DAIKIRI

Diagnostische KI für industrielle Daten

DAIKIRI
Erklärbare Diagnostische KI für industrielle Daten

Project Number: 01IS19085B **Start Date of Project:** 01/01/2020 **Duration:** 24 months

Deliverable 3.2/3.3

Semantification Service for Industry Data

Dissemination Level	Public
Due Date of Deliverable	Month 24, 31/12/2021
Actual Submission Date	Month 24, 31/12/2021
Work Package	WP3 — Semantification
Tasks	T3.3, T3.4, T3.5, T3.6
Type	Report
Approval Status	Final
Version	1.0
Number of Pages	15

The information in this document reflects only the author's views, and the Federal Ministry of Education and Research (BMBF) is not liable for any use that may be made of the information contained therein. The information in this document is provided "as is" without guarantee or warranty of any kind, express or implied, including but not limited to the fitness of the information for a particular purpose. The user thereof uses the information at his/ her sole risk and liability.

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

This project has received funding from the Federal Ministry of Education and Research (BMBF) within the project DAIKIRI under the grant no 01IS19085B.

History

Version	Date	Reason	Revised by
1.0	31/12/2021	Final version created	Hamada Zahera

Author List

Organization	Name	Contact Information
UPB	Hamada Zahera	hamada.zahera@uni-paderborn.de
UPB	Stefan Heindorf	heindorf@uni-paderborn.de

Contents

1 Introduction

Today, tabular data is the most common data type used in several applications, including medicine, finance, manufacturing, climate science, and many other applications that are based on relational databases. This data is commonly represented in tabular format –comprising samples (rows) with the same set of features (columns)–, but lacks semantic information. As a result, it becomes challenging to understand the meaning of the data and integrate it with other resources. On the other hand, representing the data as linked open data—a.k.a *semantification* [Furth and Baumeister, 2013]—is crucial for integration of different data sources (e.g., entities linking) and explainable predictive tasks (e.g., anomaly detection). This process is often performed by means of numerous hand-crafted scripts and requiring expensive maintenance by IT service providers. Recently, embedding-based methods have demonstrated significant performances in several applications, for example word embeddings for natural language processing, image embeddings for object/face recognition, and entity embeddings for knowledge graphs (KGs) completion and node classification. In our deliverable 3.1, we have employed a knowledge graph embedding (KGE) and density-based clustering to identify similar entities and group them together into the same cluster. For example, Figure 1 shows how similar entities from FB15k-237 dataset are clustered together based on their representation from KG embedding.

Clustering approaches allow identifying data instances with similar characteristics, such as types of information, etc. Due to the lack of labelled data, their performances in inferring data types (labelling) are not as noteworthy as those of supervised approaches. Most existing approaches have shown successful application of cluster labelling to learn ontologies from textual data. Despite this, few studies addressed the challenges of labelling linked data (e.g., in RDF format <*subject, predicate, object*>) to learn ontologies, i.e., types information of <*subject*> [Zhao et al., 2020]. Although, these clustered entities are lacking information types and require additional labelling process. It may be possible to explore all entity properties and manually assign types. Nevertheless, this manual labelling task is a time-consuming and costly, which requires hiring many domain experts. To address this challenges, we propose two approaches: our first approach annotates a small subset of data (e.g., 100 data samples) by employing a human expert. After that, we propagate the major type from annotated entities to other entities within the same cluster. Toward this goal, we developed a web application that presents a set of sampled entities from each cluster to a human expert for annotation.

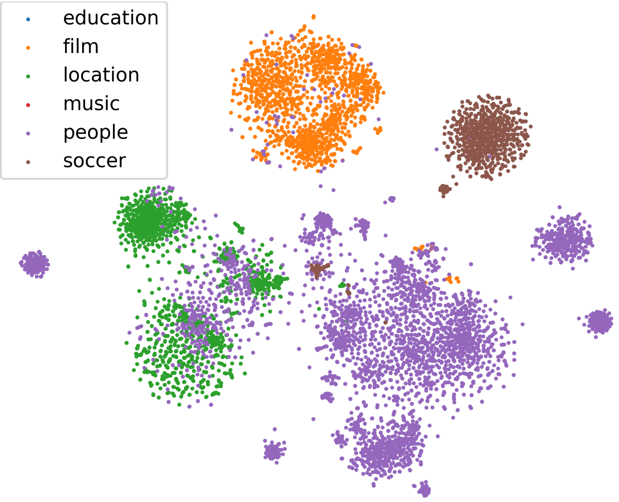


Figure 1: t-SNE plot of clustering entities based on their embedding representations in FB15k-237.

Our application shows entity properties (e.g., RDF triples) in a web interface and allows users to add *types* information (more details in Section 4.1). On the other hand, our second approach employs semi-supervised learning for labelling entities in the embedding spaces, i.e, entity typing task. In particular, our approach follows a teacher-student learning paradigm which employs learning from both massive unlabelled data and small labelled data [Zahera et al., 2021]. To verify the effectiveness of our approach, we conducted a set of experiments on two benchmark datasets for entity labelling tasks: FB15k-ET and YAGO43k-ET [Zhao et al., 2020]. Our experimental results demonstrate that our approach outperforms the state-of-the-art baselines in inferring entities types with a small labelled data.

The remainder of this report is organizing as follows: In Section 2, we discuss state-of-the-art techniques in labelling clusters, including textual and linked data. Section 3 presents the dataset used in our experiments, and Section 4 describes the details of our proposed approaches. In Section 5, we describe our approach for generating axioms for the annotated entities. Finally, Section 7 concludes the main findings in this report and shows the current challenges in high-dimensional data for future work.

2 Related Work

Cluster Labeling. data annotation brings benefits for users to analyse and understand the semantic structure. Last years, different approaches have been proposed for labelling textual data. Statistical approaches such as topic modelling (e.g., LDA) and word importance (e.g., TF-IDF) have been leveraged to extract labels from an input text, i.e., extractive labelling. However, these approaches fail to capture the semantics of linked data, where data are represented as RDF triples (*subject, predicate, object*). [Carmel et al., 2009] proposed enhancing data labelling using Wikipedia information. In particular, their approach extracts a set of candidate labels from Wikipedia in addition to important terms from input text. Aker et al. [2016] proposed a graph-based approach for labelling users comments on an online news platform. By means of graph modelling, the proposed approach demonstrated a significant performance in labelling comments using DBpedia concepts. Similarly, Hulpus et al. [2013] extracted most frequent words from text and link with DBpedia concepts. By using a graph centrality measure, the proposed approach identified efficiently the DBpedia concepts that best label topics in a document. Ajwani et al. [2018] proposed a multimodel framework based on a small set of features and Wikipedia taxonomies for labelling unstructured contents (text and image) with a small set of training data. Nevertheless, these approaches demonstrated superior performance when dealing with textual data. They are not investigated yet for labelling linked data. Moreover, processing tabular data is a challenging task, due to the lack of semantics, including poorly defined column names, their meaning, and their content. To the extent of our knowledge, this study is the first attempt that address learns ontologies from tabular data based on clustering and knowledge graph embedding.

Cluster Labeling in Knowledge Graphs. Entity typing approaches in knowledge graphs can be distinguished by their features and models: For instance, [Melo et al., 2017] employ the incoming and outgoing relations of an entity to train a hierarchical multi-label classifier. Similarly, [Xu et al., 2016] predict the types of Chinese entities by linking them to the English DBpedia and employing DBpedia's property and category information within a multi-label hierarchical classifier. [Neelakantan and Chang, 2015] employ textual descriptions of entities as features in combination with a linear classifier. While there has been a lot of research on knowledge graph embeddings and link prediction [Wang et al., 2017], most of the approaches focus on predicting non-type links, and they employed benchmarking datasets, e.g., FB15k-237, WN18RR, and YAGO3-10 do not contain type information. Only [Moon et al., 2017] and [Zhao et al., 2020] proposed approaches for embedding entities according to their types. All of these approaches assume many labelled training samples, which might not be available and expensive to obtain. In contrast, we employ a semi-supervised approach, requiring only a little training data.

3 Datasets

Benchmarking Datasets for Entity Labeling In this deliverable, we experiment with two benchmarking datasets in inferring missing types of entities in knowledge graphs. In particular, we employ FB15k-ET and YAGO43k-ET, these datasets are enriched versions with types information from the original dataset FB15k-237 and YAGO43k, respectively. Table 1 gives a statistical overview of each dataset such as number of entities ($\#Ent.$), number of relations ($\#Rel.$), number of triples used in training ($\#Train$), validation ($\#Valid$), evaluation ($\#Test$) and number of types ($\#Types$). We summarize each dataset as follows:

- FB15k-ET [Zhao et al., 2020] consists of the benchmarking dataset FB15k-237 along with type triples of the form (*entity, entity type*). We use the same dataset split (*train-valid-test*) as [Zhao et al., 2020] to ensure the same evaluation setting. In particular, we use three subsets: train (136,618 triples), valid (15,749 triples) and test (15,780 triples).
- Similarly, YAGO43k-ET [Zhao et al., 2020] enriches the benchmarking dataset YAGO43k with type information, and we use the same data split: train (375,853 triples), valid (42,750 triples), test (45,182 triples). For both datasets, we determine the $k \in \{3, 5, 10\}$ most frequent types and employ the subset of entities and triples induced by them for our experiments.

Smart Logistic Dataset (Use case): The use case for Smart Logistics stems from the logistics of small parts to the production line of two different customers, vehicle, and consumer goods assembling, each with one production line over about three years. The data contains time-based information about taking parts out or putting new parts in boxes, so-called *stock changes*. In addition, it comprises information when and how many new parts are ordered to replenish the boxes, so-called *orders*. The original Smart Logistics Dataset consists of about 72,000 entities with type *order* and about 2.9 million entities with type *stock change*.

Benmarking Dataset of SML: Lymphography dataset [Westphal et al., 2019] is one of the benchmarking dataset for structured machine learning tasks that contains structure knowledge about Lymphography diseases in OWL Language. We utilize the Lymphography dataset to assess the performance of our approach in generating Axioms. Further details can be found in Section 5.

Table 1: Statistics of datasets: FB15k-ET and YAGO43k-ET.

	FB15k-ET			**YAGO43k-ET**		
	Top 3	Top 5	Top 10	Top 3	Top 5	Top 10
Relations	1,345	1,345	1,345	37	37	37
Total Type Triples	22,849	26,184	29,058	29,528	32,193	35,225
Train Type Triples	12,748	13,726	14,376	24,078	25,792	27,201
Valid Type Triples	5,038	6,200	7,321	2,689	3,170	3,997
Test Type Triples	5,063	6,258	7,361	2,761	3,231	4,027

4 Labelling of Embedding Spaces

In the semantification process, we aim to automatically learn ontologies from tabular data. Our semantification approach consists of four steps, as show in Figure 2. First, we preprocess the tabular data to clean noisy and NULL values. After that, we use Vectograph library[1] to transform the input data to a knowledge graph. In the third step (c), we cluster similar data (i.e., entities) together in the same group. In this deliverable, we propose two approaches to label the entities based on their embedding representations and clustering outcome. In the following subsections, we describe the details of each approach for labelling entities based on their semantic representation.

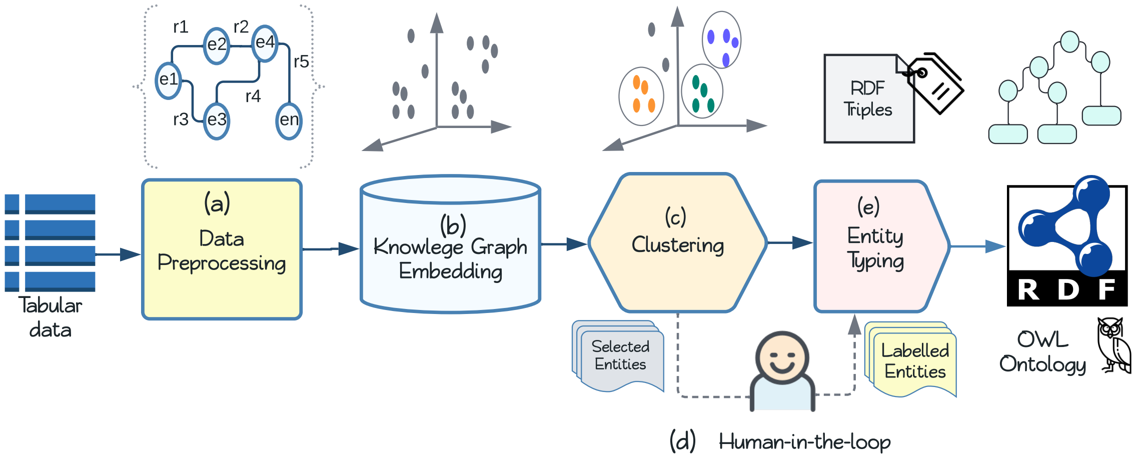


Figure 2: Semantification Pipeline, including Manual Labelling

4.1 Manual labelling (Demo)

We developed a web application (dubbed LabENT) for labelling entities manually based on their embedding representation. To do so, we sample entities from each cluster that are close to its centroid. Our demo presents a set of entities from the same cluster which have similar properties, including type information. As show in Figure 3, our demo presents the entities in an HTML table with information (*entity-ID* and *entity triples*), the user is asked to infer the entity type based on the triple's information. Finally, we propagate the majority type for all entities within the same cluster, as the cluster type information.

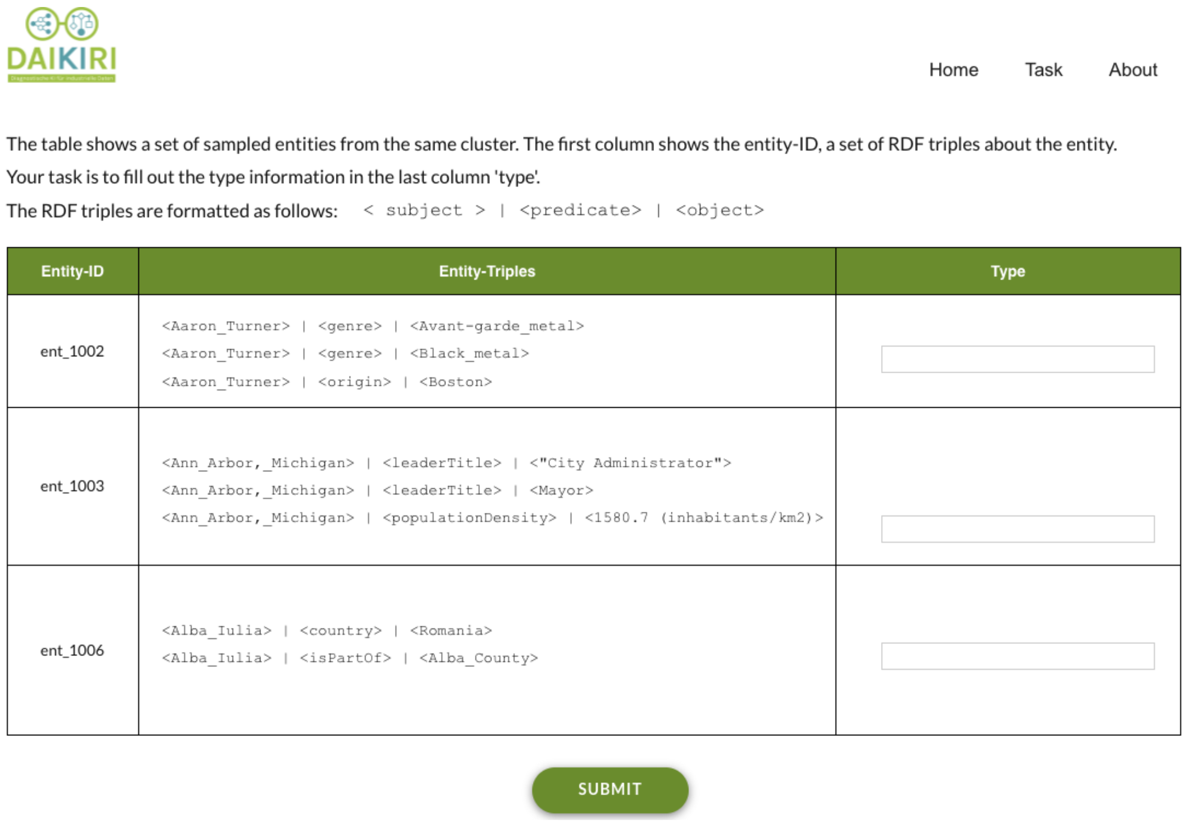
4.2 Semi-supervised labelling

While supervised approaches for entity type prediction have been proposed, e.g., based on hand-crafted features [Melo et al., 2017], knowledge graph embeddings [Zhao et al., 2020], and language models [Biswas et al., 2021], they all require a substantial amount of training data which is often not available. In contrast, unsupervised approaches based on clustering [Chen et al., 2019] do not require a priori labelled training data, but require labelling of clusters and do not reach the same predictive performance as supervised approaches.

In this section, we present our semi-supervised approach (dubbed ASSET)[2] to overcome this

[1] https://github.com/dice-group/vectograph

[2] Published in the K-CAP conference, December 2021



The table shows a set of sampled entities from the same cluster. The first column shows the entity-ID, a set of RDF triples about the entity.
Your task is to fill out the type information in the last column 'type'.
The RDF triples are formatted as follows: `< subject > | <predicate> | <object>`

Entity-ID	Entity-Triples	Type
ent_1002	<Aaron_Turner> | <genre> | <Avant-garde_metal> <Aaron_Turner> | <genre> | <Black_metal> <Aaron_Turner> | <origin> | <Boston>	
ent_1003	<Ann_Arbor,_Michigan> | <leaderTitle> | <"City Administrator"> <Ann_Arbor,_Michigan> | <leaderTitle> | <Mayor> <Ann_Arbor,_Michigan> | <populationDensity> | <1580.7 (inhabitants/km2)>	
ent_1006	<Alba_Iulia> | <country> | <Romania> <Alba_Iulia> | <isPartOf> | <Alba_County>	

Figure 3: a Screenshot of LabENT demo version 1.0 for labelling entities.

challenge and to close the gap, which require only little labelled training data. Our approach leverages unlabelled data using the teacher-student framework [van Engelen and Hoos, 2020, Lee et al., 2013]: (i) we train a teacher model on labelled data, (ii) we use the teacher model to generate *pseudo-labels* on unlabelled data, (iii) we train a student model on the combination of labelled and pseudo-labelled data. We repeat the process by treating the student as a new teacher to re-label the unlabelled data and to train a new student. To the best of our knowledge, our approach is the first to adapt semi-supervised learning to the entity typing task and distils knowledge from unlabelled data to boost its performance.

4.2.1 The Teacher-Student Framework

Our semi-supervised approach is based on self-training [van Engelen and Hoos, 2020] in the teacher-student framework. We formally describe the overall procedure in Algorithm 1. The inputs of our approach are labelled and unlabelled datasets $\mathscr{D}_l$ and $\mathscr{D}_u$, respectively, from which we sample batches. First, we train a teacher model $\mathcal{T}(\theta_t)$ only on labelled data $\mathscr{D}_l$ and compute the supervised loss $\mathcal{L}_{\mathscr{D}_l}$ using the cross-entropy function in Equation (1). Then, we employ the teacher model to generate *pseudo-labels* $\tilde{\mathbf{y}}_i$ for the unlabelled data $\mathscr{D}_u$ and we refer to the dataset along with pseudo-labels as $\tilde{\mathscr{D}}_u$. The pseudo-labels $\tilde{\mathbf{y}}_i^{(j)}$ are soft labels representing the probability of type λ_j being assigned to an entity e_i. Second, we train the student model $\mathcal{S}(\theta^s)$ on the combined dataset of *labelled* and *unlabelled* data to minimize the combined cross-entropy loss, as illustrated in Equation (2). Finally, we iterate this paradigm by replacing the teacher model $\mathcal{T}_i$ with the student model $\mathcal{S}_i$ to generate new pseudo-labels and train a new student $\mathcal{S}_{i+1}$.

Algorithm 1: Our Teacher-Student Algorithm

```
   Require: labeled and unlabeled datasets: $\mathscr{D}_l$, $\mathscr{D}_u$.
   Require: teacher and student models: $\mathcal{T}(\theta^t)$, $\mathcal{S}(\theta^s)$.
1  for num of epochs do
2      Sample batches $\beta_l$ from $\mathscr{D}_l = \{(x_i, y_i)\}, x_i \in \mathbb{R}^d$;
3      Train teacher model $\mathcal{T}_i(\theta^t)$ on $\beta_l$;
4      Calculate $\mathcal{L}_{\mathscr{D}_l}$ by cross-entropy function on $\beta_l$;
5  Use $\mathcal{T}_i(\theta^t)$ to infer pseudo-labels $\tilde{y}_i$ for $\mathscr{D}_u$ and let $\tilde{\mathscr{D}}_u$ be the dataset $\mathscr{D}_u$ together with
     pseudo-labels;
6  for num of epochs do
7      Sample batches $\beta_{l+u}$ from $\mathscr{D}_l$ and $\tilde{\mathscr{D}}_u$;
8      Train student model $\mathcal{S}_i(\theta^s)$ on $\beta_{l+u}$;
9      Calculate $\mathcal{L}_{\mathscr{D}_u}$ by a joint cross-entropy function on $\beta_{l+u}$;
10 Replace teacher model with student model $\mathcal{T}_{i+1} = \mathcal{S}_i$;
11 Repeat from Step 1 until student model $\mathcal{S}_{i+1}$ has converged.;
```

4.2.1.1 Teacher Model.

We use a neural network with one fully-connected layer with 128 units and *ReLU* activations, and an output layer with sigmoid activation. We train the teacher model for at most 100 epochs using the Adam optimizer. The model's hyperparameters are fine-tuned with grid search. To avoid over-fitting, we employ early-stopping [Yao et al., 2007]. The loss function is

$$\mathcal{L}_{\mathscr{D}_l} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}\left[y_i^{(j)}\log\left(\hat{y}_i^{(j)}\right)+\left(1-y_i^{(j)}\right)\log\left(1-\hat{y}_i^{(j)}\right)\right] \tag{1}$$

where y_i are the ground-truth types of e_i, $\hat{y}_i$ are the predicted types and N is the size of dataset $\mathscr{D}_l$.

4.2.1.2 Student Model.

We employ a network similar to the teacher model with an additional dropout layer with rate 0.20; we optimize the hyperparameters with grid search. We train the student model for 100 epochs on batches of labelled and pseudo-labelled data. Each batch has 128 samples, and we employ the Adam optimizer with early-stopping to avoid over-fitting. Our loss function is

$$\mathcal{L}_{\mathscr{D}_u} = \mathcal{L}_{\mathscr{D}_l} - \frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{k}\left[\tilde{\mathbf{y}}_i^{(j)}\log\left(\hat{y}_i^{(j)}\right)+\left(1-\tilde{\mathbf{y}}_i^{(j)}\right)\log\left(1-\hat{y}_i^{(j)}\right)\right] \tag{2}$$

where $\tilde{\mathbf{y}}_i$ denote the pseudo-labels generated by the teacher model, $\hat{y}_i$ the predicted labels by the student model, and M is the dataset size $\mathscr{D}_u$.

4.2.2 Evaluation

We conduct a set of experiments to benchmark *"how effective is our semi-supervised approach compared to state-of-the-art baselines employing the same number of labelled samples?"*, our evaluation on the two benchmarking datasets FB15k-ET and YAGO43k-ET shows that given a small amount of training data, our approach significantly outperforms supervised baselines trained on the same labelled training data.

		FB15k-ET						**YAGO43k-ET**					
		Top 3		Top 5		Top 10		Top 3		Top 5		Top 10	
		$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1
Embeddings	TransE-ET	0.34	0.60	0.42	0.40	**0.26**	0.34	0.35	0.48	0.28	0.34	0.17	0.28
	DistMult-ET	0.40	0.54	0.37	0.42	0.29	0.31	0.40	0.41	0.29	0.30	0.19	0.22
	ConnectE	0.26	0.73	0.37	0.55	0.32	0.44	0.14	0.80	0.14	0.69	0.12	0.55
Supervised	Logistic Regression	0.25	0.72	0.35	0.56	0.35	0.41	0.06	0.90	0.10	0.77	0.12	0.62
	RandomForest	0.26	0.72	0.34	0.57	**0.26**	0.46	0.09	0.76	0.11	0.60	**0.08**	0.46
	DNN	0.26	0.73	0.34	0.56	0.29	0.44	0.09	0.80	0.11	0.66	**0.08**	0.61
Semi-supervised	ASSET (*Teacher-Student*)	**0.24**	**0.74**	**0.33**	**0.59**	0.28	**0.47**	**0.04**	**0.93**	**0.09**	**0.80**	0.11	**0.64**

4.2.2.1 Result Analysis

Section 4.2.2 compares our approach both with embedding and supervised baselines on 1% of the two datasets FB15k-ET and YAGO43k-ET with varying numbers of entity types (Top 3, Top 5, and Top 10). We observe that—given such little training data—our semi-supervised approach ASSET significantly outperforms each of the baselines in terms of both $\mathcal{H}_{loss}$ and F_1-score with $p < 0.03$.[3] We attribute this to our teacher-student paradigm that augments the original training dataset with *pseudo-labelled* data from the unlabelled dataset, boosting overall performance (as discussed in Algorithm 1). Among the embedding approaches, ConnectE significantly outperforms the other KG embeddings TransE-ET and DistMult-ET in terms of both $\mathcal{H}_{loss}$ and F_1-score. For example, differences in terms of $\mathcal{H}_{loss}$ range from -0.05 (TransE-ET, YAGO43k-ET, Top 10) to -0.26 (DistMult-ET, YAGO43K-ET, Top 3) and differences of F_1-scores range from +0.10 (TransE-ET, FB15k-ET, Top 10) to +0.39 (DistMult-ET, YAGO43k-ET, Top 3). Our results corroborate previous findings [Zhao et al., 2020] that ConnectE outperforms other embedding models for the entity typing task, and we employ ConnectE embeddings as feature representation to train our supervised baselines (Logistic Regression, RandomForest, and DNN). Among the supervised approaches, the differences are less pronounced and the DNN achieves comparable performance to Logistic Regression and RandomForest. For example, the DNN is at least as good as Logistic Regression in 6 out of 12 measurements and at least as good as RandomForest in 9 out of 12 measurements.

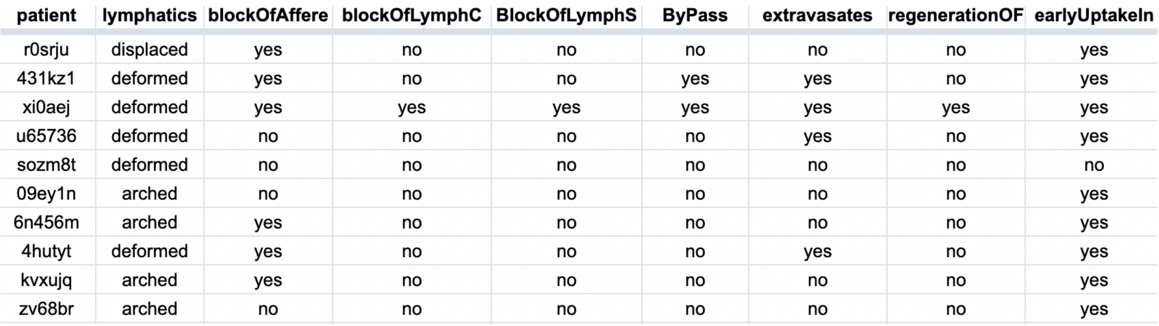
[3] For each pair of approaches, we employ a two-sided Wilcoxon signed-rank test between the $\mathcal{H}_{loss}/F_1$-scores on Top 3, Top 5, and Top 10 of FB15k-ET and YAGO43k-ET datasets of two approaches. Our null hypothesis is that the two approaches produce $\mathcal{H}_{loss}/F_1$-scores from the same distribution.

4.2.3 Research Outcome

In the 11th International Conference on Knowledge Capture (K-CAP) 2021, we presented our approach for labelling entities in knowledge graphs using semi-supervised approach (Teacher-Student Model). Further, we demonstrated our experimental to show the effectiveness of our approach compared with state-of-the-art baselines. In future work, we plan to employ our approach on Wikidata and help its community to predict newly introduced types with little training data. More details can be found at https://www.k-cap.org/2021/index.html.

5 Extraction of Axioms

An ontology is a formal set of terms (known as axioms or concepts) that are used to represent a domain of knowledge by using description logic, such as Web Ontology Language (OWL). There are four main aspects of an ontology: *classes, properties of classes, relationships between classes, and constraints between classes.* Given *classes* and *relationships* as input from the *entity labelling* process (see Step 4), we aim to generate their ontology representation and save into an OWL format. Figure 4 shows an example of input CSV file of Lymphography. The input file consists of medical information about patients with Lymphography.



patient	lymphatics	blockOfAffere	blockOfLymphC	BlockOfLymphS	ByPass	extravasates	regenerationOF	earlyUptakeIn
r0srju	displaced	yes	no	no	no	no	no	yes
431kz1	deformed	yes	no	no	yes	yes	no	yes
xi0aej	deformed	yes	yes	yes	yes	yes	yes	yes
u65736	deformed	no	no	no	no	yes	no	yes
sozm8t	deformed	no	no	no	no	no	no	no
09ey1n	arched	no	no	no	no	no	no	yes
6n456m	arched	yes	no	no	no	no	no	yes
4hutyt	deformed	yes	no	no	no	yes	no	yes
kvxujq	arched	yes	no	no	no	no	no	yes
zv68br	arched	no	no	no	no	no	no	yes

Figure 4: Lymphography Tabular Data

Our goal is to formulate this knowledge from the input file into a machine-readable format using OWL language. To do so, first we use Vectograph library to convert the input CSV file to its corresponding RDF graph, where rows represent entities and columns are their relations. Then, we use OWLready2 library[4] to process the generated RDF graph and generate the OWL ontology, as shown in 5. The following code[5] shows how Vectograph and OWLready2 libraries are used to generate an OWL ontology in Python3.

```
import os
from owlready2 import *

## file Upload
OUT_FOLDER = "uploads"

if not os.path.isdir(OUT_FOLDER):
    os.mkdir(OUT_FOLDER)

class AxiomGenerator:
```

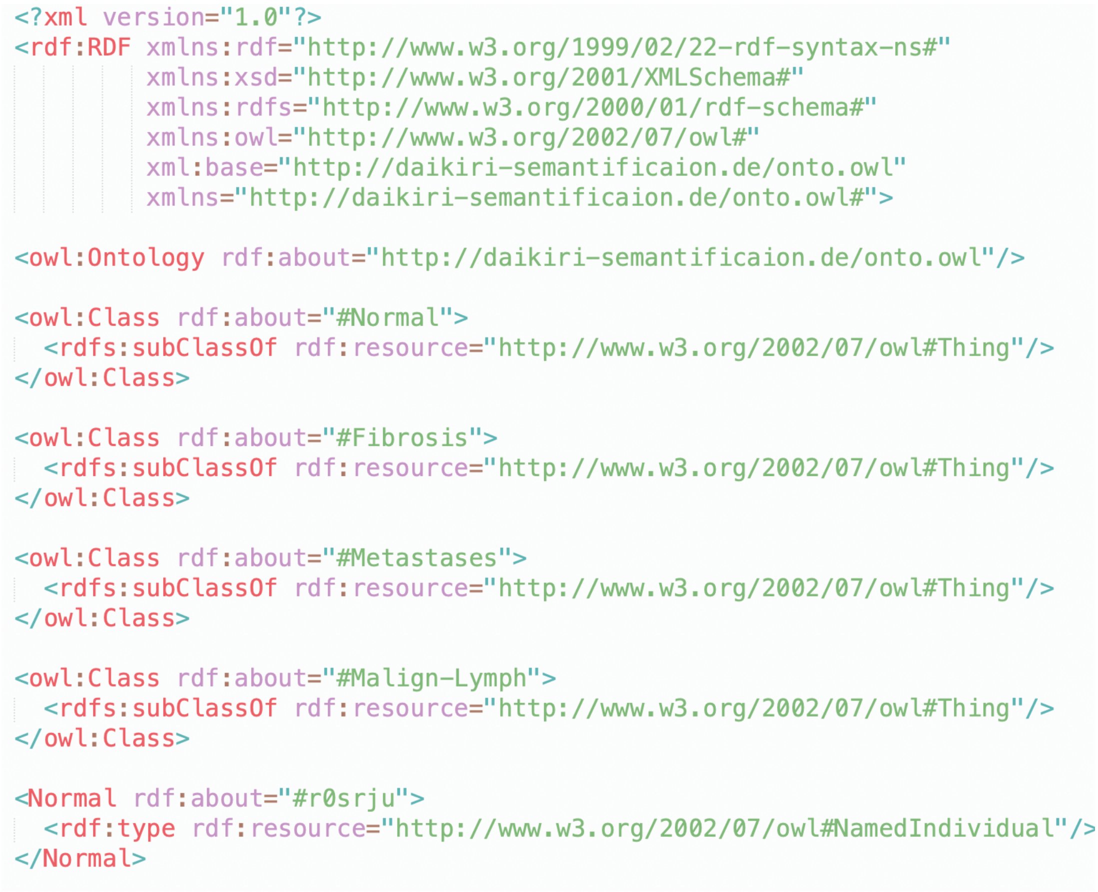
[4] https://owlready2.readthedocs.io/

[5] The full source code is available at https://github.com/dice-group/LabENT

```python
    def __init__(self, reader_file1, reader_file2):
        onto = get_ontology("http://daikiri-semantificaion.de/onto.owl")

        classes= {}
        with onto:
            for id_, type_ in reader_file1:
                parent = Thing
                Class = types.new_class(type_, (parent,))
                classes[id_]= Class

            for entity_id, cluster_id in reader_file2:
                # assign each entity it's cluster type
                individual= classes[cluster_id](entity_id)
            onto.save(OUT_FOLDER+'/semantification-ontology.owl')
```



```xml
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:owl="http://www.w3.org/2002/07/owl#"
         xml:base="http://daikiri-semantificaion.de/onto.owl"
         xmlns="http://daikiri-semantificaion.de/onto.owl#">

<owl:Ontology rdf:about="http://daikiri-semantificaion.de/onto.owl"/>

<owl:Class rdf:about="#Normal">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>

<owl:Class rdf:about="#Fibrosis">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>

<owl:Class rdf:about="#Metastases">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>

<owl:Class rdf:about="#Malign-Lymph">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>

<Normal rdf:about="#r0srju">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
</Normal>
```

Figure 5: The generated ontology of Lymphography data

To this end, the semantification process demonstrated that ontologies can be learned automatically from tabular data by the means of I) embedding-based clustering, II) labelling entities in the embedding space, and III) formally describing entities and their relations using OWL 2.0. In the next step, the learned ontology is provided as an input to the structure machine learning work package.

5.1 Research Outcome

We propose an unsupervised approach, Tab2Onto, for learning ontologies from tabular data using knowledge graph embeddings, clustering, and a human in the loop. We conduct a set of experiments to investigate our approach on a benchmarking dataset from a medical domain to learn ontology of diseases. Our Semantification approach has been accepted in the 19th Extended Semantic Conference (ESWC) 2022. More details can be found here http://tab2onto.dice-research.org/.

6 Implementation

In this section, we provide the implementation details of semantification modules. We have implemented two applications for labelling entities (LabENT and ASSET) and one application for generating ontologies. In the following, we briefly describe the details of each application:

LabENT:

we develop a crowdsourcing application with a web interface for labelling entities based on their semantic relations. As discussed in section 4.1, this application allow human expert to label entities and propagate the types based on their embedding representation and clustering. We used the WordPress framework[6] to develop the front-end interface. In particular, we use nicepages[7] module to build the HTML pages and maintain the content of LabENT V1.0 website. For the back-end functions, we created a MySQL database for maintaining the entities' information *(cluster-ID, entity-ID, proprieties, type)*. Once, the user assign types to the entities, the types' information are saved back into the MySQL database. Further, we use PHP to develop the labelling functions: I) load entities from a MySQL database and present to a user, II) save the entity types into the database. For more information about the installation and configuration, we refer the readers to the project repository on GitHub: www.github.com/dice-group/LabENT

ASSET:

This is our second application for labelling entities in the embedding space. We develop ASSET application to benefit from a few labelled entities and employ them for labelling a large unlabelled entities' dataset. Toward this goal, we employ a teacher-student learning paradigm, a semi-supervised learning algorithm that leverage learning from small labelled data to annotate another large dataset. The implementation of ASSET contains two models: student and teacher, both are developed using TensorFlow 2.0 and Python 3.6. We have released the source code of our approach and the experiments described in [Zahera et al., 2021] for research purposes. The details about the implementation, installation, and configuration can be found on the project repository on GitHub https://github.com/dice-group/ASSET

Axiom-Generator:

Finally, we developed the Axiom-Generator component as part of the WP3 to generate an OWL ontology, in particular, a taxonomy of entity classes. The output file contains a formal description of

[6] https://wordpress.org/
[7] https://nicepage.com/

entity types and their relationships. Using OWLready2, we created a Python script for loading entity types and their relationships from an input CSV file. We implemented the *axiom-generator* script in Python 3.7 and saved the output file in RDF format. This implementation is publicly available on the project's repository on GitHub at http://github.com/dice-group/Tab2Onto

7 Conclusion & Outlook

In this deliverable, we have presented two approaches for labelling entities in the embedding space. First, we proposed a manual approach via a crowdsourcing application that allows a domain expert to assign types for entities based on their semantic properties (e.g., RDF triples). Afterwards, our approach propagates the major type to all entities in each cluster. To this end, our approach annotates entities based on clustering and their embedding representation given a small labelled data. On the other hand, we proposed a semi-supervised approach (dubbed ASSET) that employ learning from both unlabelled data and labelled data using a teach-student model. Our approach achieved superior performances compared with state-of-the-art baselines on two benchmarking datasets (FB15k-ET and YAGO43k-ET). In our future work, we plan to investigate zero-shot learning approaches to learn missing types for unseen entities in knowledge graphs.

References

Deepak Ajwani, Bilyana Taneva, Sourav Dutta, Patrick K. Nicholson, Ghasem Heyrani-Nobari, and Alessandra Sala. Annotate: organizing unstructured contents via topic labels. *2018 IEEE International Conference on Big Data (Big Data)*, pages 1699–1708, 2018.

Ahmet Aker, Emina Kurtic, A. R. Balamurali, Monica Lestari Paramita, Emma J. Barker, Mark Hepple, and Robert J. Gaizauskas. A graph-based approach to topic clustering for online comments to news. In *ECIR*, 2016.

Russa Biswas, Radina Sofronova, Mehwish Alam, Nicolas Heist, Heiko Paulheim, and Harald Sack. Do judge an entity by its name! entity typing using language models. In *ESWC (Satellite Events)*, volume 12739, pages 65–70, 2021.

David Carmel, Haggai Roitman, and Naama Zwerdling. Enhancing cluster labeling using wikipedia. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 139–146, 2009.

Bo Chen, Xiaotao Gu, Yufeng Hu, Siliang Tang, Guoping Hu, Yueting Zhuang, and Xiang Ren. Improving distantly-supervised entity typing with compact latent space clustering. In *NAACL-HLT*, pages 2862–2872, 2019.

Sebastian Furth and Joachim Baumeister. Towards the semantification of technical documents. In *LWA*, pages 45–51. Universitätsbibliothek Bamberg, 2013.

Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. Unsupervised graph-based topic labelling using dbpedia. In *WSDM '13*, 2013.

Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML2013*, volume 3, page 896, 2013.

André Melo, Johanna Völker, and Heiko Paulheim. Type prediction in noisy RDF knowledge bases using hierarchical multilabel classification with graph and latent features. *Int. J. Artif. Intell. Tools*, 26(2):1760011:1–1760011:32, 2017.

Changsung Moon, Steve Harenberg, John Slankas, and Nagiza F Samatova. Learning contextual embeddings for knowledge graph completion. 2017.

Arvind Neelakantan and Ming-Wei Chang. Inferring missing entity type instances for knowledge base completion: New dataset and methods. *CoRR*, abs/1504.06658, 2015.

Jesper E. van Engelen and Holger H. Hoos. A survey on semi-supervised learning techniques. *Mach. Learn.*, 109(2):373–440, 2020.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 29(12):2724–2743, 2017.

Patrick Westphal, Lorenz Bühmann, Simon Bin, Hajira Jabeen, and Jens Lehmann. Sml-bench–a benchmarking framework for structured machine learning. *Semantic Web*, 10(2):231–245, 2019.

Bo Xu, Yi Zhang, Jiaqing Liang, Yanghua Xiao, Seung-won Hwang, and Wei Wang. Cross-lingual type inference. In *DASFAA*, volume 9642, pages 447–462, 2016.

Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

Hamada M Zahera, Stefan Heindorf, and Axel-Cyrille Ngonga Ngomo. Asset: A semi-supervised approach for entity typing in knowledge graphs. In *Proceedings of the 11th on Knowledge Capture Conference*, pages 261–264, 2021.

Yu Zhao, Anxiang Zhang, Ruobing Xie, Kang Liu, and Xiaojie Wang. Connecting embeddings for knowledge graph entity typing. In *ACL*, pages 6419–6428, 2020.