general concepts, and specifically the IT service management ontology (ITSMO, w3id) (El Yamami *et al.*, 2019; Pastuszak, Czarnecki and Orlowski, 2012). ITSMO is designed to model IT service management processes and entities, aligning with the ITIL framework. It defines classes such as Incident, Service, Configuration Item, etc., and is intended to work alongside other standard vocabularies (e.g., GoodRelations, Dublin Core, and FOAF) for interoperability. Because ITSMO already encapsulates many ITIL concepts (like Incident, Configuration Item, Process, Service), it provides an excellent domain-specific ontology to integrate with a TLO. By mapping ITSMO to DOLCE, we aim to create a new ontology that contains the high-level semantic structure from DOLCE plus the domain specifics of ITSMO. This approach follows recommendations in the literature emphasizing the use of upper ontologies to standardize and ease interoperability of domain models.

ITIL framework: Information Technology Infrastructure Library (ITIL) is a set of best practices for IT service management, encompassing processes like incident management, configuration management, service delivery, etc. (El Yamami *et al.*, 2019; Pastuszak, Czarnecki and Orlowski, 2012). ITIL provides an industry-defined structure of how IT services are managed and improved. Representing ITIL in an ontological form helps formalize concepts and their relationships, enabling automated reasoning and integration with other knowledge resources. Prior works have created ontologies or models for subsets of ITIL processes—for example, an ontological model for ITIL's service level management process and rule-based models to implement ITIL processes. These efforts show the benefit of ontologies for ensuring consistency and clarity in ITIL implementations. However, they typically address specific processes or require manual modeling. By using ITSMO (which covers ITIL concepts) and aligning it with DOLCE, our work builds on these efforts to create a more comprehensive ITIL-based ontology without manual schema creation.

ONTOLOGY MAPPING APPROACHES

Ontology mapping (or alignment) is the task of identifying correspondences between semantically related entities (classes and properties) in different ontologies. A classic review by Choi *et al.* (2006) categorizes ontology mapping outcomes into three types: (1) pairs of related classes identified (simple mappings), (2) a mediating 'bridge' ontology that connects the two (partial reference alignment), and (3) a merged ontology combining both original ontologies. Our goal corresponds to outcome of third type—creating a unified ontology by merging DOLCE and ITSMO classes via subclass relationships (Figure 2). In this process, each ITSMO class will be linked as a subclass of a DOLCE class. If an ITSMO class has no suitable parent in DOLCE (no mapping found), it will default to being a subclass of the root class *Thing*. This mapping model ensures that domain-specific classes attach to the upper ontology hierarchy, a strategy similar to those proposed in prior alignment studies.

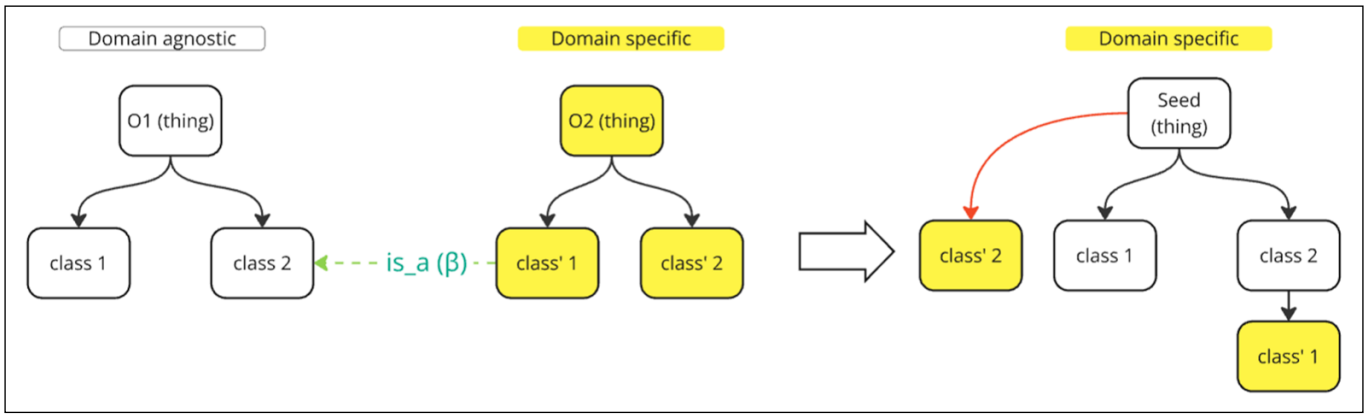

Figure 2 Identifying the most similar classes between two ontologies and linking them with an *is_a* relationship. ITSMO classes are always mapped as descendants of DOLCE classes; if an ITSMO class's parent has no DOLCE counterpart, the parent defaults to Thing. A pair of classes (*c*,*d*) becomes a candidate link when:

$$\beta \in (0,1),\ match(c,d) \Leftrightarrow cos(e(c),e(d)) \geq \beta,$$

Where $e(\cdot)$ is the class embedding and β is tunable threshold (hyperparameter). A similar mapping strategy discussed in Li *et al.* (2023) and Jiménez-Ruiz *et al.* (2022).

Several methodologies for ontology mapping have been explored in the literature:

- *Lexical matching:* Early and still-common methods rely on string similarity of class labels or descriptions. Tools like *AgreementMakerLight (AML)* and *LogMap* use lexical and simple semantic cues to match classes (Faria *et al.*, 2013; Jiménez-Ruiz and Cuenca Grau, 2011). For example, Noy and Musen's PROMPT framework supports semi-automatic merging using lexical hints and user input. However, lexical methods often fail when two ontologies use different terminologies or when class names/descriptions do not obviously match. In our case with different domains (IT vs. foundational concepts), we share a few lexical similarities. Indeed, using AML and LogMap off-the-shelf produced no valid mappings between DOLCE and ITSMO, aside from trivial exact name matches. This underscores the limitation of purely lexical approaches in our scenario.
- *Graph-based structural matching:* Some systems compare the graph structures of ontologies, attempting to align classes by the similarity of their neighborhoods (relationships to other classes). Examples include Falcon-AO and COMA (Noy and Musen, 2001), which use structural indices and matching of class hierarchies (Choi, Song and Han, 2006). Euzenat and Shvaiko (2013) provide a comprehensive overview of such structural methods. Structural approaches can capture context beyond just names by leveraging the ontology's taxonomy and properties. However, if the two ontologies' structures differ significantly, these methods may struggle to find alignments that are semantically meaningful. In our case, the DOLCE taxonomy is organized by philosophical categories (e.g., Endurant, Perdurant, and Abstract), whereas ITSMO is organized by ITIL concept categories (e.g., Service, Configuration, and Process). A pure structural match might not align these correctly because their hierarchies reflect different organizing principles.

- *Embedding-based methods:* Recent research has turned to distributed representations (embeddings) of ontology entities. Techniques like *RDF2Vec* and *OWL2Vec** learn vector embeddings for classes by walking the graph structure of ontologies and using neural language models to capture contextual semantics (Chen *et al*., 2021; Ristoski and Paulheim, 2016). Such embeddings can encode both lexical information and graph structure in a continuous vector space. Cosine similarity between class embeddings can then suggest mappings (Karadeniz and Ozgur, 2019). Embedding methods have shown high accuracy in discovering relations or class alignments in large ontologies and knowledge graphs. However, they usually require a substantial corpus or graph to train on either a large ontology or external data. In our study, both DOLCE-lite and ITSMO have fewer than 100 classes each, which is extremely small for training embedding models from scratch. Pre-trained models (like those from general text or large knowledge graphs) might not capture the nuances of these ontologies. We included RDF2Vec and a similar Node2Vec approach in our experiments for comparison, but as expected, their performance was poor given the data scarcity. These results align with the notion that embedding models need large training sets (on the order of millions of triples) to be effective.
- *Deep learning and transformer models:* Building on embeddings, researchers have applied deep neural networks and transformers to ontology alignment. *BERTMap* (He *et al*., 2022) uses BERT to generate contextual embeddings of class descriptions and aligns ontologies via a fine-tuned transformer model. These approaches often achieve higher precision by learning from labeled mappings, but they require training data (aligned ontologies or at least positive/negative class pairs) to fine-tune the model. In zero-shot settings (no training pairs), one can still use transformer language models to encode class semantics. For example, He *et al*. (2023) and Qiang *et al*. (2024) explored using large transformers to adapt ontologies to new domains, showing that LLMs can capture domain nuances if guided properly. The downside is that large models are resource-intensive and sometimes prone to errors without fine-tuning (e.g., they might 'hallucinate' incorrect mappings if prompted naively). Our work employs transformer models in two ways: (a) using pre-trained sentence embedding models (like SBERT or similar) to encode class definitions, and (b) using GPT-4 directly as a predictor to judge class similarity via prompting. We also fine-tuned a pretrained large model (DeBERTa) on an ITIL-related text corpus to see if domain adaptation improves alignment (He *et al*., 2021; He *et al*., 2023).
- *Large language model (LLM) prompting:* The emergence of powerful LLMs (GPT-3.5, GPT-4o, etc.) provides a new paradigm: instead of training a specific alignment model, we can *ask* an LLM to perform the mapping task (Qiang *et al*., 2024). Recent work has attempted to use GPT-family models to generate or validate ontology alignments. LLMs can leverage both the semantic meaning of class names/descriptions and general world knowledge. A challenge is that LLM outputs are not deterministic and can vary each run. They also are not guaranteed to understand the ontologies' strict logical constraints. However, when ontologies are small, one advantage is that we can provide *all class information as prompt context*. In our case, the combined class descriptions of DOLCE-lite and ITSMO (a total of ~80 classes) fit within GPT-4o's input limit. This enabled us to prompt GPT-4o with the entire list of classes and ask it to rate the similarity of each DOLCE class to a given ITSMO class. By repeating such prompts and aggregating results, we aimed to reduce random variance from the LLM. Our approach is in line with Amini *et al*. (2023), who suggest zero-shot LLM-based alignment, and it extends it by introducing multiple prompt trials and statistical filtering for stability.

In summary, existing approaches each have limitations in our scenario: lexical and structural methods falter due to cross-domain differences and sparse data, embedding methods suffer from the small ontology size, and deep learning models need training data which we lack. LLM prompting emerges as a promising approach for zero-shot ontology alignment, albeit one that must be carefully controlled for consistency. In this study, we exclude purely manual or lexical methods and focus on embedding-based, GNN-augmented, and LLM-based strategies, adapting them to work with minimal data and no human supervision.

METHODOLOGY

The key steps in our approach are: preparing class descriptions, generating similarity scores using two different methods (embeddings vs. LLM), and filtering these scores to choose final mappings. We also incorporate a Graph Neural Network step to enhance embeddings with structural context from the ontology graphs. Below, we detail each component of the methodology.

ONTOLOGY DATA PREPARATION

Ontology descriptions: We use DOLCE-Lite (OWL; https://www.loa.istc.cnr.it/ontologies/DOLCE-Lite.owl), and ITSMO via the canonical resolver https://w3id.org/itsmo (see References for persistent sources), each containing class definitions, relationships, and axioms, and can be represented as string values in natural language with RDF syntax on top of strings. To focus on class mapping, we extracted the *label and textual definition of each class* (e.g., rdfs:label, rdfs:comment in OWL) as its description. Table 1 summarizes key metrics of the two ontologies:

Table 1 Key metrics of DOLCE-lite and ITSMO ontologies.

	DOLCE-LITE	ITSMO
Axiom count	534	584
Logical axiom count	349	228
Declaration axioms	107	109
Class count	37	43
Object property count	70	41

As seen, DOLCE-lite (in the version we used) has 37 classes, while ITSMO defines 43 classes (*Note*: some classes in ITSMO are high-level groupings of ITIL entities, whereas DOLCE's classes partition into abstract categories.). The moderate number of object properties (relations) in each indicates that both ontologies encode rich relationships, though our mapping will consider only class-level alignment in this phase.

We treat each class's *textual description* as its primary semantic content. For DOLCE, these descriptions are philosophical, and for ITSMO, they are ITIL-specific:

OWL class description from DOLCE:

```
<!-- http://ontology.it/itsmo/v1#DataCenter -->
<owl:Class rdf:about="http://ontology.it/itsmo/v1#DataCenter">
<rdfs:subClassOf rdf:resource="http://ontology.it/itsmo/v1#Building"/>
<rdfs:comment xml:lang="en">is a facility used to house computer
systems and associated components&gt;</rdfs:comment>
<rdfs:label>DataCenter</rdfs:label>
</owl:Class>
```

OWL class description from ITSMO:

```
<!-- http://www.loa-cnr.it/ontologies/DOLCE-Lite.owl#event -->
<owl:Class rdf:about="http://www.loa-cnr.it/ontologies/DOLCE-Lite.
owl#event">
<rdfs:subClassOf rdf:resource="http://www.loa-cnr.it/ontologies/DOLCE-
Lite.owl#perdurant"/>
<rdfs:comment>An occurrence-type is stative or eventive according to
whether it holds of the mereological sum of two of its instances, i.e.
if it is cumulative or not. A sitting occurrence is stative since
the sum of two sittings is still a sitting occurrence. In general,
events differ from situations because they are not assumed to have a
description from which they depend...</rdfs:comment>
</owl:Class>
```

These texts are the input to our embedding models and also provided to the LLM in prompts. One important assumption we make is that all relevant information about a class is contained in its description and label, that is, we do not utilize the ontologies' relational structure beyond basic subclass hierarchies. We ignore complex axioms or property restrictions in this mapping step, both to simplify the task and because such details might not align directly between a TLO and a domain ontology.

SIMILARITY COMPUTATION APPROACHES

We employ two primary approaches for computing similarity between classes of DOLCE and ITSMO:

A. *Embedding cosine similarity:* Each class description is converted into a vector embedding in a semantic space, and then cosine similarity is calculated between vectors of a DOLCE class and an ITSMO class. High cosine similarity indicates that the two classes have similar contextual meaning. Figure 3 illustrates this process. We experimented with multiple embedding generation methods (detailed in Section 'Embedding Generation Methods'), including graph embeddings and pre-trained language model embeddings.

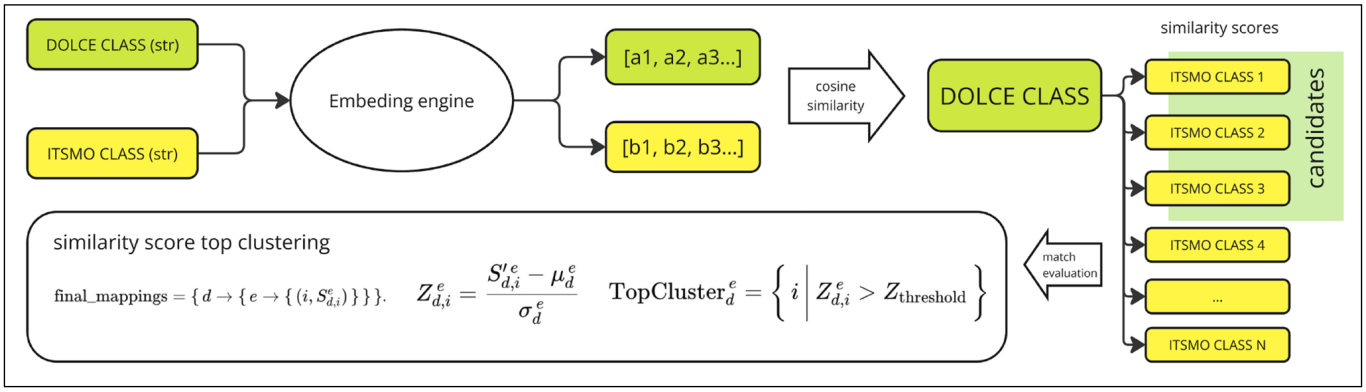

Figure 3 Textual descriptions of classes are transformed into embeddings. Cosine similarities between a DOLCE class embedding and all ITSMO class embeddings are computed. The top three candidates with highest cosine similarity for that DOLCE class form the initial *match cluster*. We then apply a statistical filter (*Z*-score) to decide which of these top candidates are significantly above the others in similarity. If fewer than three candidates survive filtering, the remaining slots are marked as 'No match' (indicating the DOLCE class may not have a clear corresponding ITSMO subclass).

B. *LLM-based compatibility scoring:* To exploit the global context of both ontologies, we prompt GPT-4o with all class descriptions from DOLCE-lite and ITSMO in a single request. The system prompt supplies short encyclopedia-style definitions of the two ontologies, while the user prompt concatenates JSON-ready records for every class (label, rdfs:comment, super-/sub-class links stripped of embedding fields).

The model is instructed to return a JSON object where each DOLCE class key maps to (see Figure 4):

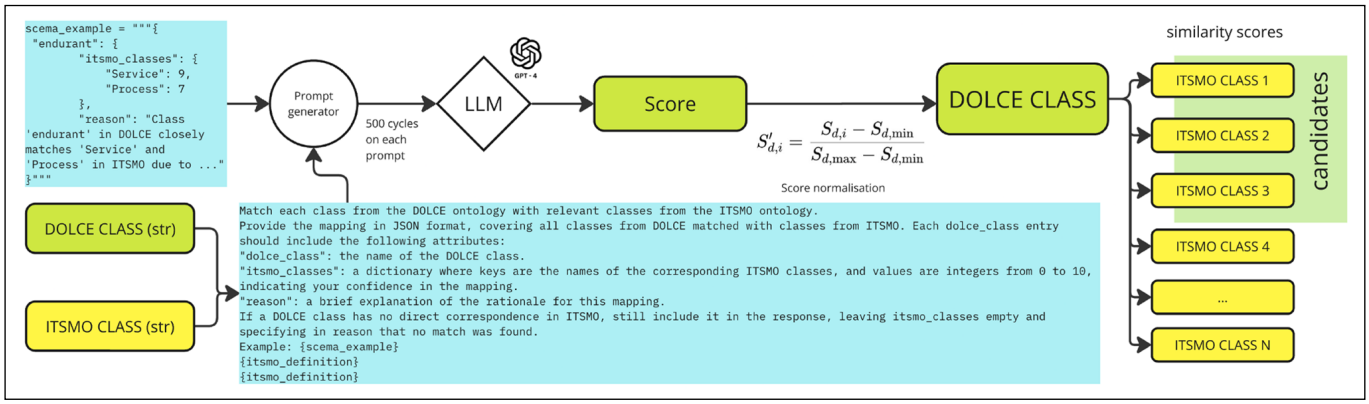

Figure 4 All class descriptions (or a subset) are provided to an LLM in a prompt, and the LLM scores the compatibility of classes. We repeated the process many times (500 trials for each DOLCE class with different sampling noise) to get stable estimates. The three ITSMO classes with the highest average scores for a given DOLCE class are taken as the top cluster, and then similarly a z-score filter is applied to those scores.

- An itsmo_classes dictionary whose keys are candidate ITSMO classes and whose values are integer confidence scores 0–10, and
- A brief reason string justifying the mapping.

JSON output example:

```
{
  "endurant": {
  "itsmo_classes": { "Service": 9, "Process": 7 },
  "reason": "Endurants are persistent entities that align with ITSMO
   'Service'..."

},
…
}
```

Because the entire cross-product is scored at once, the output already contains a similarity matrix between the 37 DOLCE and 43 ITSMO classes. We repeat the prompt 500 times with temperature = 0.8 to sample the model's epistemic uncertainty. Each run is parsed to valid JSON; confidence scores are accumulated and averaged, producing a stable matrix $S(D_i, I_j) \in [0,10]$.

For every DOLCE class D_i we then:

1. 'rank ITSMO classes by mean score';
2. 'keep the top-k = 3 entries whose mean exceeds a Z-score threshold (see Section 'Candidate Filtering via Z-Score Clustering')';
3. if no score survives the threshold, mark 'no match'.

This full-context prompting strategy eliminates the need for thousands of pairwise calls and lets GPT-4o reason over relations among all classes simultaneously, similar to recent agent-based alignment frameworks (Qiang, Wang and Taylor, 2024).

For each DOLCE class, both methods produce a ranked list of ITSMO classes by similarity. In approach (A), the similarity metric is cosine similarity; in (B), the metric is the average LLM score. We use a one-to-many matching strategy, meaning we consider all ITSMO classes as potential matches for each DOLCE class. Our aim is to find, for each DOLCE class, up to three top candidates in ITSMO that could be its subclasses.

Important assumption: We assume no external knowledge beyond the class descriptions. For instance, the LLM is not asked general ITIL questions; it strictly sees the ontology content. This focuses the task on ontology alignment rather than relying on the LLM's world knowledge (though inevitably some general knowledge might influence GPT-4o's understanding of terms).

EMBEDDING GENERATION METHODS

We tested several methods to generate class embeddings for the cosine similarity approach:

RDF2Vec: We applied the RDF2Vec algorithm to each ontology, treating the ontology as a knowledge graph. Random walks were performed to generate sequences, which were then used to train a Word2Vec model, resulting in 512-dimensional embeddings for each class. However, due to the small graph size, the utility of these embeddings is questionable—indeed, as we will see, they yielded almost no correct mappings.

Node2Vec: Similar to RDF2Vec, but relying purely on graph structure (ignoring literal attributes). We ran Node2Vec on each ontology's class graph (considering subclass relations and any other relationships between classes). Again, 512-d vectors were obtained. Like RDF2Vec, Node2Vec is hindered by the limited graph (under 50 nodes), so these serve more as a baseline.

Pre-trained language model embeddings: We utilized pre-trained transformer models to produce embeddings from the class description text. Specifically, *BERT-base* and *DeBERTa-v3-large* models were used to encode each class's textual description into a fixed-size vector. These models were not fine-tuned on our data (initially), so the embeddings reflect the general semantic similarity of the descriptions.

Domain-specific fine-tuning and augmentation: We created a fine-tuned variant of DeBERTa-v3-large, which we call *deberta_v3_large_ito*, using domain literature. We gathered a corpus of 5780 text fragments (each 256 tokens) related to ITIL and ontologies—including official ITIL documentation, prior DOLCE publications, and research papers with keywords 'ITIL' and 'ontology'. We then continued the pre-training of DeBERTa on this corpus (updating only the last four layers) for 100 epochs. To increase the effective size of this fine-tuning data, we employed a *data augmentation* strategy: we performed LLM-based paraphrase augmentation using OpenAI's gpt-4o-mini model (text generation only), which we used solely to produce paraphrases/summaries/terminology variants of the original fragments. This expanded the corpus to 17,337 samples and *significantly improved the language model's perplexity on the domain text (from 36.7 down to 4.35)*. The resulting fine-tuned model should, in theory, produce embeddings that are more sensitive to ITIL and DOLCE terminology. These embeddings (768-d for DeBERTa) were used as another input to the alignment. OpenAI gpt-4o-mini, used here as a paraphraser for data augmentation; see the official model documentation for capabilities and API usage (OpenAI, 2024).

OpenAI text embedding model: As an additional embedding source, we used the *text-embedding-ada-002* (referred to as *text-embedding-3-small* in our notes)—a pre-trained embedding model known for good performance on semantic similarity. This produced 1536-d embeddings for each class description. We included this because it may capture semantic nuances that our fine-tuned or other models miss, and it can serve as a check against which model's embeddings yield the best results.

Each embedding set is evaluated independently in the mapping step. We will later compare which embedding type led to better alignment accuracy (Section 'Results').

Note on small ontology challenges: Both RDF2Vec and Node2Vec tend to underperform on small ontologies (<100 classes) when trained without external corpora or literals; with enrichment (e.g., text literals and background graphs), they may yield meaningful signals, but in our setting they lagged behind transformer-based text embeddings. The transformer-based embeddings have the advantage of being pre-trained on large corpora, so they bring in general semantic understanding (e.g., knowing that 'hardware' and 'server' are related, or that 'incident' implies an event). Fine-tuning on ITIL-specific text aimed to inject domain-specific context (like understanding 'change request' in ITIL terms). Nonetheless, fine-tuning a model as large as DeBERTa on only ~17k short texts is borderline in terms of data adequacy—we observed only slight improvements, and indeed our results will show that the fine-tuned model did not dramatically outperform its base version (see following results).

GRAPH NEURAL NETWORK FOR EMBEDDING ENRICHMENT

To incorporate the graph structure of the ontologies into our semantic representations, we applied a Graph Neural Network (GNN) model that refines the class embeddings by considering their neighborhood in the ontology graph. The intuition is that classes are not isolated: for example, *Incident* in ITSMO might be linked to *Service* or *Impact* classes; capturing these connections could help align *Incident* to a DOLCE class like *Perdurant (Event)*, which also might connect to *Process* or *State* in DOLCE.

We used a two-layer GraphSAGE GNN with mean aggregation (Hamilton, Ying and Leskovec, 2017). The process is as follows:

Initialization: Each ontology's classes are initially assigned the embeddings computed by one of the methods in Section 3.3 (e.g., BERT or Node2Vec). These serve as initial node features.

Message passing (Layer 1): For each class node, we take the vector representations of its neighbor nodes (in the ontology graph) and the node itself. Neighbors include directly connected classes; in our case, we considered the *subclass hierarchy* and any equivalent class links (though DOLCE and ITSMO do not share equivalences prior to mapping). We then compute the *mean* of the neighbor vectors (this is the aggregation). This aggregated vector represents the local context of the node. We

concatenate the node's own vector with the neighbor aggregate and pass it through a linear transformation (with weight matrix W) and a non-linear activation to produce an updated embedding for the node.

Neighbor aggregation: For each node v, the embeddings of neighboring nodes $N(v)$ and the node v itself are aggregated.

$$h^{(1)}(v) = Aggregate(\{x_u : u \in N(v) \cup \{v\}\})$$

Aggregation function: The aggregation function used is the mean:

$$h^{(1)}(v) = \frac{1}{|N(v)|+1} \sum_{u \in N(v) \cup \{v\}} x_u$$

Embedding update: The aggregated embedding is transformed using a trainable weight matrix W:

$$z^1(v) = \sigma\left(W^1 \cdot h^1(v)\right)$$

Message passing (Layer 2): We repeat a similar operation, now each node gathers the embeddings of neighbors updated by Layer 1, computes the mean, and applies the second layer transformation.

For the second layer, the process is repeated, but the input is the embedding $z^1(v)$ from the first layer:

$$h^{(2)}(v) = \frac{1}{|N(v)|+1} \sum_{u \in N(v) \cup \{v\}} z_u^{(1)}$$

$$z^{(2)}(v) = \sigma\left(W^{(2)} \cdot h^{(2)}(v)\right)$$

Output projection: After two GNN layers, we add a final linear layer that projects the node embeddings back to the original embedding size (e.g., 768-d if BERT was used). This ensures the enriched embeddings reside in the same vector space as the originals, making them comparable to the original embedding-based similarity scores.

$$h_v = W^{(proj)} \cdot z^{(2)}(v)$$

Training objective: We train the GNN in an *unsupervised* manner, using a *reconstruction loss*. Specifically, we aim to make the GNN's output embeddings as close as possible to the original input embeddings for each node. The rationale for this is to *preserve the original semantic information* (which came from text or pre-trained models) while *injecting structural context* by forcing the GNN to account for neighbors. We used the Adam optimizer and employed early stopping if the loss did not improve, to prevent overfitting.

The model is trained on a reconstruction task, minimizing the mean squared error (MSE) between the output embeddings h_v and the original embeddings x_v:

$$\mathcal{L}(\theta) = \frac{1}{|V|} \sum_{v \in V} ||h_v - x_v||^2 = \frac{1}{|V|} \sum_{v \in V} \sum_{i=1}^{d} (h_{v,i} - x_{v,1})$$

where $h_{v,i}$ and $x_{v,i}$ are the ith components of the output and input embeddings for node v, respectively.

Hypothesis: We hypothesized that enriching embeddings with GNN would improve mapping results. The GNN could, for example, adjust embeddings so that classes that are siblings or share a parent in the ontology get more similar vectors, which might help correctly cluster them. It also might 'smooth' noisy embeddings—if one class's text embedding was off-target, the GNN will pull it closer to its neighbors' vectors.

We trained separate GNN models for each embedding type on each ontology (DOLCE and ITSMO graphs considered independently). The resulting enriched embeddings (denoted as GNN_node2vec, GNN_bert_base, etc.) were then fed into the similarity and mapping pipeline

just like the original embeddings. Figure 5 depicts the GNN enrichment process, showed that the GNN training converged, with different initial embeddings leading to different loss levels. Notably, when initialized with the text-embedding-3-small vectors, the GNN achieved the lowest reconstruction error, suggesting those embeddings were most consistent with the ontologies' graph structure.

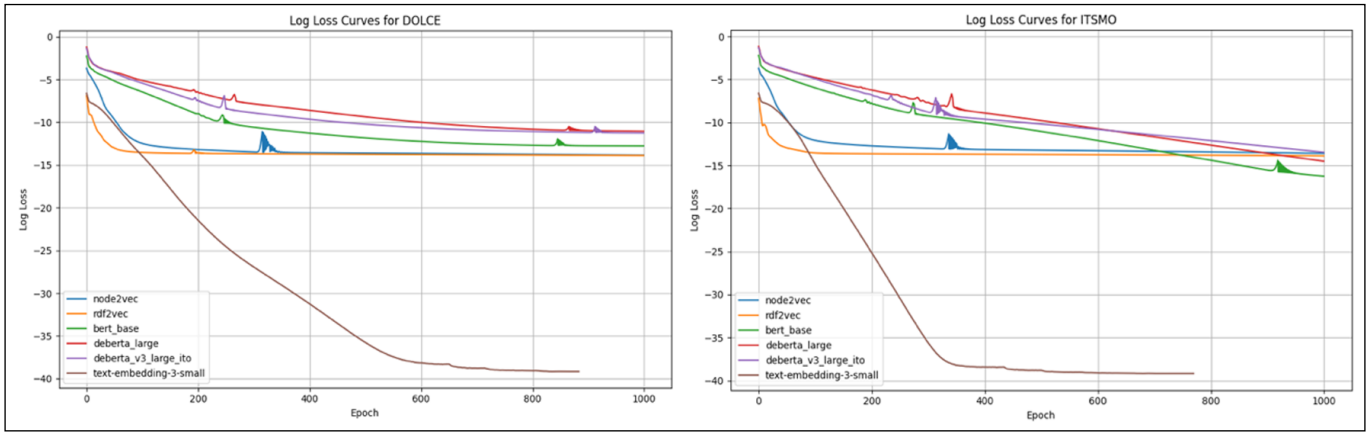

Figure 5 Loss function graphs for a graph neural network initialized with different embeddings.

In practice, as we will discuss, the GNN's effect was mixed: it did modify the similarity distributions (we observed some smoothing of cosine similarities across class pairs), and it increased the recall of correct mappings in the top-3 cluster for some methods, but it sometimes reduced the top-1 precision. This is likely due to the very small size of our graphs—with only dozens of nodes, the GNN has limited room to improve representation and might instead overfit to trivial patterns (like making all siblings more similar even if they are semantically different). We include the GNN results to analyze this effect.

CANDIDATE FILTERING VIA Z-SCORE CLUSTERING

Regardless of similarity method (embedding or LLM), each DOLCE class ends up with a ranked list of ITSMO classes. However, not every DOLCE class will have a valid corresponding subclass in ITSMO—some DOLCE upper categories might not be represented in the ITSMO scope. We therefore need a way to decide how many of the top candidates (0, 1, 2, or 3) to accept as final mappings for each DOLCE class.

We achieve this by analyzing the distribution of similarity scores for each DOLCE class and applying a Z-score threshold:

After similarity computation—cosine similarity for embeddings or mean confidence for GPT-4o—we have, for every DOLCE class D, a score vector

$$S_D = (s_{D,1}, \ldots, s_{D,|I|}), I = \{ITSMOclasses\},$$

We compute the mean (μ) and standard deviation (σ) of these scores for class D.

We identify the top three highest-scoring ITSMO classes:

$$S_1 \geq S_2 \geq S_3.$$

For each shortlisted score, compute its Z-score relative to D's distribution:

$$Z_k = \frac{S_k - \mu_D}{\sigma_D} (k = 1,2,3).$$

Z high Z_k means the candidate deviates positively from the background of random matches for that particular DOLCE class.

Empirical tuning on a manually checked validation subset showed that LLM-derived scores have a broader spread than embedding-based scores. Accordingly, we set two method-specific cut-offs:

$$Z_0 = \begin{cases} 0.005(LLM), \\ 1.3(embedings) \end{cases}; \text{a mapping} (D, I_k) \text{ is retained if } Z_k \geq Z_0.$$

The *Z*-score filter acts as an outlier detector: only those ITSMO classes whose similarity to a given DOLCE class exceeds the class-specific mean by ≥0.005 σ (LLM scores) or ≥1.3 σ (embedding scores) are accepted. This suppresses spurious matches that often arise for very general DOLCE nodes such as Entity or Abstract, whose score distributions are otherwise almost flat. At the same time, the rule still permits legitimate one-to-many alignments—on average 1.8 ITSMO subclasses are retained per mapped DOLCE class—yielding a clean set of candidate subclass assertions that forms the draft merged ontology, later verified by an OWL reasoner and a domain expert.

RESULTS

After implementing the mapping methodology, we applied it to map all 43 ITSMO classes into the DOLCE-lite hierarchy. The outcome is the new ontology—an OWL ontology containing DOLCE's classes and the mapped ITSMO classes as subclasses. The total class count is 80 (37 DOLCE + 43 ITSMO).

Figure 6 shows the merged taxonomy exactly as it appears in Protégé. Gray boxes on the left are the 37 original DOLCE-lite classes; purple boxes on the right are the 43 ITSMO classes that have been re-attached under their most plausible upper-level parents.

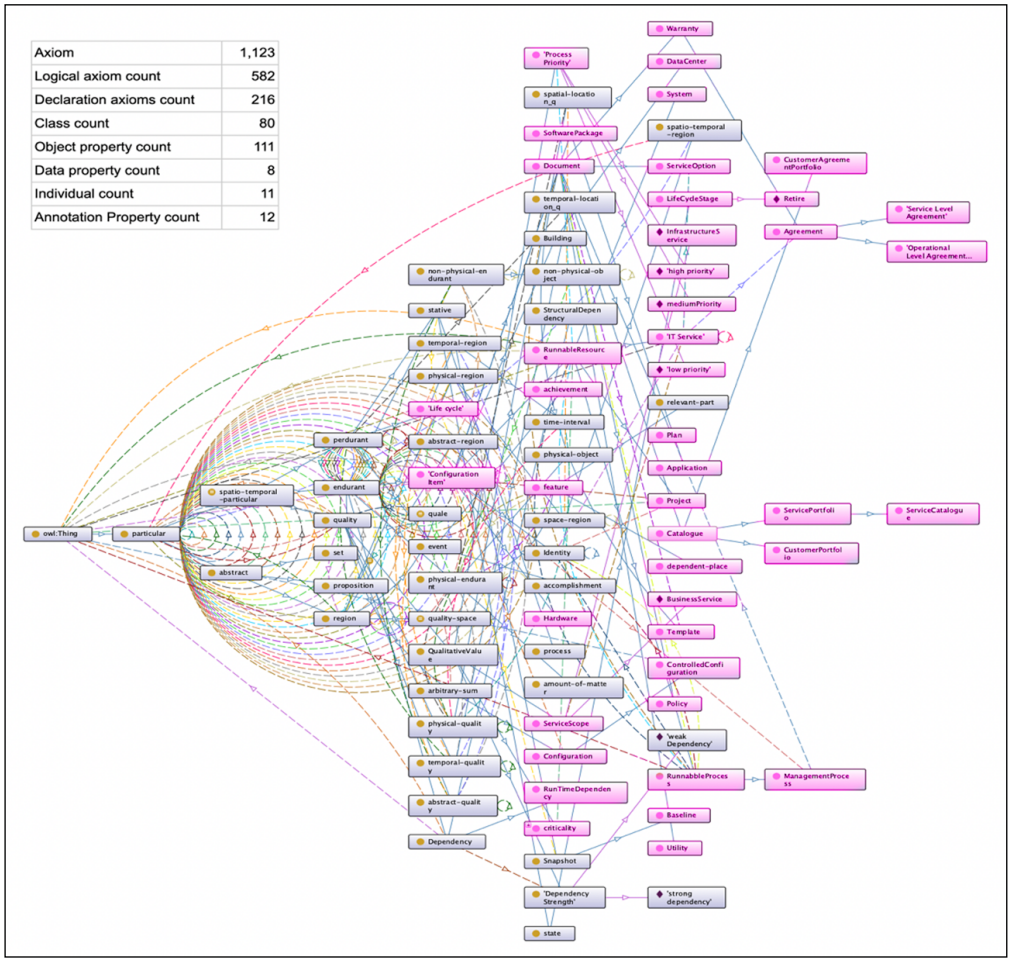

Figure 6 Resulting ontology after merging. Screenshot from Protégé OntoGraph. Extended classes marked in purple.

A few concrete alignments already make the merged model valuable for analytics. Event-like notions—Incident, Problem, and Change—are classified as subclasses of DOLCE Perdurant, so temporal reasoning is straightforward (e.g., spotting overlapping incidents). Tangible assets such as Rack, Server, and Data Center fall under Physical Object, while governance artifacts like Policy and Contract are placed beneath Social Object. Less obvious yet crucial: Service and Process are likewise treated as Perdurants, reflecting their time-dependent nature and enabling queries that hop from a configuration item to the services it hosts and the processes those services enact. With every ITIL term anchored to DOLCE's rigorously defined categories, disparate enterprise data (ticket logs, CMDBs, SLA documents, etc.) can be fused into one

knowledge graph and queried against a single semantic backbone—shrinking triage times, sharpening root-cause analysis, and paving the way for explainable AI assistants that can formally justify why an outage mattered.

GOLD-STANDARD CREATION AND EVALUATION PROTOCOL (HUMAN IN THE LOOP, TOP-K CHECKING)

Stage A—Candidate proposals (LLM-assisted). We first used an LLM to generate candidate alignments from DOLCE to ITSMO. For each DOLCE class, the model proposed one or more ITSMO classes together with a confidence-like score and a short rationale. This step served only to produce suggestions and to accelerate expert review.

Stage B—Expert consolidation (human evaluation). A domain expert then adjudicated the candidates: for each DOLCE class, the expert confirmed, replaced, or rejected the proposed pairs (binary validation at the pair level), and could introduce no-match labels when appropriate. The outcome is a human-evaluated gold standard matching: either a single adjudicated ITSMO class for DOLCE class or no-match. This adjudicated mapping subsequently materialized in the merged ontology and is the mapping we would deploy. After materialization, we ran OWL 2 DL reasoners (HermiT and Pellet) over the merged ontology to check logical consistency and class satisfiability. No inconsistencies or unsatisfiable classes were reported, providing an orthogonal, symbolic confirmation of the validity of the expert-consolidated gold (Thieblin *et al.*, 2021).

Stage C—Automatic methods to be evaluated. Independent of Stages A and B, we evaluated several automatic scoring methods that, for each DOLCE class d, produce a ranked list of ITSMO classes: (1) embedding-based similarity (multiple encoders), (2) GNN-enriched variants, and (3) an LLM scoring pipeline that assigns scores to (d,c) pairs. For fairness, the LLM scoring outputs were normalized to the same range as vector-based scores and treated exactly like another method. None of these methods influenced the gold labels.

Stage D—Top-k extraction. For each method m and class d, we extracted the top-3 candidates (or fewer if fewer candidates were available): $C_m^{(3)}(d) = \{c_{d,1}, c_{d,2}, c_{d,3}\}$.

Stage E—Metrics against human-evaluated gold standard. All metrics are computed exclusively against the expert gold. We report Top-1 Accuracy (%) and Top-3 Inclusion (%).

Non-circularity. The expert-adjudicated gold defines correctness; all methods—including the LLM scoring variant—are evaluated against this human gold. The initial LLM proposals in Stage A were assistive and did not determine the labels: the expert had full authority to confirm/reject/replace and to assign a no-match. Therefore, proposal and evaluation are decoupled, and no circular evaluation is performed; the HermiT/Pellet reasoning results further support the soundness of the consolidated gold.

EMBEDDING ANALYSIS AND SIMILARITY DISTRIBUTION

Before looking at accuracy numbers, it is informative to see how the various embedding strategies differed. We visualized the embeddings for all classes of both ontologies in a 2D plane using PCA (principal component analysis). Figure 7 shows one such visualization for a representative embedding set (e.g., BERT-base). In almost all embedding methods, the DOLCE class vectors and ITSMO class vectors occupy distinct regions of the space. They are nearly linearly separable as two groups, which is not surprising: the two ontologies have different vocabularies and contexts, so a model can usually distinguish 'this is a DOLCE concept' versus 'this is an ITSMO concept'. However, this separation does *not* imply that the closest ITSMO vector to a given DOLCE vector is a correct mapping. Linear separability means a simple classifier could tell which ontology a class belongs to, but our task is to bridge them, which requires deeper semantic alignment. Indeed, many of the nearest-neighbor pairs in the raw embedding space were not meaningful mappings, hence the need for further refinement and filtering.

To measure GNN impact on the embedding pairs during cosine similarity calculation, we quantify these effects on the ranking distribution by (1) the per-column variance of ITSMO→DOLCE cosine similarities, (2) the per-column IQR (interquartile range), and (3) the normalized entropy (0–1) of each DOLCE column. We also report a two-sample Kolmogorov–

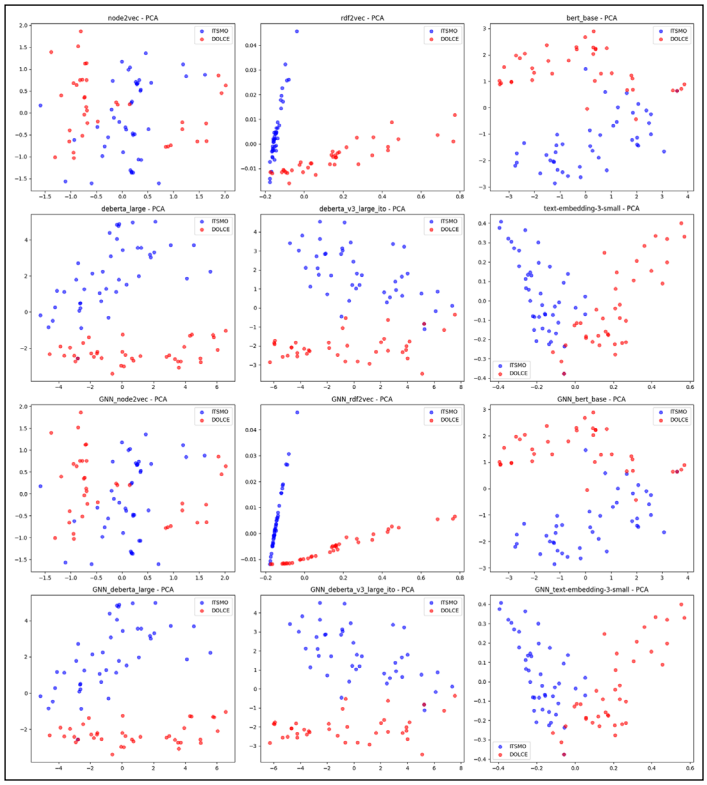

Figure 7 Visualization of embedding distribution in two-dimensional space using the PCA method.

Smirnov (K–S) test comparing all similarity values before vs. after GNN, and the mean column-wise maximum similarity ('mean max') as a proxy for peak concentration on the top candidate (see Table 2).

Table 2 The GNN induced a pronounced smoothing for RDF2Vec and minimal change for transformer-based embeddings (DeBERTa/BERT/text-embedding-3-small) and Node2Vec.

	rdf2vec	deberta_large	text-embedding-3-small	bert_base	deberta_v3_large_ito	node2vec
mean_var_before	**0,04977**	0,00156	0,004112	0,002142	0,002241	0,01593
mean_var_after	**0,037378**	0,00156	0,004112	0,002142	0,002242	0,01593
delta_mean_var	**–1,24E-02**	–2,90E-08	9,48E-10	6,80E-08	3,50E-07	4,40E-07
median_IQR_before	**0,31139**	0,041323	0,084101	0,058961	0,068114	0,146014
median_IQR_after	**0,112369**	0,041257	0,084101	0,058981	0,068094	0,146956
delta_median_IQR	**–1,99E-01**	–6,69E-05	4,63E-08	2,01E-05	–2,03E-05	9,43E-04
mean_Hnorm_before	**0,963367**	0,97806	0,955335	0,964831	0,969643	0,965019
mean_Hnorm_after	**0,977953**	0,978057	0,955335	0,964851	0,969646	0,965041
delta_mean_Hnorm	**1,46E-02**	–2,74E-06	–9,35E-09	1,92E-05	2,18E-06	2,22E-05
ks_D	**0,567669**	0,003759	0,000627	0,003759	0,002506	0,005639
ks_p	**1,40E-237**	1,00E+00	1,00E+00	1,00E+00	1,00E+00	1,00E+00
mean_max_before	**0,890243**	0,90308	0,394644	0,867358	0,886934	0,71366
mean_max_after	**0,918296**	0,903112	0,394644	0,867395	0,886976	0,713109
delta_mean_max	**2,81E-02**	3,18E-05	5,10E-08	3,67E-05	4,17E-05	–5,51E-04

The 'smoothing/balancing' effect of our GNN is embedding-dependent: it is strong for distributional graph embeddings (RDF2Vec), negligible for modern transformer text embeddings, and Node2Vec under our settings. We therefore refine our claim: GNN post-processing can substantially stabilize similarity distributions when base embeddings are noisy/high-variance but provides limited gains when base distributions are already tight.

MAPPING ACCURACY AND COMPARISON OF APPROACHES

Table 3 summarizes the mapping accuracy results for each approach we tested. Each row corresponds to an embedding or method (with and without GNN enhancement), plus a random baseline and the GPT-4 LLM approach. The key columns are the Top-1 Accuracy (%) and Top-3 Inclusion (%) as defined above, and Matched Classes, which is the total count of ITSMO classes mapped (including duplicates if a class appeared as a top candidate for multiple DOLCE classes).

Table 3 Summary of class alignment accuracy results.

EMBEDDING/METHOD	TOP-1 ACCURACY (%)	TOP-3 INCLUSION (%)	MATCHED CLASSES
Random (baseline)	27.6	31.0	63
node2vec	10.5	21.1	103
GNN_node2vec	7.9	18.4	102
rdf2vec	0.0	0.0	2
GNN_rdf2vec	0.0	0.0	0
bert_base	26.3	50.0	105
GNN_bert_base	28.9	57.9	105
deberta_v3_large	39.3	64.3	60
GNN_deberta_v3_large	35.7	60.7	60
deberta_v3_large_ito (FT)	34.2	63.2	92
GNN_deberta_v3_large_ito	28.9	57.9	92
text-embedding-3-small	28.9	63.2	113
GNN_text-embed-3-small	21.1	71.1	113
GPT-4o (LLM labels)	**73.5**	**82.4**	55

Several clear patterns emerge:

- LLM leads. GPT-4o tops the table (73.5 % Top-1, 82.4 % Top-3), confirming that zero-shot LLM reasoning is the most reliable alignment signal.
- Transformer embeddings are solid runners-up. DeBERTa-v3-large achieves ~64% Top-3; fine-tuning on ITIL text adds little, indicating that large pre-trained models already capture enough semantics for small ontologies.
- Classical graph embeddings collapse. RDF2Vec and Node2Vec score at or below a random 27.6% Top-1 baseline, showing that structural cues alone are ineffective at this scale.
- GNN smoothing is inconsistent. GraphSAGE sometimes boosts Top-3 recall but typically lowers Top-1 precision; the tiny graphs provide too little structure for a net gain.
- Fine-tuning gains are marginal. Extra domain data did not noticeably raise performance, underscoring the strength of zero-shot transformers.

Overall, the pipeline maps about four-fifths of ITSMO classes to plausible DOLCE parents without manual labels, with GPT-4o providing the decisive edge and transformer embeddings offering a practical non-LLM fallback.

DISCUSSION

The results demonstrate the feasibility of creating a merged ontology without direct expert input, confirming our central hypothesis. However, a number of insights, limitations, and implications emerge from this study.

PRACTICAL APPLICABILITY AND VALUE

From a practical perspective, the ability to generate an initial ontology mapping automatically can greatly accelerate knowledge graph development in enterprises. In our case, the new ontology provides a formal integration of ITIL concepts (via ITSMO) with an upper ontology

(DOLCE). This merged ontology can now serve as a backbone for building an IT knowledge graph that is both consistent (through DOLCE's rigor) and rich in domain detail (through ITSMO).

One of the most significant implications of our work is proof that ontology engineers can leverage AI to do the heavy lifting of class alignment. This could lower the barrier for organizations to adopt semantic technologies. Instead of requiring a panel of ontology experts in ITIL to map every class, one could run an automated alignment like ours to get a draft ontology, which an expert then only needs to review and refine, not create from scratch. Even with our ~82% accuracy, an expert would only need to correct or add mappings for the remaining 18%, a much smaller effort than manual mapping of 100% of classes.

Our method could be seen as an Agent-assisted ontology engineering: the AI does the first pass mapping, then experts refine. This is akin to how, in other domains, an AI Agent is used to generate a draft design or plan, which humans finalize.

SCALABILITY AND DOMAIN GENERALITY

Scalability: A current limitation of our pipeline is the reliance on GPT-4o prompts for best performance. GPT-4o was used with the entire set of classes as input, which worked because we only had ~80 classes to compare. If we were to map a much larger ontology (say thousands of classes) to another, we cannot simply include all class descriptions in a single prompt at larger scales. While modern GPT-4-class models support long contexts (≥128k tokens and, for the GPT-4.1 family, up to ~1M tokens), very large ontologies may still exceed practical context and cost budgets; thus, we adopt blocking or hierarchical strategies. The computational cost of 500 GPT-4o queries for each class is non-trivial as well—though for 37 DOLCE classes it was manageable, for hundreds it would be expensive and slow. That said, we can optimize: if domain ontology has N classes and TLO has M classes, doing N × M LLM evaluations is the worst case. In practice, a hierarchical or blocking approach could limit comparisons (e.g., first map broad categories, then refine within sub-trees). Alternatively, one could use a smaller LLM or a fine-tuned model to emulate GPT-4o for large tasks. Scalability of embedding methods is better—computing embeddings for thousands of classes is fine, and even GNN on thousands of nodes is fine. So if needed, one might resort to a hybrid: use embeddings to narrow down candidate matches, then use an LLM on the short-listed pairs.

Generality to other domains: We believe our approach is quite general. The prerequisite is having a top-level ontology and a domain ontology to align. Many domains have their own ontologies (medical, financial, manufacturing, etc.) and could use a foundational ontology alignment. For example, mapping a medical ontology to BFO, or a geospatial ontology to SUMO. The challenges might be domain-specific (e.g., jargon that LLMs might not know well), but with domain literature or description text, it should be feasible. The key components—embedding computation, optional GNN enrichment, LLM prompting, *Z*-score filtering—are domain-agnostic.

However, one must be cautious: an ontology alignment is only as good as the ontologies' documentation. If class descriptions are poor or non-existent, embedding and LLM methods falter. We were fortunate that both DOLCE and ITSMO have decent textual descriptions for classes. This is a limitation in generality—we assume well-documented ontologies.

Automated ontology extension: Another aspect of scalability is extending beyond just mapping existing classes to adding new classes. Our current mapping is a one-time alignment. The future extension (as mentioned in the 'Future Work' section below) involves discovering if there are concepts in the IT domain not covered by ITSMO and adding them. That moves beyond mapping into *ontology learning*.

LIMITATIONS AND POTENTIAL IMPROVEMENTS

Despite the success, there are important *limitations* to acknowledge:

> *Reliance on LLM availability:* Our best results came from a proprietary model (GPT-4o). This raises questions of reproducibility and long-term availability. Organizations might be hesitant to rely on an external API for building their ontologies due to privacy or

cost concerns. A possible improvement is to use open-source LLMs or smaller models fine-tuned for this task (Guarino and Welty, 2009) (OntoClean).

Limited use of axioms. Apart from the subclass lattice, we ignored richer axioms (e.g., part-of, domain/range restrictions). Consequently, contextual cues—such as an ITSMO class heavily linked to hardware—never influence alignment. Future iterations should combine symbolic reasoning with embeddings so that structural constraints reinforce, or override, purely lexical similarity.

Although the same family of LLMs was used to propose candidates, the evaluation relied exclusively on human gold labels obtained via expert adjudication; therefore, the study does not employ circular evaluation. The LLM's role is strictly assistive—to prioritize candidates and reduce expert time—not to arbitrate correctness.

FUTURE EXTENSIONS

We envisage three technical strands.

1. Ontology expansion. Using the merged ITO-seed as a backbone, we will mine enterprise corpora—incident tickets, CMDB records, knowledge articles—via NER + relation-extraction models to surface terms and relations not yet represented. Candidate concepts will be clustered in embedding space, provisionally typed by an LLM, and inserted as new subclasses or properties, followed by reasoner validation and expert veto. The cycle 'text → candidates → ontology update' can iterate continuously.
2. Richer alignment. Extending beyond subclass links, we will map object/data properties and allow justified multi-inheritance or equivalence assertions. Instance data will be reconciled against the ontology to flag misclassifications and refine the schema. The same pipeline can merge additional domain ontologies (e.g., business-process or security models) through the common DOLCE spine, yielding an interoperable, cross-domain knowledge fabric.
3. Tooling and deployment. A lightweight interface will present model-suggested mappings with confidence scores, let engineers approve or correct them, and feed the feedback into active-learning retraining. Pilot deployment in an IT-service desk will measure downstream impact—for example, incident routing accuracy and data-entry consistency—to quantify the value of AI-assisted ontology maintenance.

CONCLUSION

We presented a method for automatic mapping of a top-level ontology (DOLCE-lite) to a domain-specific ontology (ITSMO) in the IT service management domain, achieving a high alignment accuracy without human-curated training data or direct expert involvement. Our approach combined semantic embeddings, graph neural network augmentation, and large language model prompting to generate and evaluate candidate mappings between ontology classes. The resulting integrated new ontology extends DOLCE with ITIL-related concepts from ITSMO, demonstrating the viability of creating a rich domain ontology autonomously.

GNN-based enrichment did smooth similarity distributions for RDF2Vec but had minimal effect on transformer embeddings and Node2Vec in our setting. Thus, its practical benefit is context- and embedding-dependent being most useful when base similarities are high-variance.

Our case study indicates that semi-automatic, expert-in-the-loop alignment can substantially reduce effort for class-level mappings. In our setting, methods that leverage textual and learned semantic features performed best, while graph structure contributed only marginal gains at this scale. However, these results are conditional rather than universal. The approach was effective because (1) both ontologies were moderately sized and well documented (labels, definitions, and glosses), (2) there was conceptual overlap between source and target domains, (3) the candidate space fit within practical context-length and compute budgets, and (4) a human adjudicator resolved borderline cases and enforced consistency.

Accordingly, we do not claim that upper-level ontologies can be extended automatically in general. For very large, poorly documented, or highly domain-shifted ontologies; for mappings

beyond classes (e.g., properties, restrictions, and axioms); or where terminology is sparse/ambiguous, manual curation remains essential, and we expect quality to degrade without substantial additional supervision. Even under favorable conditions, we recommend governance steps—expert adjudication, OWL-DL reasoning (e.g., HermiT/Pellet) to check consistency and satisfiability, and audit trails—to mitigate spurious alignments.

In sum, our contribution should be read as an existence proof for assistive automation under favorable conditions, not as evidence that ontology extension is broadly automatic. The path forward, in our view, is a hybrid workflow that marries expert oversight and symbolic checks with LLM-based candidate generation and ranking.

Future research should focus on refining this synergy—for example, investigating how explanation capabilities of LLMs might justify mappings (increasing trust in automatically generated ontologies) or how ontologies can help curb LLMs' well-known issues like hallucinations by providing a structure to conform to. Our work opens the door to a future where building and updating ontologies is a continuous, largely autonomous process, driven by data and AI, with humans steering at a high level. This, we believe, will be key to maintaining relevant and accurate knowledge bases in the fast-changing landscape of enterprise IT and beyond.

COMPETING INTERESTS

The authors have no competing interests to declare.

AUTHOR AFFILIATIONS

Andrey Khalov orcid.org/0009-0005-4584-8245
Moscow Institute of Physics and Technology, National Research University, Moscow, Russia
Olga Ataeva orcid.org/0000-0003-0367-5575
Federal Research Center, Computer Sciences and Control, Russian Academy Of Sciences, Moscow, Russia

REFERENCES

Amini, S. *et al*. (2023) 'Zero-shot ontology alignment with large language models', in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8974–8989.

Borgo, S. *et al*. (2023) 'DOLCE: A descriptive ontology for linguistic and cognitive engineering', *Applied Ontology*, 17(1), pp. 45–69. Available at: https://doi.org/10.3233/AO-210259

Chen, J. *et al*. (2021) 'OWL2Vec*: Embedding of OWL ontologies', *Machine Learning*, 110(8), pp. 1813–1845. Available at: https://doi.org/10.1007/s10994-021-05997-6

Choi, N., Song, I.-Y. and **Han, H.** (2006) 'A survey on ontology mapping', *ACM SIGMOD Record*, 35(3), pp. 34–41. Available at: https://doi.org/10.1145/1168092.1168097

DOLCE overview (official): Laboratory for applied ontology (ISTC-CNR). DOLCE: Overview. Available at: loa.istc.cnr.it (Accessed: 08 August 2025).

DOLCE-Lite (OWL): DOLCE-Lite OWL (ontology file). Available at: https://www.loa.istc.cnr.it/ontologies/DOLCE-Lite.owl (Accessed: 08 August 2025).

El Yamami, A. *et al*. (2019) 'An ontological representation of ITIL framework service level management process', in *Proceedings of the 3rd International Conference on Signals, Distributed Systems and Artificial Intelligence (SDSAI 2018)*. Springer. Available at: https://doi.org/10.1007/978-3-030-11914-0_9

Euzenat, J. and **Shvaiko, P.** (2013) *Ontology matching*, 2nd ed. Springer. Available at: https://doi.org/10.1007/978-3-642-38721-0

Faria, D. *et al*. (2013) 'The AgreementMakerLight ontology matching system', in *Proceedings of the OTM 2013 Workshops*, LNCS 8185, pp. 527–541. Available at: https://doi.org/10.1007/978-3-642-41030-7_38

Fernandez-Lopez, M. and **Gomez-Porez, A.** (2002) 'Overview and analysis of methodologies for building ontologies', *The Knowledge Engineering Review*, 17(2), pp. 129–156. Available at: https://doi.org/10.1017/S0269888902000462

Glimm, B. *et al*. (2014) 'HermiT: An OWL 2 Reasoner', *Journal of Automated Reasoning*, 53(3), pp. 245–269. Available at: https://doi.org/10.1007/s10817-014-9305-1

Guarino, N. and **Welty, C.A.** (2009) 'An overview of OntoClean', in S. Staab and R. Studer (eds.) *Handbook on Ontologies*. Springer, pp. 151–171. Available at: https://doi.org/10.1007/978-3-540-92673-3_9

Hamilton, W.L., Ying, R. and **Leskovec, J.** (2017) 'Inductive representation learning on large graphs', in *Advances in Neural Information Processing Systems 30* (NeurIPS 2017), pp. 1024–1034.

He, P. *et al*. (2021) 'DeBERTa: Decoding-enhanced BERT with disentangled attention', in *International Conference on Learning Representations (ICLR 2021)*.

He, Y. *et al*. (2022) 'BERTMap: A BERT-based ontology alignment system', *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(5), pp. 5684–5691. Available at: https://doi.org/10.1609/aaai.v36i5.20510

He, Y. *et al*. (2023) 'Exploring large language models for ontology alignment', in *Proceedings of the ISWC 2023* (Posters & Demos Track), CEUR Workshop Proc. vol. 3632.

ITSMO: IT Service Management Ontology (ITSMO). Canonical resolver. Available at: https://w3id.org/itsmo; catalog entry in LOV: "IT Service Management Ontology (itsmo)". (Accessed: 08 August 2025). ontology.it; lov.linkeddata.es.

Jimenez-Ruiz, E. and **Cuenca Grau, B.** (2011) 'LogMap: Logic-based and scalable ontology matching', in *Proceedings of the 10th International Semantic Web Conference (ISWC 2011)*, LNCS 7031, pp. 273–288. Available at: https://doi.org/10.1007/978-3-642-25073-6_18

Jimenez-Ruiz, E. *et al*. (2022) 'Ontology alignment evaluation initiative: Six years of experience', *Journal of Data Semantics*, 11(3), pp. 191–207.

Karadeniz, I. and **Ozgur, A.** (2019) 'Linking entities through an ontology using word embeddings and syntactic re-ranking', *BMC Bioinformatics*, 20, p. 156. Available at: https://doi.org/10.1186/s12859-019-2678-8

Lewis, P. *et al*. (2020) 'Retrieval-augmented generation for knowledge-intensive NLP tasks', *Advances in Neural Information Processing Systems*, 33, pp. 9459–9474.

Li, J. *et al*. (2023) 'On the initialization of graph neural networks', in *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*, PMLR 202, pp. 19911–19931.

Noy, N.F. and **McGuinness, D.L.** (2001) *Ontology development 101: A guide to creating your first ontology*. Stanford University.

Noy, N.F. and **Musen, M.A.** (2001) 'Anchor-PROMPT: Using non-local context for semantic matching', in *Proceedings of the IJCAI-2001 Workshop on Ontologies and Information Sharing*.

OpenAI. GPT-4o mini model documentation. Available at: https://platform.openai.com/docs/models/gpt-4o-mini (Accessed 08 August 2024).

Pastuszak, J., Czarnecki, A. and **Orlowski, C.** (2012) 'Ontologically aided rule model for the implementation of ITIL processes', *Frontiers in Artificial Intelligence and Applications*, 243, pp. 1428–1438.

Qiang, Z., Wang, W. and **Taylor, K.** (2024) 'Agent-OM: Leveraging LLM agents for ontology matching', *Proceedings of the VLDB Endowment*, 18(3), pp. 516–529. Available at: https://doi.org/10.14778/3712221.3712222

Ristoski, P. and **Paulheim, H.** (2016) 'RDF2Vec: RDF graph embeddings for data mining', in *Proceedings of the 15th International Semantic Web Conference (ISWC 2016)*, LNCS 9981, pp. 498–514. Available at: https://doi.org/10.1007/978-3-319-46523-4_30

Sirin, E. *et al*. (2007) 'Pellet: A practical OWL-DL reasoner', *Journal of Web Semantics*, 5(2), pp. 51–53. Available at: https://doi.org/10.1016/j.websem.2007.03.004

Smith, B. (2008) 'Ontology (science)', in *Formal Ontology in Information Systems (FOIS 2008)*. IOS Press, pp. 21–35.

Thieblin, E., Haemmerle, O. and **Trojahn, C.** (2021) 'Automatic evaluation of complex alignments: An instance-based approach', *Semantic Web*, 12(5), pp. 767–787. Available at: https://doi.org/10.3233/SW-210437

TO CITE THIS ARTICLE:
Khalov, A. and Ataeva, O. 2025 Automating Ontology Mapping in IT Service Management: A DOLCE and ITSMO Integration. *Data Science Journal*, 24: 23, pp. 1–19. DOI: https://doi.org/10.5334/dsj-2025-023

Submitted: 28 November 2024
Accepted: 11 August 2025
Published: 03 September 2025

Data Science Journal is a peer-reviewed open access journal published by Ubiquity Press.