DeepOnto: A Python Package for Ontology Engineering with Deep Learning

Yuan He [a], Jiaoyan Chen [b,a], Hang Dong [a], Ian Horrocks [a], Carlo Allocca [c], Taehun Kim [c] and
Brahmananda Sapkota [c]

[a] *Department of Computer Science, University of Oxford, UK*
E-mails: yuan.he@cs.ox.ac.uk, hang.dong@cs.ox.ac.uk, ian.horrocks@cs.ox.ac.uk
[b] *Department of Computer Science, The University of Manchester,UK*
E-mail: jiaoyan.chen@manchester.ac.uk
[c] *Samsung Research, UK*
E-mails: c.allocca@samsung.com, taehun11.kim@samsung.com, b.sapkota@samsung.com

Abstract.

Integrating deep learning techniques, particularly language models (LMs), with knowledge representation techniques like ontologies has raised widespread attention, urging the need of a platform that supports both paradigms. Although packages such as OWL API and Jena offer robust support for basic ontology processing features, they are not designed to process ontologies into various data formats (e.g., text and graph) crucial for supporting subsequent, deep learning-based applications. Moreover, widely-used ontology APIs are primarily Java-based while deep learning frameworks like PyTorch and Tensorflow are mainly for Python programming. To address the needs, we present DeepOnto, a Python package designed for ontology engineering with deep learning. The package encompasses a core ontology processing module founded on the widely-recognised and reliable OWL API, encapsulating its fundamental features in a more "Pythonic" manner and extending its capabilities to incorporate other essential components including reasoning, verbalisation, normalisation, taxonomy, projection, and more. Building on this module, DeepOnto offers a suite of tools, resources, and algorithms that support various ontology engineering tasks, such as ontology alignment and completion, by harnessing deep learning methods, primarily pre-trained LMs. In this paper, we also demonstrate the practical utility of DeepOnto through two use-cases: the Digital Health Coaching in Samsung Research UK and the Bio-ML track of the Ontology Alignment Evaluation Initiative (OAEI).

Keywords: Ontology, Ontology Engineering, Deep Learning, Language Model, OWL, Python

Repository: https://github.com/KRR-Oxford/DeepOnto
Documentation: https://krr-oxford.github.io/DeepOnto/
License: Apache License, Version 2.0

1. Introduction

An ontology is a formal, explicit specification of knowledge within the scope of a domain. It provides a vocabulary of concepts and properties that enables a shared understanding of semantics among humans and machines, with wide applications in many domains such as bioinformatics, information systems, and the Semantic Web. Ontology engineering, a sub-field of knowledge engineering, underpins the various stages of ontology development, encompassing ontology design, construction, curation, evaluation, and maintenance, among others [1]. Concrete tasks of ontology engineering include: *(i)* defining the entities and constructing the logical axioms that make up an ontology, *(ii)* validating and ensuring quality (e.g., completeness and correctness) of an ontology, *(iii)* inserting new knowledge into an ontology, *(iv)* integrating domain ontologies that come from heterogeneous sources, and so on. These tasks can collectively enhance an ontology's practical utility, making it applicable to different real-world scenarios.

Deep learning approaches have gained significant popularity across various research and engineering domains. These techniques have shown notable advantages over conventional ontology engineering tools. For example, LogMap [2], a long-standing state-of-the-art ontology alignment system, relies on lexical similarity and lacks the ability to capture textual contexts. In contrast, BERTMap [3], a language model (LM)-based system which leverages the attention mechanism of the transformer architecture for contextual text embeddings [4], can be more robust to linguistic variations such as synonyms and polysemies. Another example concerns ontology completion. Traditional systems, leveraging formal logics and/or heuristic rules, are capable of inferring entailed knowledge (e.g., HermiT for ontology reasoning [5]). But the requirement of manual design and curation for these logics and rules frequently results in their under-specification. This constraint significantly curtails their capability to deduce absent but plausible knowledge. As opposed to these rule-based solutions, deep learning-based techniques can automatically learn patterns from the existing knowledge, the metadata and different kinds of other information, and make predictions accordingly [6–8]. Nevertheless, the benefit is not single sided. The formal and structural semantics embedded in ontologies can augment deep learning models, enhancing not only their training efficiency but also their interpretability. For instance, leveraging structural contexts via graph-based attention mechanisms has yielded substantial advancements in predictive modelling [9]. Furthermore, the issue of mitigating hallucinated responses produced by recent large language models (LLMs) can be addressed through the incorporation of attributions [10, 11], potentially from symbolic knowledge. In this context, ontologies can play a crucial role, acting as robust and trustworthy reference points to validate and support the generated answers.

However, there lacks a systematic support for integrating ontology engineering with deep learning, posing challenges for both developers and users. While there are packages such as OWL API [12] and Jena [13] that effectively support basic ontology processing features, particularly those associated with OWL (Web Ontology Language) [14] – a prominent ontology language grounded in Description Logic – to the best of our knowledge, no existing packages are designed to transform ontologies into diverse data formats (e.g., text and graph) to facilitate a broad spectrum of deep learning-based ontology engineering solutions. This is further complicated by the fact that leading deep learning frameworks like Tensorflow [15] and PyTorch [16] primarily offer Python-based APIs, whereas the majority of ontology APIs are Java-based. Although the mOWL toolkit [17] resolves this gap by employing JPype[1] to bridge Python and Java, it mainly focuses on ontology embedding models and their applications to the biomedical domain, thereby neglecting fundamental ontology processing features and a variety of essential tasks pivotal for ontology construction and curation. In order to provide a comprehensive, general, and versatile package for supporting deep learning-based ontology engineering, we develop DeepOnto.

Figure 1 depicts the architecture of DeepOnto. It consists of an ontology processing module as the foundation, which can support basic operations like loading, saving, retrieving entities, querying for ancestors/descendants, and modifying entities and axioms, as well as more advanced functions like reasoning, verbalisation, normalisation, and projection (see Section 3 for their specifications). Built upon this basic module, DeepOnto features a collection of ontology engineering tools and resources, devised for ontology alignment, completion, and ontology-based LM probing. DeepOnto provides a fairly flexible and extensible interface for further implementations. This includes our ongoing efforts, such as logic embedding [18] and the identification and insertion of new concepts [19], as well as other typical works that can facilitate ontology construction and curation, such as the OWL ontology embedding method OWL2Vec* [8]. The incorporation of these new tools and resources may necessitate the integration of other fundamental features into the core ontology processing module, further enhancing its capabilities and robustness. Through this positive development cycle, we expect DeepOnto to emerge as a powerful package for the community, providing general support and fostering innovation within the field.

2. Design Principle

2.1. Dependencies

We chose the OWL API as the backend dependency due to its stability, reliability, and widespread adoption in notable projects and tools, such as Protégé [20], ROBOT [21] and HermiT [5]. DeepOnto was initially built on Owl-

---

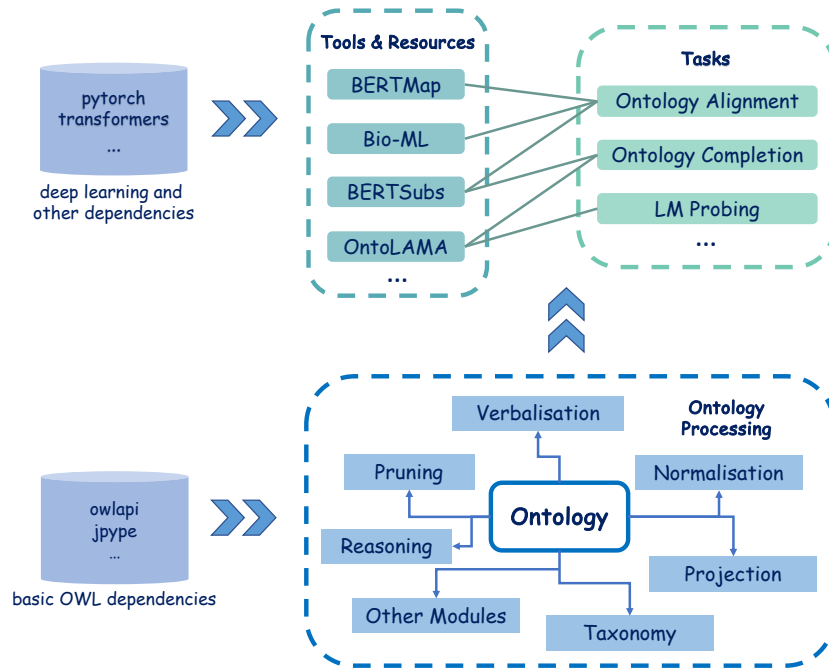[1]https://jpype.readthedocs.io/en/latest/install.html

Fig. 1. Illustration of DeepOnto's architecture, with the lower half depicting the core ontology processing module, and the upper half presenting various tools and resources for diverse ontology engineering tasks. The arrow signs indicate dependency support.

ready2[2] [22], a Python-based ontology API. However, we found that Owlready2 is still in a preliminary stage and lacks support for several fundamental features. For instance, runtime errors are frequently triggered when attempting to delete entities from an ontology. Furthermore, it also posed difficulties when dealing with multiple ontologies simultaneously, as entities are loaded into a shared space without a reference to their original ontologies. While the underlying library RDFLib[3], on which Owlready2 is built, does offer a platform to process RDF triples directly and introduce some missing functionalities for handling OWL ontologies, its use would necessitate significant additional development. This makes RDFLib a less viable option considering the developmental effort and resources required. To facilitate the import of OWL API, we adopted the solution from the mOWL[4] toolkit [17], which utilises JPype to connect the Java Virtual Machine (JVM) with Python programming. Specifically, the Java dependency files are compiled into the Java Archive (JAR) format and will be shipped upon package installation. JVM can then be launched within a Python program, establishing a connection to the JAR files. We constrain the direct import of the Java dependencies within the central ontology processing module, maintaining them in a relatively static version. This strategy ensures long-term stability and streamlines the updating and management processes of the codebase.

DeepOnto adopts PyTorch [16] as the backbone for deep learning dependencies. PyTorch is characterised by its dynamic computation graph, which enables runtime modification of the model's architecture, providing flexibility and ease of use for users. Also, significant efforts are invested to ensure backward compatibility. Currently, ontology engineering modules in DeepOnto mainly target applications of language models (LMs), which are well supported by the Huggingface's Transformers library [23]. Building upon this, the OpenPrompt library [24] supports the prompt learning paradigm, which is one of the key foundations of recent cutting-edge large language models, such as ChatGPT [25] and LLaMA 2 [26].

---

[2]https://github.com/pwin/owlready2
[3]https://rdflib.dev/
[4]mOWL is not used as a direct dependency but parts of its functionalities (e.g., normalisation and projection) encapsulated in Java were migrated to DeepOnto to avoid duplicated implementations.
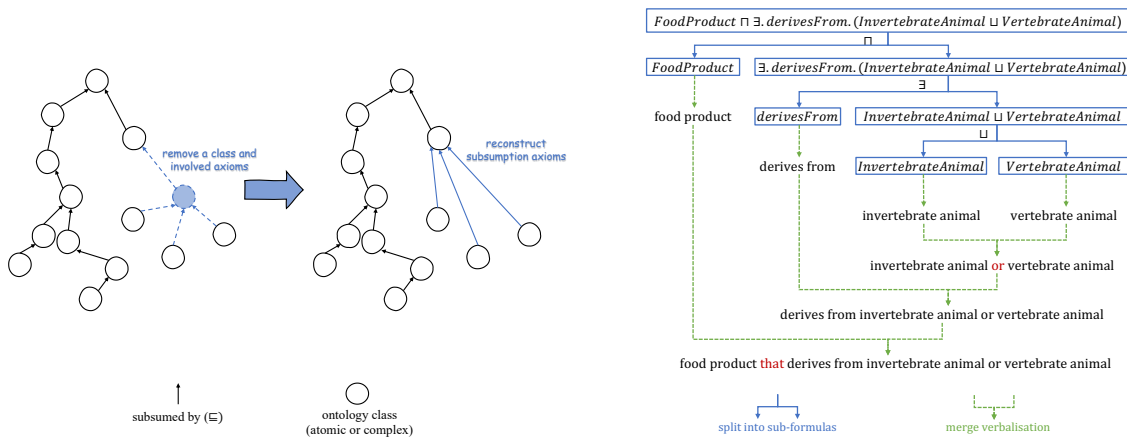
Fig. 2. The left figure illustrates the process of removing a concept while preserving the subsumption hierarchy in the ontology pruning algorithm proposed in Bio-ML [27]. The right figure illustrates an example of the application of the recursive concept verbalisation algorithm proposed in OntoLAMA [28].

2.2. Architecture

The architecture of DeepOnto is straightforward and succinct. As shown in Figure 1, the basis of DeepOnto is the core ontology processing module, which comprises a collection of essential sub-modules that revolve around the main class[5] `Ontology`. The ontology class serves as the main entry point for introducing the OWL API's features, such as accessing ontology entities, querying for ancestor/descendent (and parent/child) concepts, deleting entities, modifying axioms, and retrieving annotations. These functions are encapsulated in a more cohesive and easy-to-use[6] manner. Along with these basic functionalities, we introduce several essential sub-modules to enhance the core module, such as reasoning, pruning, verbalisation, normalisation, projection, and more. One shared objective of all these components is to transform an ontology into various data forms, such as verbalising an ontology entity into natural language text and projecting an ontology into an RDF graph, thereby facilitating deep learning-based ontology engineering solutions.

The core ontology processing module paves the way for implementing individual tools and resources to support ontology construction and curation. At present, DeepOnto mainly incorporates systems based on pre-trained LMs, and has covered the tasks of ontology alignment, ontology completion with subsumptions, and ontology-based LM probing. In the near future, DeepOnto is expected to incorporate new tools for ontology embedding, concept insertion, and more. Notably, the DeepOnto system can be easily updated and extended with new tools to support other knowledge engineering scenarios, such as entity linking, entity alignment, and link prediction for knowledge graphs mainly composed of relational facts.

3. Ontology Processing

In this section, we present a brief description for each component in the core ontology processing module.

Ontology The base class of DeepOnto is named as `Ontology`, which offers basic operations for accessing or modifying an ontology. An instance of `Ontology` can be initialised by taking an ontology file as input. Users can then access named entities (concepts and properties) through their IRIs, obtain asserted parents and children of an

---

[5]In this work, the term *"class"* refers to a blueprint for creating objects – encapsulating attributes and methods in Python programming.

[6]For instance, when using the OWL API, retrieving subsumption axioms for different entity types requires distinct codes. We have consolidated these similar functions into a single Python method for easier use.

entity, retrieve and modify axioms, and so on. When implemented in Java using OWL API, even simple features like these may require several lines of code. DeepOnto's encapsulation in Python improves code cleanliness and readability.

Ontology Reasoning Every instance of `Ontology` is accompanied by an instance of `OntologyReasoner` as its attribute. It is used for conducting reasoning activities, including obtaining inferred subsumers and subsumees, as well as checking entailment and consistency. By encapsulating these basic reasoning functions, we can implement more complex or specific reasoning algorithms. For example, we have implemented the *assumed disjointness* proposed by [28], which can be particularly useful in the negative sampling process frequently employed in various machine learning-driven ontology curation tasks, including but not limited to, subsumption prediction. Currently, DeepOnto supports three types of reasoners:

- **HermiT** [5], a sound and complete OWL reasoner based on hypertableau calculus.
- **ELK** [29], a highly optimised reasoner tailored to the OWL 2 EL profile.
- **Structural**[7], a simple structural reasoner.

Notice that DeepOnto can be easily extended to incorporate other reasoners[8] that inherit the reasoner interface of OWL API.

Ontology Pruning Real-world ontologies frequently exhibit a large-scale nature, leading to diminished efficiency during system evaluation. To extract a scalable subset from an ontology, we often prune the ontology by removing its concepts based on certain criteria, e.g., semantic types. In order to maintain the hierarchical structure during the pruning process, we implement the pruning algorithm proposed in [27] (found in `OntologyPruner`), which introduces subsumption axioms between the asserted (atomic or complex) parents and children of the concept targeted for removal (see Figure 2; left). For future development, we aim to incorporate more pruning approaches. A key addition will be ontology modularisation, a technique that seeks to extract a (small) sub-ontology that entails a given axiom or is sufficient to answer a specific concept of queries [30]. We also aim to explore its various variants that involve approximation, catering to different scenarios such as the construction of personalised knowledge graphs.

Ontology Verbalisation Verbalising an entity into natural language text with close meaning as its counterpart OWL statements can improve an ontology's accessibility and support many ontology engineering tasks. For example, ontology alignment systems often rely on string similarity or other text-level features to achieve successful results. While a named entity can be easily verbalised using its name (or labels), complex expressions that involve logical operators need a more sophisticated algorithm. In DeepOnto, we implement a recursive concept verbaliser (found in `OntologyVerbaliser`) proposed in [28], which can automatically transform a complex logical expression into a textual sentence based on entity names or labels available in the ontology. An example is shown in the right of Figure 2, where the complex concept expression $FoodProduct \sqcap \exists derivesFrom(InvertebrateAnimal \sqcup VertebrateAnimal)$ is parsed into a syntax tree of sub-formulas (this intermediate function is encapsulated in `OntologySyntaxParser`). The leaf nodes are named concepts or properties and they are verbalised directly. The recursion occurs when merging verbalised child nodes according to the logical pattern in their parent node, and terminates when the sentence is complete. In this example, the final output is *"food product that derives from invertebrate animal or vertebrate animal"*.

Ontology Normalisation Normalisation refers to the transformation of axioms into one of the following *normal forms*: $C \sqsubseteq D$, $C \sqcap C' \sqsubseteq D$, $C \sqsubseteq \exists r.D$, $\exists r.C \sqsubseteq D$, $r \sqsubseteq s$, and $r \circ r' \sqsubseteq s$, with no loss of semantics [31]. Here, C and C' can be named concepts or $\top$, D can be a named concept or $\bot$; and r, r', and s represent roles. The normalisation algorithm has been implemented in several $\mathcal{EL}$ embedding models [18, 32] and this feature is also implemented in DeepOnto for easy access. Normalisation can facilitate the training deep learning-based ontology engineering models because it simplifies the set of axiom patterns. For example, a complex subsumption $C \sqsubseteq D \sqcap \exists r.E$ can be normalised into $C \sqsubseteq D$ and $C \sqsubseteq \exists r.E$ such that a model trained on normalised axioms can make independent predictions for each axiom and then combine the results for a joint prediction.

---

[7]https://owlcs.github.io/owlapi/apidocs_4/org/semanticweb/owlapi/reasoner/structural/StructuralReasoner.html
[8]E.g., https://owlapi.sourceforge.net/reasoners.html

Ontology Taxonomy   An ontology defines a hierarchy of concepts through asserted subsumption axioms, resulting in a taxonomy. This taxonomy simplifies the ontology's structure, allowing easier navigation and retrieval by focusing exclusively on hierarchical relationships. This, in turn, facilitates graph-based deep learning models. In DeepOnto, the ontology taxonomy (found in `OntologyTaxonomy`) is implemented as a directed acyclic graph with named concepts as nodes and subsumption relations as directed edges. The top concept *owl:Thing* is used as the root node. To establish the subsumption edges, we employ an ontology reasoner (refer to the Ontology Reasoning section for more details) to infer *direct* subsumers for each named concept. According to the OWL API documentation[9], a direct subsumption implies that if the ontology entails $C_1 \sqsubseteq C_2$ and there is no intermediate concept C such that $C_1 \sqsubseteq C$ and $C \sqsubseteq C_2$, then C_2 is considered as a direct subsumer of C_1. Utilising a reasoner in this context helps to circumvent potential issues, such as mistakenly placing named concepts that are part of equivalence axioms directly under *owl:Thing*. To illustrate, if $C_1 \equiv C_2 \sqcap C_3$ is the only axiom about C_1 asserted in an ontology, then the reasoner can infer C_2 and C_3 as direct subsumers of C_1, thus avoiding placing C_1 directly under *owl:Thing* and making the taxonomy more complete.

Ontology Projection   DeepOnto offers the capability to transform an OWL ontology into a set of RDF triples. A default method for this transformation, which adheres to the W3C standard[10], is provided by OWL API. This method preserves the ontology's semantics and is effective for storing or exchanging ontologies. However, as it may introduce many blank nodes for representing complex logical expressions, its utility for ontology visualisation or applying graph-based algorithms, such as Random Walk and Graph Neural Networks, is limited. In such situations, a simplified graph representation is often needed, an approach sometimes termed as *ontology projection*[11]. DeepOnto has implemented the algorithm (found in `OntologyProjector`) originally used in the ontology visualisation system OptiqueVQS [33], to transform axioms into a set of simplified RDF triples. Briefly, a concept subsumption axiom $C \sqsubseteq D$ is transformed into $\langle C, rdfs{:}subClassOf, D \rangle$, an individual membership axiom $D(d)$ is transformed into $\langle d, rdf{:}type, D \rangle$, a role assertion axiom $r(a, b)$ is transformed into $\langle a, r, b \rangle$, a restriction axiom in the form of $C \sqsubseteq \exists r.D$ or $C \sqsubseteq \forall r.D$ is transformed into $\langle C, r, D \rangle$. Here, C and D are concepts; a, b, and d denote individuals.

4. Tools and Resources

DeepOnto has implemented several tools and resources for various ontology engineering purposes. For ontology matching (OM), DeepOnto has BERTMap [3] for concept equivalence matching, BERTSubs (Inter) [7] for concept subsumption matching, and Bio-ML [27], a collection of biomedical datasets and evaluation protocols to support OM system benchmarking. For ontology completion, DeepOnto has BERTSubs (Intra), and the prompt-based approach proposed in OntoLAMA [28]. The work of OntoLAMA also involves a collection of subsumption inference datasets for language model probing. In the following sub-sections, we present each of these modules in more detail.

4.1. BERTMap

Ontology matching (OM) is the task of identifying mappings that represent a semantic relationship between entities of two different ontologies. BERTMap[12] targets on equivalence matching between named concepts. It adopts Bidirectional Encoder Representations from Transformers (BERT) [34], a masked language model pre-trained on extensive text corpora such as English Wikipedia, in the computation of a mapping score. Specifically, BERTMap utilises concept labels available in the input ontologies to extract pairs of synonyms and non-synonyms, and then fine-tunes a BERT model for synonym classification. The mappings score is determined by the aggregation of synonym scores between labels of two named concepts. To reduce the time complexity of mapping search, BERTMap

---

[9]https://owlcs.github.io/owlapi/apidocs_5/org/semanticweb/owlapi/reasoner/OWLReasoner.html
[10]https://www.w3.org/TR/owl2-mapping-to-rdf/
[11]Note that ontology taxonomy can be seen as a special case of projection, but we separate the functionalities into different modules because the former is purely for hierarchy and the latter is for triples.
[12]BERTMap tutorial: https://krr-oxford.github.io/DeepOnto/bertmap/

adopts a sub-word inverted index for candidate selection. This approach takes advantage of the sub-word tokenisation capability inherent in the BERT model. To enrich the mapping set and capture potential missed matches during candidate selection, BERTMap incorporates an iterative mapping extension algorithm based on the locality principle, i.e., if two concepts are matched, their parents or children are likely to be matched. Lastly, BERTMap utilises the mapping repair module proposed in [2] to remove a minimal set of inconsistent mappings.

BERTMap provides flexible configurations to accommodate various needs. For instance, users can easily switch between different masked language models, such as RoBERTa [35] and ALBERT [36], or opt for BERT variants pre-trained on specialised corpora, such as BioBERT [37] and ClinicalBERT [38], by simply altering the input name of the language model. Moreover, BERTMap supports both unsupervised and semi-supervised settings; the former relies solely on the input ontologies, while the latter can leverage a small number of provided mappings for enhanced performance. Data augmentation from external ontologies is also possible and would be very helpful when the input ontologies are short of concept labels. Last but not the least, a light-weight version of BERTMap called BERTMapLt. This version does not necessitate the BERT fine-tuning nor the mapping refinement, rendering it considerably more efficient. Despite its simplified operations, BERTMapLt can attain promising results on certain datasets.

4.2. BERTSubs

BERTSubs[13] [7] aims at predicting *(i)* the missing concept subsumptions within an OWL ontology for completion, and *(ii)* the subsumptions between concepts from two OWL ontologies for alignment. In our latest implementation in DeepOnto (since v0.7.0), the super-concepts in both situations can be either named concepts or complex concepts such as existential restrictions. Following the architecture of BERTMap, BERTSubs also fine-tunes a pre-trained BERT (or one of its variants) together with an attached binary classifier which outputs a score for an input candidate subsumption. It extracts the existing subsumptions within the given ontology or ontologies for constructing positive samples, and replaces the super-concepts of these subsumptions by a randomly selected named concept (or complex concept if the original super-concept is complex) for constructing negative samples.

Each concept in a subsumption is transformed into a text sentence, and the subsumption is transformed into a sentence pair as the model input. For a complex concept, we directly call the verbalisation function implemented in the ontology processing module of DeepOnto (see Section 3). For a named concept, we have implemented three approaches (see details in [7]) to generate its (context-aware) text sentence as the model input:

– **Isolated Class (IC)** which directly uses the given concept's name according to a pre-defined annotation property like *rdfs:label*;
– **Path Context (PC)** which extracts a subsumption path from the ontology's concept hierarchy, starting from the given super-concept up to the root (or starting from the given sub-concept down to a leaf), and concatenates the names of the concepts in the path, separated by some special token;
– **Breadth-first Context (BC)** which traverses a set of neighbouring subsumptions of the given concept from the ontology's concept hierarchy, starting from the given super-concept up to the root (or starting from the given sub-concept down to a leaf) via breadth-first search, and concatenates the names of the concepts of these subsumptions in a specific way, separated by a special token.

4.3. Bio-ML

Benchmarking has been a critical challenge that limits OM academic research and system development. Thus, we added Bio-ML into DeepOnto. It currently includes five OM datasets derived from biomedical ontologies, accompanied by a comprehensive OM evaluation framework. The involved biomedical ontologies include SNOMED-CT [39], FMA [40], NCIT [41], OMIM [42], ORDO [43], and DOID [44], based on the human-curated thesaurus and alignment in the integrated ontologies, UMLS [45] and Mondo [46]. It aims to address several limitations of the existing OM datasets and evaluation settings, including the following:

---

[13]BERTSubs tutorial: https://krr-oxford.github.io/DeepOnto/bertsubs/

	Ontology Pair	Category	#Concepts ($\equiv$)	#Refs ($\equiv$)	#Concepts ($\sqsubseteq$)	#Refs ($\sqsubseteq$)
Mondo	OMIM-ORDO	Disease	9,642-8,838	3,721	9,642-8,735	103
	NCIT-DOID	Disease	6,835-8,848	4,686	6,835-5,113	3,339
UMLS	SNOMED-FMA	Body	24,182-64,726	7,256	24,182-59,567	5,506
	SNOMED-NCIT	Pharm	16,045-15,250	5,803	16,045-12,462	4,225
	SNOMED-NCIT	Neoplas	11,271-13,956	3,804	11,271-13,790	213

Table 1

Data statistics for ontology pairs in Bio-ML, including the data sources, ontology names, categories (semantic types) of preserved concepts during ontology pruning, numbers of named concepts and reference mappings in the equivalence ($\equiv$) and the subsumption ($\sqsubseteq$) settings.

– **Sub-optimal ground truth mappings**: Many OM datasets have incomplete ground truth mappings, and conventional evaluation metrics Precision, Recall, and F-score become biased towards high-precision and low-recall systems, as they are more likely to match the available ground truth mappings.
– **Limited to equivalence matching**: The majority of existing OM datasets focus solely on matching equivalent concepts.
– **Lack of support for machine learning-based systems**: Current OM datasets do not accommodate data splitting (e.g., training, validation, and testing) and the conventional evaluation metrics, which are based on the final output mappings, are inefficient for developing and debugging machine learning-based OM models.

To address these limitations, Bio-ML incorporates the following strategies:

– **Human-curated mappings**: Bio-ML utilises ground truth mappings from UMLS and Mondo, which have been curated and validated by experts.
– **Expanded evaluation metrics**: In addition to conventional metrics, Bio-ML proposes the use of ranking metrics such as Hits@K and MRR to provide a more comprehensive assessment of OM systems and support more efficient evaluation of machine learning-based OM systems.
– **Subsumption matching**: Bio-ML extends the scope of matching beyond equivalence, including subsumption relationships; the subsumption mappings are generated from the ground truth equivalence mappings.
– **Data split settings**: Bio-ML formulates different data split configurations (unsupervised and semi-supervised) to better support the development and evaluation of machine learning-based OM systems.

Furthermore, given that biomedical ontologies are often large-scale and some OM systems cannot deal with such ontologies or will cost a very long time for computation, Bio-ML employs a pruning algorithm (see Ontology Pruning in Section 3) subject to semantic types, which allows for the extraction of scalable sub-ontologies. For subsumption matching, target concepts that appear in a ground truth equivalence mapping used to construct a ground truth subsumption mapping is purposely deleted in order to enforce direct subsumption matching. The resulting datasets can be downloaded through Zenodo[14], the detailed instructions and data statistics (also shown in Table 1) are provided in DeepOnto[15]. Moreover, Bio-ML has been introduced as a new track[16] of the Ontology Alignment Evaluation Initiative (OAEI 2022), and the corresponding result report is available [47]. We will discuss more about its usage in the OAEI in Section 5.2.

4.4. OntoLAMA

The investigation of a language model's comprehension of knowledge and reasoning capability is a widely discussed topic, commonly referred to as "LMs-as-KBs" [48]. While most existing works on 'LMs-as-KBs' focus on knowledge graphs composed of relational facts (sometimes known as RDF triple knowledge graphs), OntoLAMA explores OWL ontologies which represent conceptual knowledge with logical reasoning supported. OntoLAMA is

---

[14]Bio-ML dataset download: https://zenodo.org/record/6946466
[15]Bio-ML instructions: https://krr-oxford.github.io/DeepOnto/bio-ml/
[16]Bio-ML Track of the OAEI: https://www.cs.ox.ac.uk/isg/projects/ConCur/oaei/

System	With out-of-scope concepts			Without out-of-scope concepts		
	Precision	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**
BERTMap						
$\lambda = 0.995$	0.805	0.883	0.842	0.837	0.883	0.859
$\lambda = 0.999$	0.922	0.820	**0.868**	0.937	0.820	**0.874**
Sub-string Match	0.965	0.489	0.649	0.965	0.489	0.649

Table 2

Human evaluation results based on consensus of three domain experts for matching medical concepts from NHS conditions to DOID concepts. Since DOID is mainly about diseases, non-disease concepts are considered as **out-of-scope**.

essentially a set of subsumption inference (SI) datasets concerning both atomic and complex concept expressions. These datasets are extracted from ontologies of various domains and scales. The probing method proposed in OntoLAMA is based on prompt learning, which is a paradigm used to effectively extract knowledge from language models through prompts. In this method, concept expressions are first transformed into natural language text using the recursive ontology verbaliser (see Ontology Verbalisation in Section 3). Then, the verbalised phrases are wrapped into a template along with the prompt text. The language model's task is to classify if two concept expressions have a subsumption relationship given this prompt-enhanced text format. Using this approach, a significant and consistent improvement in performance has been observed even with a small number of training and development samples (in a few-shot setting). This outcome highlights the potential of LM-based ontology engineering works in the future, without the need for excessive training resources.

To summarise, OntoLAMA contributes the following to DeepOnto: *(i)* a set of LM probing datasets[17] extracted from ontology subsumptions, *(ii)* a verbalisation algorithm that can handle complex concept expressions, *(iii)* a method to use LMs with prompts to predict subsumptions without fully supervised fine-tuning.

5. Impact and Use

DeepOnto has been gaining attention from both the industry and academia. On the industrial front, notable adoptions of DeepOnto include its utilisation by Samsung Research UK for Digital Health Coaching[18] and Madrid Digital[19], where it played a crucial role in their Proof of Concept process. Additionally, DeepOnto is actively being integrated into various ongoing projects, details of which remain confidential due to anonymity clauses. Our user feedback indicates DeepOnto's primary applications within the health and biomedical sectors. From the academia side, DeepOnto contributes to the Bio-ML track of the Ontology Alignment Evaluation Initiative (OAEI), and our tools and resources have facilitated numerous research works, evident in existing scholarly citations. For example, BERTMap serves as a typical baseline in subsequent BERT-based ontology alignment studies, including but not limited to Truveta Mapper [49], LaKERMap [50], as well as in large language model-based frameworks such as Retrieve-Rank [51] and LLMap [52]. In the following sub-sections, we elaborate on two use cases of DeepOnto, i.e., Digital Health Coaching in Samsung Research UK and the Bio-ML track of the Ontology Alignment Evaluation Initiative (OAEI).

5.1. Digital Health Coaching

Digital Health Coaching is commonly required in health management. A paramount priority is centred on the delivery of **context-sensitive, personalised, and explainable health recommendations**, aiming to aid end-users

---

[17]OntoLAMA dataset download is available at Zenodo: https://zenodo.org/record/7700458, and Huggingface: https://huggingface.co/datasets/krr-oxford/OntoLAMA.

[18]https://www.comunidad.madrid/en/servicios/sede-electronica/madrid-digital

[19]https://www.comunidad.madrid/en/servicios/sede-electronica/madrid-digital

in understanding and ameliorating their health conditions. Consider, for instance, individuals afflicted with Gastro-oesophageal Reflux Disease who are utilising Amitriptyline to alleviate back pain. The responsibility lies with the Digital Health Coach to understand that the use of Amitriptyline requires the simultaneous administration of another medication to safeguard the stomach, such as Antacids or Duloxetine. Therefore, it should deliver a recommendation that includes an explanation, such as *"Given that Amitriptyline can exacerbate reflux issues, it is recommended to discuss your stomach condition with your doctor as preventative measures might be necessary."*

In health scenarios where explainability and in-depth reasoning are prioritised, fully automatic Deep Learning algorithms prove inappropriate due to inherent constraints. These include a dependency on vast datasets for optimal performance, an inability to reason abstractly, and a nature that is largely inscrutable to human comprehension. A rising trend to alleviate these issues involves integrating a symbolic knowledge base (e.g., OWL ontology) that facilitates rigorous and complex reasoning into the process. One of the tasks involves automatically identifying entities from semi-structured and/or unstructured data, and subsequently mapping them to the knowledge base. The BERTMap model, as discussed in Section 4.1, is utilised and evaluated in aligning health and/or medical concepts extracted from the NHS conditions[20] to concepts in the DOID ontology[21].

The NHS conditions (at the time of investigation) consist of 984 concepts that cover health conditions, symptoms, medical treatments/tools, possible causes, and related daily-life situations. Each concept is associated with a web-page of detailed information. As the NHS conditions are essentially a collection of web-pages, we conducted simple transformation over them into a *"flat"* ontology, i.e., each term is modelled as an OWL ontology concept but there is no subsumption relationship among them. The concept names and aliases were manually collected from the web-pages to serve as concept labels. The DOID ontology has 10,924 concepts for the sake of providing consistency, reusability, and sustainability of descriptions of human disease terms. As the NHS conditions are not all about diseases, some of the concepts are considered as *"out-of-scope"* (i.e., non-diseases). We then considered task settings with and without the out-of-scope concepts. Since there are no ground truth mappings available, the alignment results were evaluated by human curators (see Table 2) because there are no existing ground truth mappings available. From the results we can observe that BERTMap attains a promising F-score and a consistently better performance than the Sub-string Match baseline model, which considers two concepts *C* and *D* as matched if a label of *C* is a sub-string of a label of *D*, or vice versa. λ denotes the threshold for filtering BERTMap's output mappings, i.e., only mappings with scores $\geqslant \lambda$ will be preserved. Sliding λ is essentially a trade-off of Precision and Recall. BERTMap achieves an F-score of 0.868 if considering out-of-scope concepts and an F-score of 0.874 if not. This slight difference indicates that BERTMap wrongly matches some of the non-disease concepts to the disease concepts. For example, the drug concept *Asprin* is wrongly mapped to the disease concept *Aspirin-induced Respiratory Disease*. A potential reason for such mismatches is that BERTMap relies on sufficient ontology concept labels for high performance – but only DOID provides multiple labels for each concept. Although the overall efficacy of BERTMap is satisfying according to the project coordinator, these mismatches underscore the need for further improvement by, e.g., utilising pertinent contexts to disambiguate the concepts.

5.2. Ontology Alignment Evaluation Initiative

The Ontology Alignment Evaluation Initiative (OAEI) is an internationally coordinated initiative aiming at performing systematic assessments of ontology alignment techniques, also known as ontology matching (OM). As part of the OAEI 2022, we have introduced a new track, Bio-ML, replacing the previous LargeBio track. As detailed in Section 4.3, the objective of Bio-ML is to furnish datasets and a comprehensive framework for the evaluation of both traditional and machine learning-based OM systems.

The OAEI 2022 version of Bio-ML comprises two ontology pairs derived from Mondo and three pairs from UMLS (see data statistics in Table 1). Every pair is linked with two types of matching: **equivalence** and **subsumption**. The equivalence matching incorporates both **global matching** (evaluated using traditional metrics like Precision, Recall, and F-score) and **local ranking** (assessed using ranking metrics such as MRR and Hits@K). The

---

	Unsupervised (90% Test Mappings)					Semi-supervised (70% Test Mappings)				
System	**Precision**	**Recall**	**F-score**	**MRR**	**Hits@1**	**Precision**	**Recall**	**F-score**	**MRR**	**Hits@1**
LogMap	0.827	0.498	0.622	0.803	0.742	0.783	0.547	0.644	0.821	0.743
LogMap-Lite	0.935	0.259	0.405	-	-	0.932	0.519	0.667	-	-
AMD	0.664	0.565	0.611	-	-	0.792	0.528	0.633	-	-
BERTMap	0.730	0.572	0.641	0.873	0.817	0.575	0.784	0.664	0.965	0.947
BERTMapLt	0.819	0.499	0.620	0.776	0.729	0.775	0.713	0.743	0.900	0.876
Matcha	0.743	0.508	0.604	-	-	0.704	0.564	0.626	-	-
Matcha-DL	-	-	-	-	-	0.956	0.615	0.748	0.654	0.640
ATMatcher	0.940	0.247	0.391	-	-	0.835	0.286	0.426	-	-
LSMatch	0.650	0.221	0.329	-	-	0.877	0.238	0.374	-	-

Table 3

Equivalence matching results for OMIM-ORDO (Disease) in the Bio-ML track of OAEI 2022.

	Unsupervised (90% Test Mappings)				Semi-supervised (70% Test Mappings)			
System	MRR	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10
Word2Vec+RF	0.512	0.368	0.694	0.834	0.577	0.433	0.773	0.880
OWL2Vec*+RF	0.603	0.461	0.782	0.860	0.666	0.547	0.827	0.880
BERTSubs (IC)	0.530	0.333	0.786	0.948	0.638	0.463	0.859	0.953

Table 4

Subsumption matching results for SNOMED-NCIT (Neoplas) in Bio-ML track of OAEI 2022.

subsumption matching, due to its inherent incompleteness of ground truth mappings, only has the local ranking setting. Moreover, for each matching setup, there are two settings for data split: **unsupervised** and **semi-supervised**. The unsupervised setting does not provide any training mapping, while the semi-supervised setup provides a small partition designated for training and development. Note that DeepOnto contributes to the construction of Bio-ML, the implementation of BERTMap and BERTMapLt systems, and the evaluation workaround.

The results for OMIM-ORDO equivalence matching and SNOMED-NCIT (Neoplas) subsumption matching are illustrated in Table 3 and Table 4, respectively. Full results can be accessed on the Bio-ML (OAEI 2022) webpage[22]. The findings from the Bio-ML track can be summarised as: *(i)* ML-based systems generally outperform others; *(ii)* no single system leads in all tasks, indicating that the effectiveness of systems varies based on the specific task; *(iii)* traditional OM systems struggle with the subsumption matching task, primarily due to their dependency on lexical similarity, which is very useful in equivalence matching but not that much in subsumption matching.

6. Conclusion and Future Work

In this system paper, we introduce DeepOnto, a Python library designed to facilitate ontology engineering with deep learning. The package provides a broad spectrum of ontology engineering capabilities, enabling users to (semi-) automatically and efficiently deal with ontologies and develop novel systems. The library relies on Python programming in synergy with deep learning methodologies, with a particular emphasis on pre-trained language models. For ease of use, DeepOnto has encapsulated basic ontology processing functions from OWL API, and implemented several essential components such as reasoning, verbalisation, pruning, taxonomy, and projection. Based on these features, DeepOnto has integrated a range of tools and resources for different ontology engineering tasks such as ontology alignment and completion, with comprehensive tutorials and detailed instructions[23] provided. Furthermore,

---

[22]Bio-ML Track (OAEI 2022) Results: https://www.cs.ox.ac.uk/isg/projects/ConCur/oaei/2022/index.html#results

[23]See the "TUTORIALS" section at https://krr-oxford.github.io/DeepOnto/.

we show evidence of DeepOnto's successful applications in both industry and academia, with promising results reported in different application contexts.

For future development, we aim to incorporate more automated ontology engineering tasks. These include, but are not limited to, embedding concepts and properties with both formal semantics and literals considered, placing new concepts derived from text mentions into an ontology, generating descriptions for concepts based on their contexts in an ontology, and identifying, generating, and placing appropriate common parents for concepts clustered under certain criteria. These enhancements will lead to more ontology processing requirements, all of which will be seamlessly encapsulated into our API.

7. Acknowledgements

References

[1] S. Staab and R. Studer, *Handbook on ontologies*, Springer Science & Business Media, 2010.

[2] E. Jiménez-Ruiz and B. Cuenca Grau, Logmap: Logic-based and scalable ontology matching, in: *The Semantic Web–ISWC 2011: 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I 10*, Springer, 2011, pp. 273–288.

[3] Y. He, J. Chen, D. Antonyrajah and I. Horrocks, BERTMap: a BERT-based ontology alignment system, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36, 2022, pp. 5684–5691.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser and I. Polosukhin, Attention is all you need, *Advances in neural information processing systems* 30 (2017).

[5] B. Glimm, I. Horrocks, B. Motik, G. Stoilos and Z. Wang, HermiT: an OWL 2 reasoner, *Journal of Automated Reasoning* 53 (2014), 245–269.

[6] N. Li, Z. Bouraoui and S. Schockaert, Ontology completion using graph convolutional networks, in: *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part I 18*, Springer, 2019, pp. 435–452.

[7] J. Chen, Y. He, Y. Geng, E. Jiménez-Ruiz, H. Dong and I. Horrocks, Contextual semantic embeddings for ontology subsumption prediction, *World Wide Web* (2023), 1–23.

[8] J. Chen, P. Hu, E. Jimenez-Ruiz, O.M. Holter, D. Antonyrajah and I. Horrocks, Owl2vec*: Embedding of owl ontologies, *Machine Learning* 110(7) (2021), 1813–1845.

[9] E. Choi, M.T. Bahadori, L. Song, W.F. Stewart and J. Sun, GRAM: graph-based attention model for healthcare representation learning, in: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 787–795.

[10] H. Rashkin, V. Nikolaev, M. Lamm, L. Aroyo, M. Collins, D. Das, S. Petrov, G.S. Tomar, I. Turc and D. Reitter, Measuring attribution in natural language generation models, *Computational Linguistics* (2023), 1–64.

[11] B. Bohnet, V.Q. Tran, P. Verga, R. Aharoni, D. Andor, L.B. Soares, J. Eisenstein, K. Ganchev, J. Herzig, K. Hui et al., Attributed question answering: Evaluation and modeling for attributed large language models, *arXiv preprint arXiv:2212.08037* (2022).

[12] M. Horridge and S. Bechhofer, The owl api: A java api for owl ontologies, *Semantic web* 2(1) (2011), 11–21.

[13] J.J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne and K. Wilkinson, Jena: implementing the semantic web recommendations, in: *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, 2004, pp. 74–83.

[14] B. Motik, B.C. Grau, I. Horrocks, Z. Wu, A. Fokoue, C. Lutz et al., OWL 2 web ontology language profiles, *W3C recommendation* 27(61) (2009).

[15] M. Abadi, TensorFlow: learning functions at scale, in: *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, 2016, pp. 1–1.

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., Pytorch: An imperative style, high-performance deep learning library, *Advances in neural information processing systems* 32 (2019).

[17] F. Zhapa-Camacho, M. Kulmanov and R. Hoehndorf, mOWL: Python library for machine learning with biomedical ontologies, *Bioinformatics* 39(1) (2023), btac811.

[18] M. Jackermeier, J. Chen and I. Horrocks, Box^2 EL: Concept and Role Box Embeddings for the Description Logic EL++, *arXiv preprint arXiv:2301.11118* (2023).

[19] H. Dong, J. Chen, Y. He, Y. Liu and I. Horrocks, Reveal the Unknown: Out-of-Knowledge-Base Mention Discovery with Entity Linking, *arXiv preprint arXiv:2302.07189* (2023).

[20] M.A. Musen, The protégé project: a look back and a look forward, *AI matters* 1(4) (2015), 4–12.

[21] R.C. Jackson, J.P. Balhoff, E. Douglass, N.L. Harris, C.J. Mungall and J.A. Overton, ROBOT: a tool for automating ontology workflows, *BMC bioinformatics* 20 (2019), 1–10.

[22] J.-B. Lamy, Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies, *Artificial intelligence in medicine* **80** (2017), 11–28.

[23] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz et al., Huggingface's transformers: State-of-the-art natural language processing, *arXiv preprint arXiv:1910.03771* (2019).

[24] N. Ding, S. Hu, W. Zhao, Y. Chen, Z. Liu, H.-T. Zheng and M. Sun, Openprompt: An open-source framework for prompt-learning, *arXiv preprint arXiv:2111.01998* (2021).

[25] OpenAI, GPT-4 Technical Report, *ArXiv* **abs/2303.08774** (2023). https://api.semanticscholar.org/CorpusID:257532815.

[26] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale et al., Llama 2: Open foundation and fine-tuned chat models, *arXiv preprint arXiv:2307.09288* (2023).

[27] Y. He, J. Chen, H. Dong, E. Jiménez-Ruiz, A. Hadian and I. Horrocks, Machine Learning-Friendly Biomedical Datasets for Equivalence and Subsumption Ontology Matching, in: *The Semantic Web–ISWC 2022: 21st International Semantic Web Conference, Virtual Event, October 23–27, 2022, Proceedings*, Springer, 2022, pp. 575–591.

[28] Y. He, J. Chen, E. Jiménez-Ruiz, H. Dong and I. Horrocks, Language Model Analysis for Ontology Subsumption Inference, *arXiv preprint arXiv:2302.06761* (2023).

[29] Y. Kazakov, M. Krötzsch and F. Simančík, ELK: A Reasoner for OWL EL Ontologies, System Description, University of Oxford, 2012, available from http://code.google.com/p/elk-reasoner/wiki/Publications.

[30] M. d'Aquin, A. Schlicht, H. Stuckenschmidt and M. Sabou, Ontology modularization for knowledge selection: Experiments and evaluations, in: *Database and Expert Systems Applications: 18th International Conference, DEXA 2007, Regensburg, Germany, September 3-7, 2007. Proceedings 18*, Springer, 2007, pp. 874–883.

[31] F. Baader, S. Brandt and C. Lutz, Pushing the EL envelope, in: *Proceedings of the 19th international joint conference on Artificial intelligence*, 2005, pp. 364–369.

[32] M. Kulmanov, W. Liu-Wei, Y. Yan and R. Hoehndorf, El embeddings: Geometric construction of models for the description logic el++, *arXiv preprint arXiv:1902.10499* (2019).

[33] A. Soylu, E. Kharlamov, D. Zheleznyakov, E. Jimenez-Ruiz, M. Giese, M.G. Skjæveland, D. Hovland, R. Schlatte, S. Brandt, H. Lie et al., OptiqueVQS: A visual query system over ontologies for industry, *Semantic Web* **9**(5) (2018), 627–660.

[34] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv:1810.04805* (2018).

[35] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, *arXiv preprint arXiv:1907.11692* (2019).

[36] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma and R. Soricut, Albert: A lite bert for self-supervised learning of language representations, *arXiv preprint arXiv:1909.11942* (2019).

[37] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C.H. So and J. Kang, BioBERT: a pre-trained biomedical language representation model for biomedical text mining, *Bioinformatics* **36**(4) (2020), 1234–1240.

[38] K. Huang, J. Altosaar and R. Ranganath, Clinicalbert: Modeling clinical notes and predicting hospital readmission, *arXiv preprint arXiv:1904.05342* (2019).

[39] K. Donnelly et al., SNOMED-CT: The advanced terminology and coding system for eHealth, *Studies in health technology and informatics* **121** (2006), 279.

[40] C. Rosse and J.L. Mejino Jr, The foundational model of anatomy ontology, in: *Anatomy ontologies for bioinformatics: principles and practice*, Springer, 2008, pp. 59–117.

[41] N. Sioutos, S. de Coronado, M.W. Haber, F.W. Hartel, W.-L. Shaiu and L.W. Wright, NCI Thesaurus: a semantic model integrating cancer-related clinical and molecular information, *Journal of biomedical informatics* **40**(1) (2007), 30–43.

[42] J.S. Amberger, C.A. Bocchini, F. Schiettecatte, A.F. Scott and A. Hamosh, OMIM. org: Online Mendelian Inheritance in Man (OMIM®), an online catalog of human genes and genetic disorders, *Nucleic acids research* **43**(D1) (2015), D789–D798.

[43] D. Vasant, L. Chanas, J. Malone, M. Hanauer, A. Olry, S. Jupp, P.N. Robinson, H. Parkinson and A. Rath, Ordo: an ontology connecting rare disease, epidemiology and genetic data, in: *Proceedings of ISMB*, Vol. 30, researchgate. net, 2014.

[44] L.M. Schriml, C. Arze, S. Nadendla, Y.-W.W. Chang, M. Mazaitis, V. Felix, G. Feng and W. Kibbe, Disease Ontology: a backbone for disease semantic integration, *Nucleic Acids Research* **40** (2012), D940–D946.

[45] O. Bodenreider, The unified medical language system (UMLS): integrating biomedical terminology, *Nucleic acids research* **32**(suppl_1) (2004), D267–D270.

[46] N. Vasilevsky, S. Essaid, N. Matentzoglu, N.L. Harris, M. Haendel, P. Robinson and C.J. Mungall, Mondo Disease Ontology: harmonizing disease concepts across the world, in: *CEUR Workshop Proceedings, CEUR-WS*, Vol. 2807, 2020.

[47] M.A.N. Pour, A. Algergawy, P. Buche, L.J. Castro, J. Chen, H. Dong, O. Fallatah, D. Faria, I. Fundulaki, S. Hertling, Y. He, I. Horrocks, M. Huschka, L. Ibanescu, E. Jiménez-Ruiz, N. Karam, A. Laadhar, P. Lambrix, H. Li, Y. Li, F. Michel, E. Nasr, H. Paulheim, C. Pesquita, T. Saveta, P. Shvaiko, C. Trojahn, C. Verhey, M. Wu, B. Yaman, O. Zamazal and L. Zhou, Results of the Ontology Alignment Evaluation Initiative 2022, in: *OM@ISWC*, CEUR Workshop Proceedings, Vol. 3324, CEUR-WS.org, 2022, pp. 84–128.

[48] F. Petroni, T. Rocktäschel, S. Riedel, P. Lewis, A. Bakhtin, Y. Wu and A. Miller, Language Models as Knowledge Bases?, in: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 2463–2473. doi:10.18653/v1/D19-1250. https://aclanthology.org/D19-1250.

[49] M. Amir, M. Baruah, M. Eslamialishah, S. Ehsani, A. Bahramali, S. Naddaf-Sh and S. Zarandioon, Truveta Mapper: A Zero-shot Ontology Alignment Framework, *arXiv preprint arXiv:2301.09767* (2023).

[50] Z. Wang, Contextualized Structural Self-supervised Learning for Ontology Matching, *arXiv preprint arXiv:2310.03840* (2023).

[51] Q. Wang, Z. Gao and R. Xu, Exploring the In-context Learning Ability of Large Language Model for Biomedical Concept Linking, *arXiv preprint arXiv:2307.01137* (2023).

[52] Y. He, J. Chen, H. Dong and I. Horrocks, Exploring Large Language Models for Ontology Alignment, *arXiv preprint arXiv:2309.07172* (2023).

[53] G. Alterovitz, M. Xiang, D.P. Hill, J. Lomax, J. Liu, M. Cherkassky, J. Dreyfuss, C. Mungall, M.A. Harris, M.E. Dolan et al., Ontology engineering, *Nature biotechnology* **28**(2) (2010), 128–130.

[54] A. Gulli and S. Pal, *Deep learning with Keras*, Packt Publishing Ltd, 2017.

[55] T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell et al., Language models are few-shot learners, *Advances in neural information processing systems* **33** (2020), 1877–1901.

[56] J.H. Caufield, H. Hegde, V. Emonet, N.L. Harris, M.P. Joachimiak, N. Matentzoglu, H. Kim, S.A. Moxon, J.T. Reese, M.A. Haendel et al., Structured prompt interrogation and recursive extraction of semantics (SPIRES): A method for populating knowledge bases using zero-shot learning, *arXiv preprint arXiv:2304.02711* (2023).

[57] R. Anil, A.M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen et al., Palm 2 technical report, *arXiv preprint arXiv:2305.10403* (2023).

[58] S. Suresh, K. Mukherjee and T.T. Rogers, Semantic Feature Verification in FLAN-T5, *arXiv preprint arXiv:2304.05591* (2023).