UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4: Informatik

WeST
People and Knowledge Networks
Institute for Web Science
and Technologies

Enrichment of ontological taxonomies using a neural network approach

Bachelorarbeit

zur Erlangung des Grades einer Bachelor of Science (B.Sc.)
im Studiengang Informatik

vorgelegt von
Alex Baier

Erstgutachter: Prof. Dr. Steffen Staab
Institute for Web Science and Technologies

Zweitgutachter: Max Mustermann
Institute for Web Science and Technologies

Koblenz, im Februar 2017

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

	Ja	Nein
Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden.	□	□
Der Veröffentlichung dieser Arbeit im Internet stimme ich zu.	□	□
Der Text dieser Arbeit ist unter einer Creative Commons Lizenz verfügbar.	□	□
Der Quellcode ist unter einer Creative Commons Lizenz verfügbar.	□	□
Die erhobenen Daten sind unter einer Creative Commons Lizenz verfügbar.	□	□

. . . .

(Ort, Datum) (Unterschrift)

Contents

1 Introduction 1

2 Foundations 1
2.1 Wikidata . . . 1
2.2 Taxonomy . . . 3
2.3 Similarity . . . 6
2.4 Problem statement . . . 7
2.5 k-nearest-neighbors classification . . . 8

3 Analysis of the Wikidata taxonomy 9

4 Ontology learning 13
4.1 Process and architecture for ontology learning . . . 16
4.2 Approaches for learning taxonomic relations . . . 16
4.3 Ontology learning using neural networks . . . 17

5 Neural networks 17
5.1 Recursive neural networks for graph representation . . . 17
5.2 Deep neural networks for graph representation . . . 17
5.3 Continuous Bag-of-Words . . . 17
5.4 Skip-gram with negative sampling . . . 18
5.5 Comparison . . . 18

6 Algorithm 18
6.1 Baseline . . . 18
6.2 Supplementing with other resources . . . 18

7 Evaluation 18
7.1 Method . . . 18
7.2 Generation of gold standard . . . 18
7.3 Results . . . 18

8 Conclusion and future work 18

A Appendix: Graphs 21

List of Figures

1 Example of Wikidata class: photographic film (Q6239) 2
2 The spectrum of ontology kinds. [25] 4
3 Example for k-nearest neighbors for 3 classes with k=4 and k=10. . . 8
4 Distance of subclasses to root class *entity (Q35120)*. Wikidata (2016/11/07) 10
5 Percentage of unlinked classes with a specific amount of unique properties. Wikidata (2016/11/07) 11
6 Frequency of properties in unlinked classes. Wikidata (2016/11/07) . 11
7 Percentage of unlinked classes with a specific amount of instances. Wikidata (2016/11/07) 12
8 Percentage of unlinked classes with a specific amount of subclasses. Wikidata (2016/11/07) 12
9 Percentage of unlinked, labeled, instantiated classes with a specific amount of unique properties. Wikidata (2016/11/07) 14
10 Frequency of properties in unlinked, labeled, instantiated classes. Wikidata (2016/11/07)) 14
11 Percentage of unlinked, labeled, instantiated classes with a specific amount of instances. Wikidata (2016/11/07)) 15
12 Percentage of unlinked, labeled, instantiated classes with a specific amount of subclasses. Wikidata (2016/11/07) 15

1 Introduction

TODO: Motivation. Related work. Justify novelty. Solution. Evaluation.

The thesis is structured as follows. Section 2 defines the concepts related to Wikidata, taxonomies, similarity measures and k-nearest-neighbors classification. Subsequently, the problem statement is formalized and its corresponding challenges listed. Section 3 presents the results of the analysis of Wikidata's taxonomy. Characteristics of unlinked classes are identified, and a specific subset of unlinked classes chosen, which fulfills necessary requirements as input for an algorithm. Section 4 compares the problem, solved by the thesis, to related work in the field of ontology learning. The problem is classified and solutions to similar problems analyzed. The novelty of the work is justified. Section 5 explains the notion of neural networks at the example of a simple feedforward network with backpropagation. Subsequently, neural networks for graph and word representation are compared in regards to suitability in the problem's context. Section 6 describes the developed baseline algorithm, which combines Word2Vec by Mikolov et al. [18] and k-nearest-neighbors. Possible variations of the algorithm are presented and compared in regards to possible gains and problems. One of these variations is also implemented for evaluation purposes. Section 7 describes the evaluation methodology and presents the results. The baseline algorithm, using multiple different sets of different hyperparameters, and a variation of it will be compared. Section 8 summarizes the results of the thesis and lists possible future work, which could extend upon it.

2 Foundations

2.1 Wikidata

Wikidata is an open, free, multilingual and collaborative knowledge base. It is a structured knowledge source for other Wikimedia projects. It tries to model the real world, meaning every concept, object, animal, person, etc. Wikidata is mostly edited and extended by humans, which in general improves the quality of entries compared to fully-automated systems, because different editors can validate and correct occurring errors. However, Wikidata, like most knowledge bases, is incomplete and therefore has to be operated under the *Open World Assumption* (OWA). OWA states that if a statement is not contained in a knowledge base, it is not necessarily false but rather unknown [12].

In Wikidata items and properties exist. Items are the aforementioned concepts, objects, etc. While properties are used to make claims about items, e.g. *photographic film (Q6293)* is a *subclass of (P279) data storage device (Q193395)* (see Figure 1). Each item and property has an unique identifier, which starts with the letter Q for items and the letter P for properties and is followed by a numeric code. The identifiers in Wikidata are essential to avoid ambiguity and to make items and properties multilingual.

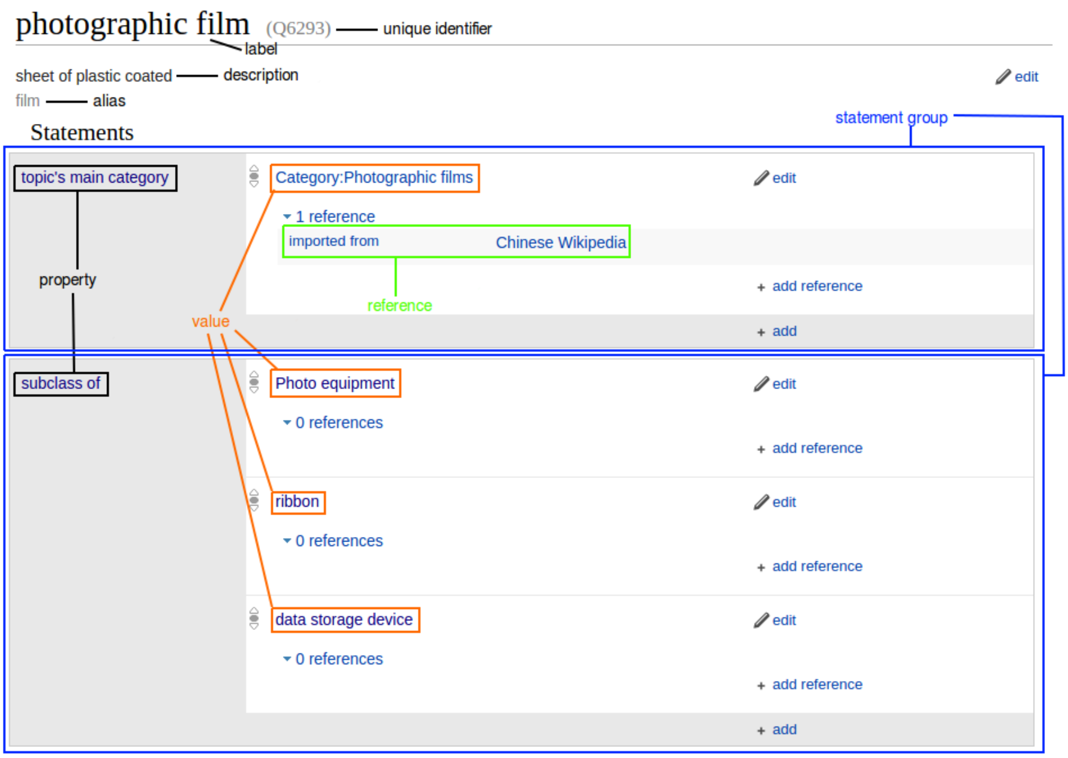

Figure 1: Example of Wikidata class: photographic film (Q6239)

Items consist of labels, aliases and descriptions in different languages. Sitelinks connect items to their corresponding pages of Wikimedia projects like Wikipedia articles. Most importantly item are described by statements. Statements are in their simplest form a pair of property and value, assigned to a specific item. A value is either a literal value or another item. It should be noted that an item can have multiple statements with the same property. The set of statements with the same property is called statement group. Statements can be annotated with qualifiers, which specify the context of the statement, e.g. population at a certain point of time. Additionally, references can be used for statements to include its source. See Figure 1 for an example of a Wikidata item.

Following, the terms of item and statement are defined in the context of Wikidata.

Definition 1 (Item)**.** Anitem is a tuple $(id, label, aliases, description, sitelinks)$:

- $id \in \mathbb{N}$ is the numerical item ID;
- $label \in String$ is the English label of the item;
- $aliases \in \mathcal{P}(String)$ is the set of English synonyms for the label;
- $description \in String$ is a short sentence describing the item;

- $sitelinks \in String \times String$ is a set of tuples $(site, title)$, where $site$ refers to a specific site of the Wikimedia projects, e.g. enwiki, and $title$ is the corresponding article title of the item on this site.

Definition 2 (Statement)**.** A statement is a tuple $(itemid, pid, value, refs, qualifiers)$:

- $itemid \in \mathbb{N}$ is a numerical item ID, to which the statement belongs;
- $pid \in \mathbb{N}$ is a numerical property ID;
- $value$ is either a constant value like string, int, etc., or an item ID;
- $refs$ is a set of references, containing the source of information for the statement;
- $qualifiers$ is a set of qualifiers, which further specifies the statement.

In Wikidata, there is no strict distinction between classes and instances. Both groups are represented as items. This leads to the issue, that recognizing, whether an item is a class or instance is not trivial. Based on which statements connect two items, a distinction can be made. A class is any item, which has instances, subclasses or is the subclass of another class. In Wikidata, the properties *instance of (P31)* and *subclass of (P279)* exist, which describe this relation between items. Therefore to identify whether an item is a class, it needs to be checked, whether the items fulfills any of the three above criteria.

Definition 3 (Class)**.** Given a set of items I and a set of statements R. $c = (classid, \_, \_, \_, \_) \in I$ is a class, if at least one of the following assertions are true:

$$\exists i = (instanceid, \_, \_, \_, \_) \in I\ \exists s = (itemid, pid, value, \_, \_) \in R : \\ instanceid = itemid \land pid = 31 \land value = classid \text{ (has instance)}$$

$$\exists s = (itemid, pid, \_, \_, \_) \in I : itemid = classid \land pid = 279 \text{ (is subclass)}$$

$$\exists i = (subclassid, \_, \_, \_, \_) \in I\ \exists s = (itemid, pid, value, \_, \_) \in R : \\ itemid = subclassid \land pid = 279 \land value = classid \text{ (has subclass)}$$

_ is used as an anonymous placeholder, for the purpose of not naming unused elements in tuples. For example, *photographic film (Q6293)* (Figure 1) is a class, because it is the subclass of three other classes.

2.2 Taxonomy

"*Ontologies* are (meta)data schemas, providing a controlled vocabulary of concepts, each with an explicitly defined and machine processable semantics" [17]. Additionally it is possible for ontologies to contain axioms used for validation and constraint enforcement. Ontologies enable the modeling and sharing of knowledge in a specific domain and support the knowledge exchange via web by extending syntactic to

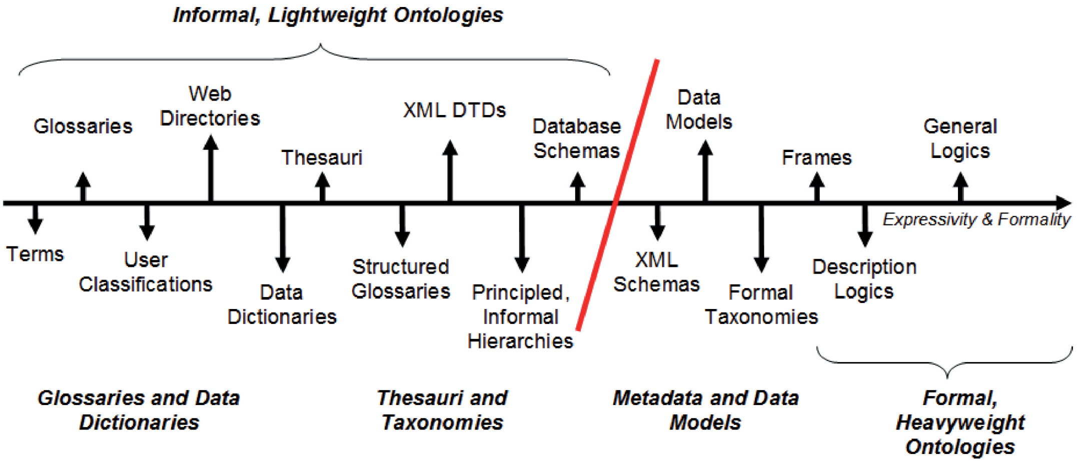

Figure 2: The spectrum of ontology kinds. [25]

semantic interoperability [14]. In comparison, a knowledge base like Wikidata can be seen as an instantiation of such an ontology, since every knowledge base has to be conceptualized by an ontology [25]. Different types of ontologies can grouped by their level of formality and expressiveness. Wong et al. [25] differentiates ontologies as lightweight and heavyweight ontologies (see Figure 2). *Taxonomies* are concept or class hierarchies. They typically represent a parent-child structure, which can be formalized with a single relationship called for example *subclass-of* in the case of Wikidata. The observed taxonomy in Wikidata belongs to the category of lightweight ontologies, specifically *principled, informal hierarchies*, as the only enforced rule for the subclass-of relation is that it should connect two entities [2].

For the purpose of developing a formal definition of the thesis' problem statement the notion of taxonomy needs to be formalized. Cimiano [7] defines a heavyweight ontology, which includes a taxonomy, as follows:
"

Definition (Ontology)**.** An ontology is a structure

$$\mathcal{O} := (C, \leq_C, R, \sigma_R, \leq_R, \mathcal{A}, \sigma_{\mathcal{A}}, \mathcal{T})$$

consisting of

- four disjoint sets C, R, $\mathcal{A}$, and $\mathcal{T}$ whose elements are called concept identifiers, relation identifiers, attribute identifiers and data types, respectively,
- a semi-upper latice $\leq_C$ on C with top element $root_C$, called concept hierarchy or taxonomy,
- a function $\sigma_R : R \to C^+$ called relation signature,
- a partial order $\leq_R$ on R, called relation hierarchy, where $r_1 \leq_R r_2$ implies $|\sigma_R(r_1)| = |\sigma_R(r_2)|$ and $\pi_i(\sigma_R(r_1)) \leq_C \pi_i(\sigma_R(r_2))$, for each $1 \leq i \leq |\sigma_R(r_1)|$, and

- a function $\sigma_{\mathcal{A}} : \mathcal{A} \to C \times \mathcal{T}$, called attribute signature,
- a set $\mathcal{T}$ of datatypes such as strings, integers, etc.

Hereby, $\pi_i(t)$ is the i-th component of tuple t. [...] Further, a semi-upper lattice $\leq$ fulfills the following conditions:

$\forall x (x \leq x)$ (reflexive)
$\forall x \forall y (x \leq y \wedge y \leq x \implies x = y)$ (anti-symmetric)
$\forall x \forall y \forall z (x \leq y \wedge y \leq z \implies x \leq z)$ (transitive)
$\forall x x \leq top$ (top element)
$\forall x \forall y \exists z (z \geq \wedge z \geq y \wedge \forall w (w \geq x \wedge w \geq y \implies w \geq z))$ (supremum)

So every two elements have a unique most specific supremum. ″

A taxonomy can be modeled as a semi-upper lattice. This induces two important assumptions about the structure and to some degree completeness of the observed taxonomies. First, there is only one *root class*, top element of the lattice, of which every other class is (transitively) a subclass. Second, because of the supremum property, the taxonomy is fully connected, which means each class, but the root class, has a superclass. Wikidata's taxonomy does therefore not fulfill the definition by Cimiano [7], as it is not fully connected.
In the following, definitions wil bel presented, which attempt to model an incomplete taxonomy based on the already presented data model and structure of Wikidata. Refer to Appendix A for the necessary definitions on graphs.

In Wikidata, a class can have multiple superclasses, therefore a tree structure is not sufficient to model the taxonomy. However a directed acyclic graph, can model the taxonomy. The acyclic constraint is necessary to ensure that no class is transitively a subclass of itself.

Definition 4 (Taxonomy)**.** A taxonomy $T = (C, S)$ is a directed acyclic graph, where C is a set of class identifiers, and S is the set of edges, which describe the subclass-of relation between two classes. such that c_1 is the subclass of c_2, if $(c_1, c_2) \in S$.

Definition 5 (Subclass-of relation)**.** The transitive binary relation $\lhd_T$ on the taxonomy $T = (C, S)$ represents the subclass relationship of two classes in T. Given $c_1, c_2 \in C$, $c_1 \lhd_T c_2$, if there is a walk $W = (c_1, \ldots, c_2)$ with length $n \geq 1$, which connects c_1 and c_2. $\lhd_T$ is transitive, $\forall c_1, c_2, c_3 \in C : c_1 \lhd_T c_2 \wedge c_2 \lhd_T c_3 \implies c_1 \lhd_T c_3$.

If the taxonomy defined by Cimiano [7] is mapped on this graph-based taxonomy model, the following assumption is true, for $T = (C, S)$:

$$|\{c \in C \mid \neg \exists s \in C : c \lhd_T s\}| = 1 \tag{1}$$

Only one class in this taxonomy has no superclasses. This class is called *root class*. However in the case of Wikidata, this assumption does not hold true. The following state is the case:

$$|\{c \in C \mid \neg\exists s \in C : c \lhd_T s\}| > \qquad (2)$$

There are classes other than the root class, which also have no superclasses. These classes will be called *unlinked classes*.

Definition 6 (Root class)**.** Given a taxonomy $T = (C, S)$, the root class $root_T$ is a specific, predefined class with no superclasses in T. For $root_T$, $|succ_T(root_T)| = 0$ applies.

Definition 7 (Unlinked class)**.** Given a taxonomy $T = (C, S)$ with a root class $root_T$, a class $u \in C$ is called unlinked class, if $u \neq root_T \wedge |succ_T(u)| = 0$.

In Wikidata, the root class is *entity (Q35120)* [3]. All other classes, which are not subclasses of *entity (Q35120)*, are therefore either unlinked classes, or subclasses of unlinked classes. In Section 3, it will be shown that the Wikidata taxonomy graph is not fully connected. But the component, which contains the root class *entity (Q35120)*, contains 97% of all classes. This component will be referred to as *root taxonomy* in later sections.

2.3 Similarity

- semantic similarity e.g. distributional similarity
  Lin [16]
  Rodríguez and Egenhofer [22]
- geometrical similarity e.g. distance based-similarity, cosine similarity

For the task of ontology learning [14] as well as classification, e.g. k-nearest-neighbors, the concept of similarity is of importance. A basic intuition of similarity is for example given by Lin [16]. Similarity is related to the commonalities and differences between two objects. More commonalities implies higher similarity. Vice versa, more differences implies lower similarity. Two identical objects should have the maximum similarity. In addition, only identical objects should be able to achieve maximum similarity. A similarity measure computes the similarity between two objects [24]:

Definition 8 (Similarity measure)**.** $sim : O \times O \mapsto [0, 1]$ is called similarity measure for a set of objects O.

A similarity of 1 implies identical objects and a similarity of 0 implies that there are no commonalities between the input objects.

TODO: Whole paragraph needs citations In ontology learning, semantic similarity is used to great effect . Semantic similarity compares the semantic content of objects or documents . This can be achieved by comparing which features can be found in both objects (commonalities) and which features are unique to the compared objects (differences). Rodríguez and Egenhofer [22] develops a semantic similarity measure for comparing entity classes in ontologies. **Write**

Similarity between vectors can be computed by calculating their distance [24]. **Write**

2.4 Problem statement

The task of this thesis is the classification of unlinked classes in Wikidata. In other words a function is needed, which given an unlinked class u of a taxonomy $T = (C, S)$ with a root class $root_T$, find an appropriate superclass for T. Doan et al. [10] suggests that for the task of placing a class into an appropriate position in T, either finding the most similar class, most specific superclass, or most general subclasses of u, are sensible approaches. This induces that the appropriate superclass for an unlinked class u is the superclass of the most similar class to u. Therefore we can define the problem, as follows:

Definition 9 (Problem definition)**.** Given a taxonomy $T = (C, S)$ with root class $root_T$ and a similarity function sim over T, find a function $f : \mathbb{N} \mapsto \mathcal{P}(\mathbb{N})$, which, given an unlinked class $u \in C$, returns a set of classes $P = f(u)$, fulfilling the following criteria:

$$\neg(\forall p \in S : p \lhd_T u \text{ no children} \tag{3}$$

$$= succ(\max_{c \in C}(sim(u, s))) \text{ superclasses of most similar classes} \tag{4}$$

Because it is possible for a class to have multiple superclasses, it is necessary to define f in such a way, that it returns a set of classes. Therefore the described problem is a multi-label classification problem.

The stated problem induces several challenges, which will be listed here, but addressed in later sections.

1. **Multi-label classification**. Algorithms for classification typically map entered objects to one label **TODO:Reference**. It has to be decided whether the solution will be simplified to assign one superclass per input, or attempt multi-label classification.

2. **High number of labels**. An entered, unlinked class has to be assigned to a class in the root taxonomy. The only restricting condition is that the chosen class cannot be a subclass of the unlinked class. As shown in Section 3 97% of 12999501 classes are part of the root taxonomy. Classification like SVM or neural networks are usually used for classification with a small number of labels. SVM is for example a binary classification method. A classification method, which is able to handle over a million of labels, has to be found.

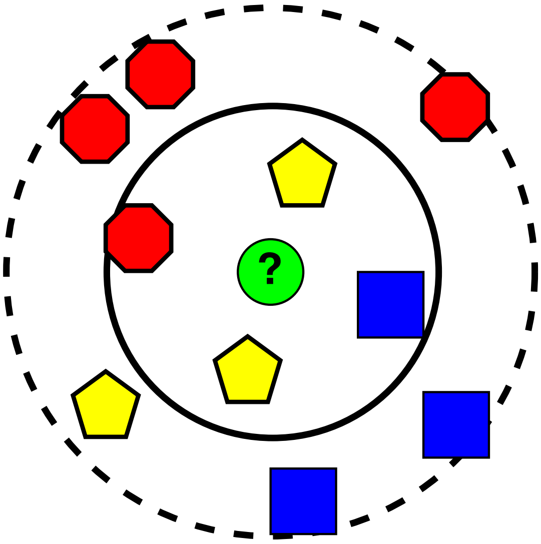

Figure 3: Example for k-nearest neighbors for 3 classes with k=4 and k=10.

3. **Representation of items**. Items in Wikidata are structured information, similar to nodes in RDF graphs. Items are nodes and statements can be simplified to triples of $(itemid, pid, value$. As motivated in Section 1, the solution should exploit the apparent power of neural networks. Neural networks, which are introduced in Section 5, require input to be represented as vectors. Therefore it will be necessary to map items to vectors.

2.5 k-nearest-neighbors classification

Chen et al. [5]
Zhang and Zhou [26]
Based on the characteristics of the classification problem, described by the problem statement, and the challenges attached to it, the k-nearest-neighbor algorithm (kNN) seems like an appropriate tool for solving the task. Nearest-neighbors classification is a lazy method, as it does not require training before testing. This is useful for applications with high amounts of data, large numbers of classes, and changing data [26]. For the considered use case of classification in Wikidata, these are very important strengths, as the number of classes in the taxonomy is very high and Wikidata is being constantly edited.

k-nearest-neighbors is a distance- or similarity-based classification method, which maps its input to a single label. This means, that kNN, given an input u only requires the distance or similarity between u and all training samples. Additional knowledge about the structure or content of the data is not required.

Algorithm 1 (weighted kNN)**.** Let O be a specific set of objects, e.g. $\mathbb{R}^n$, with a similarity measure sim. Let $n \in \mathbb{N}$ be the number of labels. Let $T = \{(o_i, l_i)\}$ with

$i = 0, \dots, m-1$ and be the set of training samples with m samples, where $o_i \in O$ is an object and $l_i \in [0, n-1]$ is its corresponding class. Let $u \in O$ be an unclassified object. **TODO: insert missing definitions**

1. Find the set K of k nearest neighbors to u in T. $K = \{\}$
2. Each item $k \in K$ a weight is assigned based on its similarity to u. $w_k = sim(k, u)/sum(sim(o_j, u))$.
3. Each item votes for its label using its weight. The label which has the highest vote score is assigned as label to u. l

The task defined in this thesis can be considered a multi-label classification problem. Zhang and Zhou [26] developed a multi-label nearest neighbor algorithm, which is based on kNN. **TODO: explain it**

3 Analysis of the Wikidata taxonomy

TODO: maybe separate this section into subsections For the task of developing an algorithm, which takes unlinked classes as input, it is necessary to know, what information the classes carry and if there are certain patterns among the classes. For this purpose an analysis of the taxonomy needs to be carried out, which may answer questions.

The taxonomy contained in the Wikidata dump of 2016/11/07 was analyzed. Classes were recognized as defined in Section 2.1. Unlinked classes were identified by checking whether a class does not have the *subclass of (P279)* property. The taxonomy contained a total of 1299501 classes at this time.

The state of the taxonomy was captured in regards to the root class *entity (Q35120)* (see Figure 4). 1260842 classes are currently subclasses of *entity (Q35120)*. 97% of all classes are therefore nodes in the root taxonomy. This implies a high agreement in the Wikidata community on which class is considered root, and thereby also supports the modeling decision made in Section 2.2, which assumes that a taxonomy only has one root, and this root is *entity (Q35120)* in Wikidata. **Last sentences of this paragraph are mostly interesting trivia, but have currently no further use.** The longest shortest distance between the root and a leaf class is 20. The classes *Q639064*, *Q15978631* and *Q151055* fulfill this characteristics. Each of them is subclass of *Homo (Q171283)*.

The characteristics of all unlinked classes and the root class were analyzed. This set contains a total of 16373 classes. 13807 classes have an English label and 11534 classes have a corresponding English Wikipedia page.

Regarding the number of unique properties (or statement groups) per class (see Figure 5) the median is 3 and the average is ≈ 4.8. Combined with the analyzed

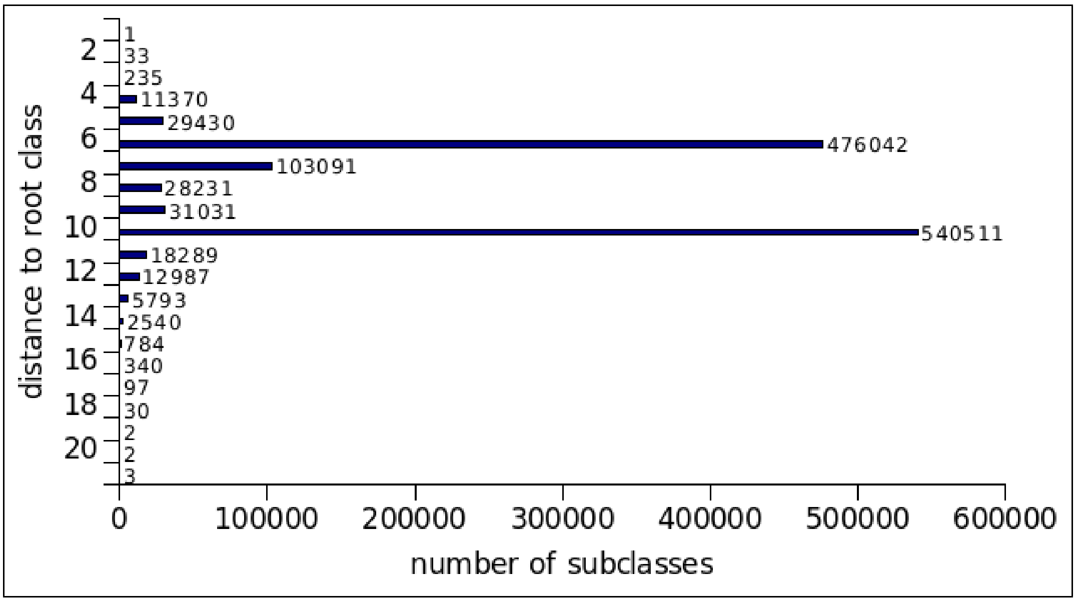

Figure 4: Distance of subclasses to root class *entity (Q35120)*. Wikidata (2016/11/07)

unique property frequency (see Figure 6), it can be seen that many unlinked classes are linked to other knowledge bases and taxonomies , e.g. Freebase or GND, using an identifier. Another observation is that 901 of the unlinked classes have properties related to taxons, which is a, to some degree, separate taxonomy in Wikidata, created with properties like *taxon name (P225)*, *parent taxon (P171)*, etc.

Most unlinked classes have only 0 to 1 instances, the median is 1 and the average is ≈ 4.65 instances per class (see Figure 7). There are however outliers with big numbers of instances, which skew the average number of instances per class. This may for example imply, that classes are created mainly for the purpose of grouping newly created instances.

The median and average for subclasses per class are 0 and ≈ 0.85 respectively (see Figure 8). This implies that the graph components of the unlinked classes are very small, and in most cases contain only the respective unlinked class.

The question has to be asked, whether an algorithm should try to handle the complete set of unlinked classes, or only a specific subset, which fulfills certain requirements. The answer depends on what information such an algorithm requires as input. The full set of unlinked classes is problematic for use in an algorithm as there are few shared characteristics over all classes. $\approx 16\%$ of classes are not labeled and $\approx 31.4\%$ of classes have no instances. Both labels and instances are useful characteristics for a class to have. The label allows the recognition of the class in natural text and ensures that the class fulfills a basic quality criteria, since the label is the first characteristic of a Wikidata item, which should be created by a user [1]. Instances represent classes and thereby describe them. For example, Rodríguez and Egenhofer [22] use instances to compute semantic similarity between classes. There-

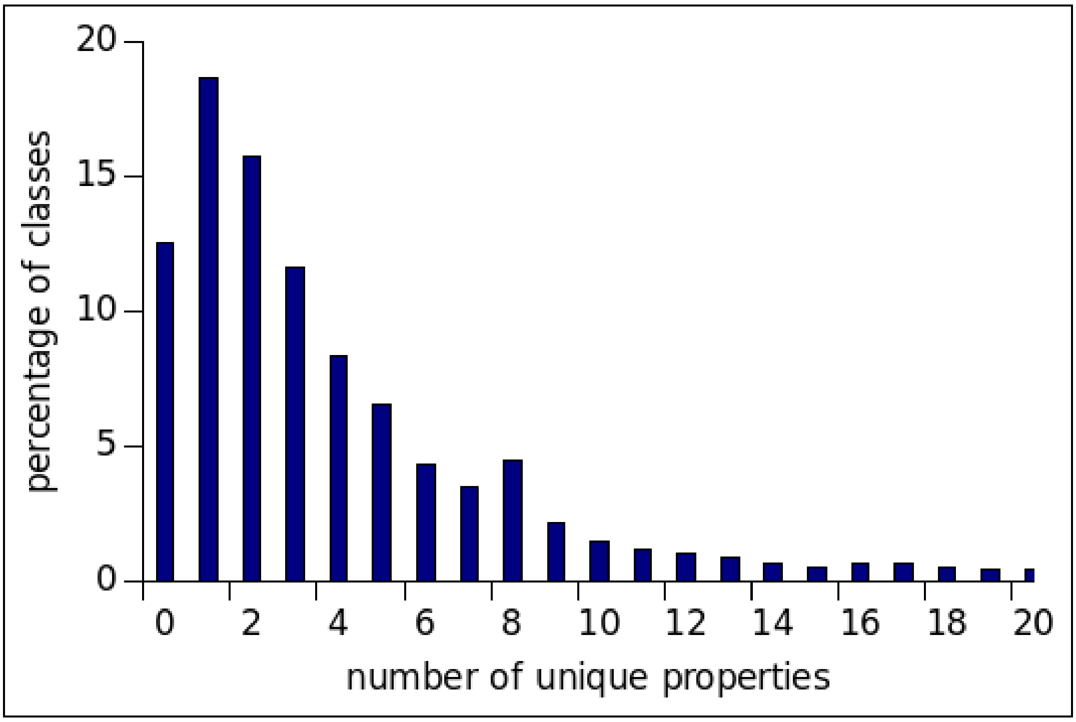

Figure 5: Percentage of unlinked classes with a specific amount of unique properties. Wikidata (2016/11/07)

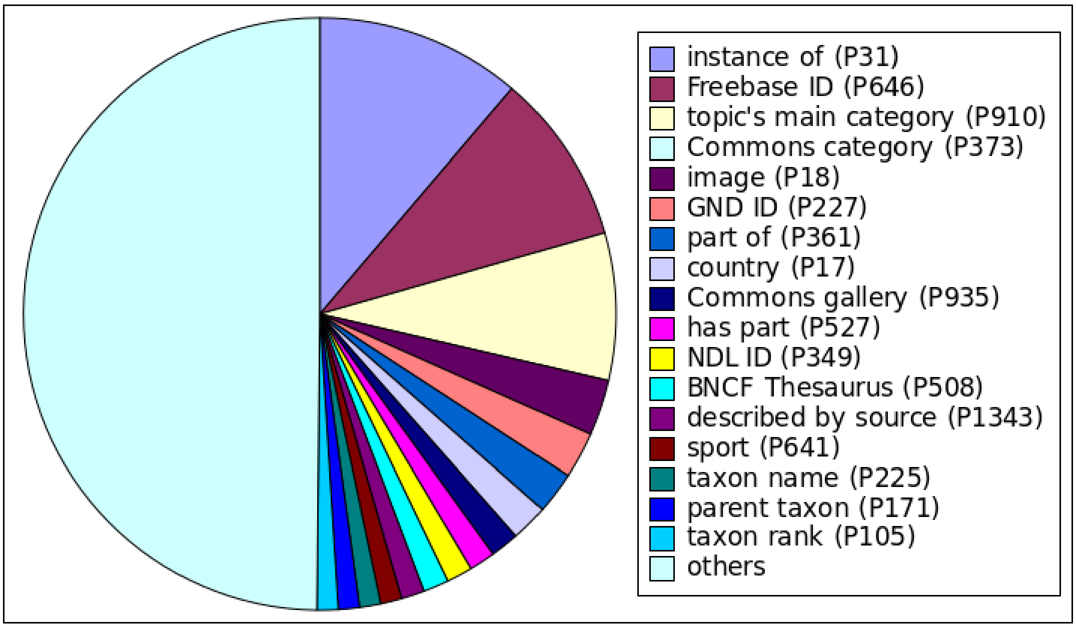

Figure 6: Frequency of properties in unlinked classes. Wikidata (2016/11/07)

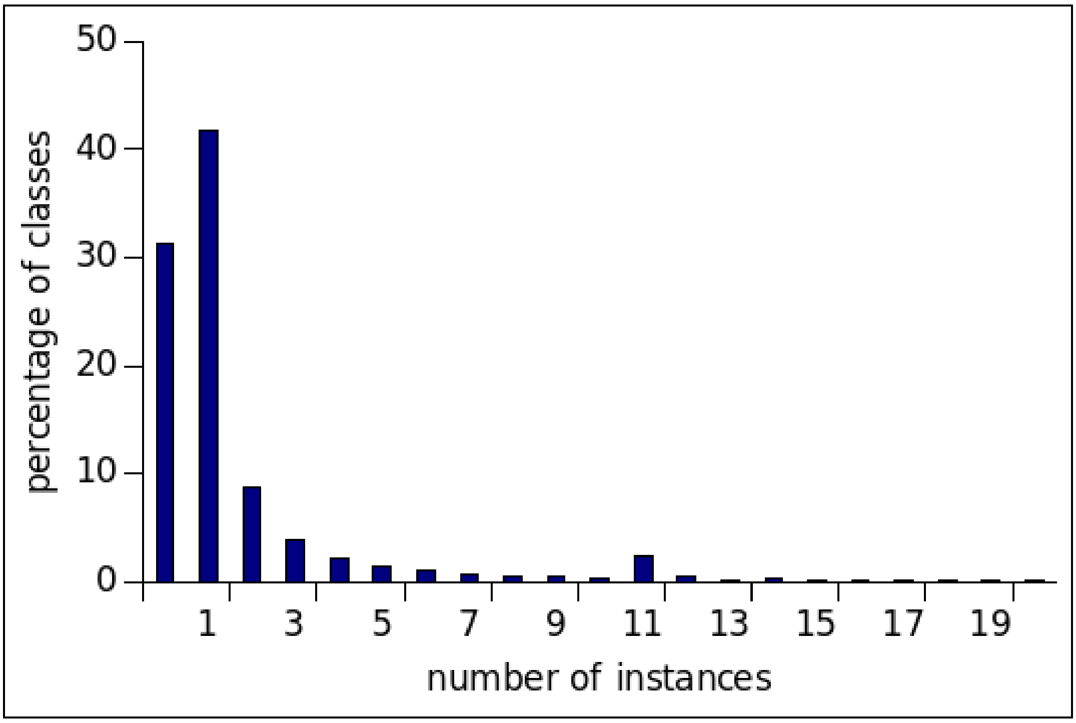

Figure 7: Percentage of unlinked classes with a specific amount of instances. Wikidata (2016/11/07)

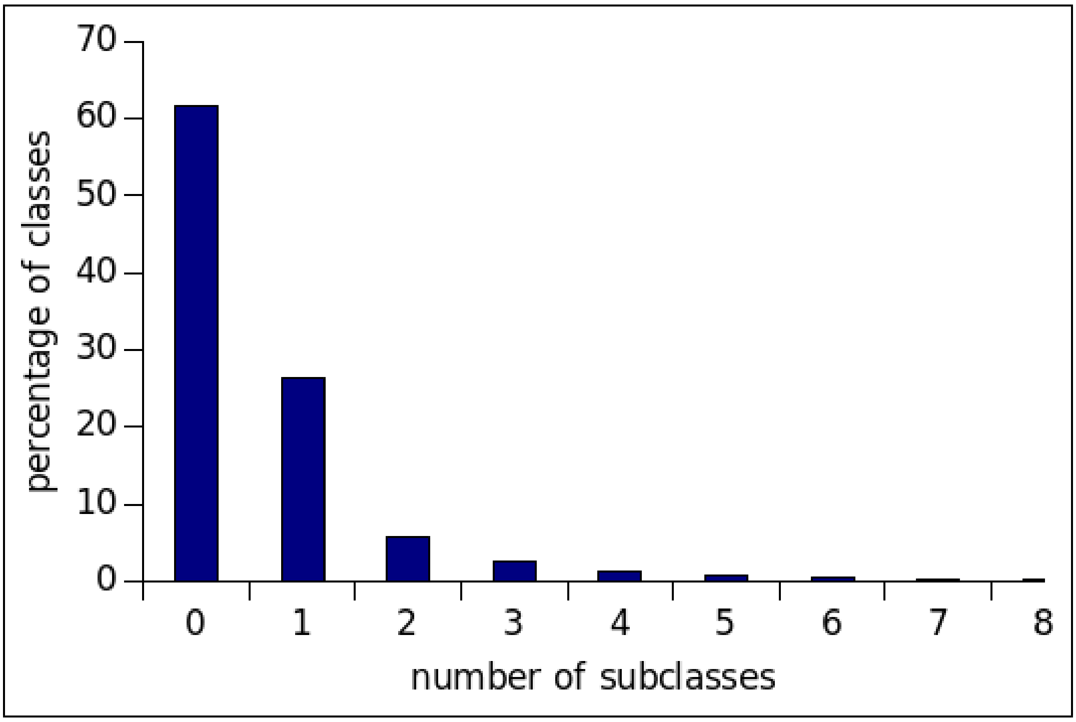

Figure 8: Percentage of unlinked classes with a specific amount of subclasses. Wikidata (2016/11/07)

fore, I propose to only consider the set of classes, which have a label in the English language and at least 1 instance, as input for the algorithm. **TODO: Find a short name or acronym for this subset.**

Following this argument, the subset of labeled, instantiated and unlinked classes was analyzed. This set contains a total of 9157 classes. 7557 classes have an English Wikipedia page, which is $\approx 82.5\%$ of all analyzed classes in the subset, and an increase of $\approx 12\%$ in comparison to the full set of unlinked classes.

The average of unique properties per class increases from 4.8 to 5.5, while the median remains at 3 (see Figure 9). The frequency of different properties appearing in classes however is more distributed (see Figure 10). This implies that the classes of this subset are related to different topics, and share only few commonalities.

As expected the average of instances per class increased to ≈ 7.37 from ≈ 4.65. However the median is still 1 (see Figure 11). $\approx 59\%$ of classes have only 1 instance.

At the same time the average of subclasses per class halved to ≈ 0.4 (see Figure 12). **can i just do such an assumption? the data supports it a bit, but i am not searching for counterclaims. and the assumption is not directly relevant to the thesis.** Combined with the unchanged median of 1 instance per class, supports the assumption, that classes are mainly created for the purpose of grouping instances. While adding the newly created class to the main taxonomy is being neglected.

The classes of the subset have, on average, a better level of description, more unique properties and instances, than the full set of unlinked classes. Between these classes the number of commonalities is lower, because the same properties occur not as frequently for different classes. The lack of commonalties between unlinked classes is not very problematic, as the algorithm would not compare unlinked classes with each other but with classes in the root taxonomy.

After completing the analysis, it can be seen that *entity (Q35120)* is the root class of the taxonomy, and Wikidata's taxonomy is in a good state, since 97% of classes are part of the root taxonomy. **TODO: not sure if i can just say that it is in a good state** Unlinked classes share few commonalities, and as shown above, a percentage of classes lack even basic information like a label. For developing an algorithm it is necessary to set a baseline, of what can be expected from the input. In this case, input is required to be labeled and have at least one instance. The benefit of this specific restriction is, that a basic level of descriptiveness can be assumed for every class. A label additionally allows supplementing the information, provided by Wikidata, to be supplemented with natural text sources, like Wikipedia, since it is now possible to recognize occurrences of the class in text.

4 Ontology learning

General concepts. Classification of considered problem in the task of ontology learning. Related work.

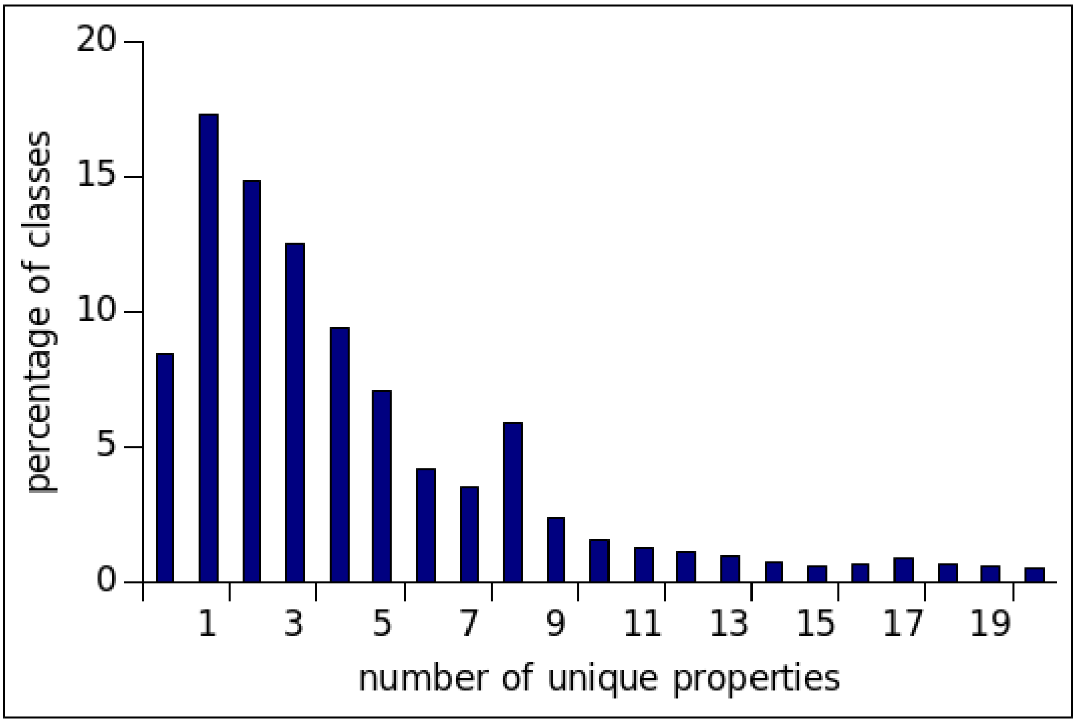

Figure 9: Percentage of unlinked, labeled, instantiated classes with a specific amount of unique properties. Wikidata (2016/11/07)

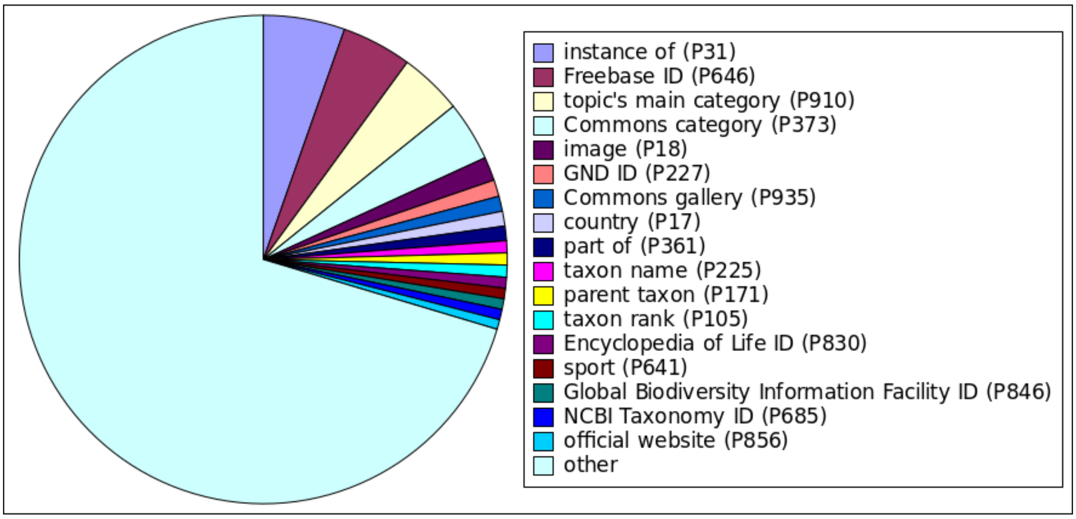

Figure 10: Frequency of properties in unlinked, labeled, instantiated classes. Wikidata (2016/11/07))

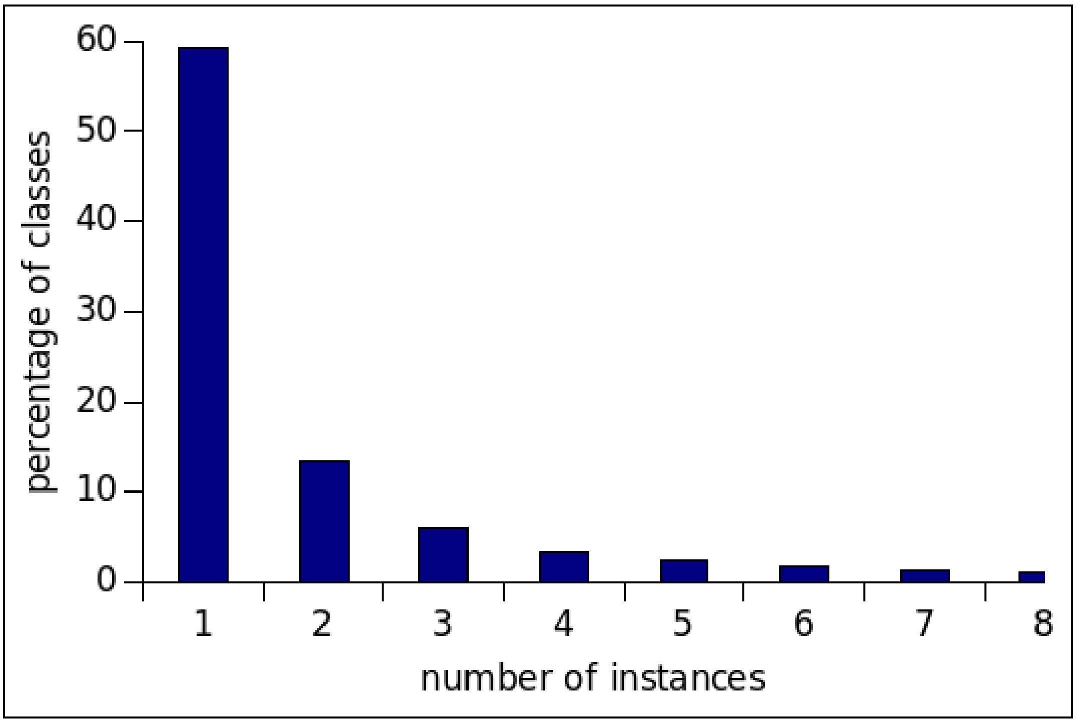

Figure 11: Percentage of unlinked, labeled, instantiated classes with a specific amount of instances. Wikidata (2016/11/07))

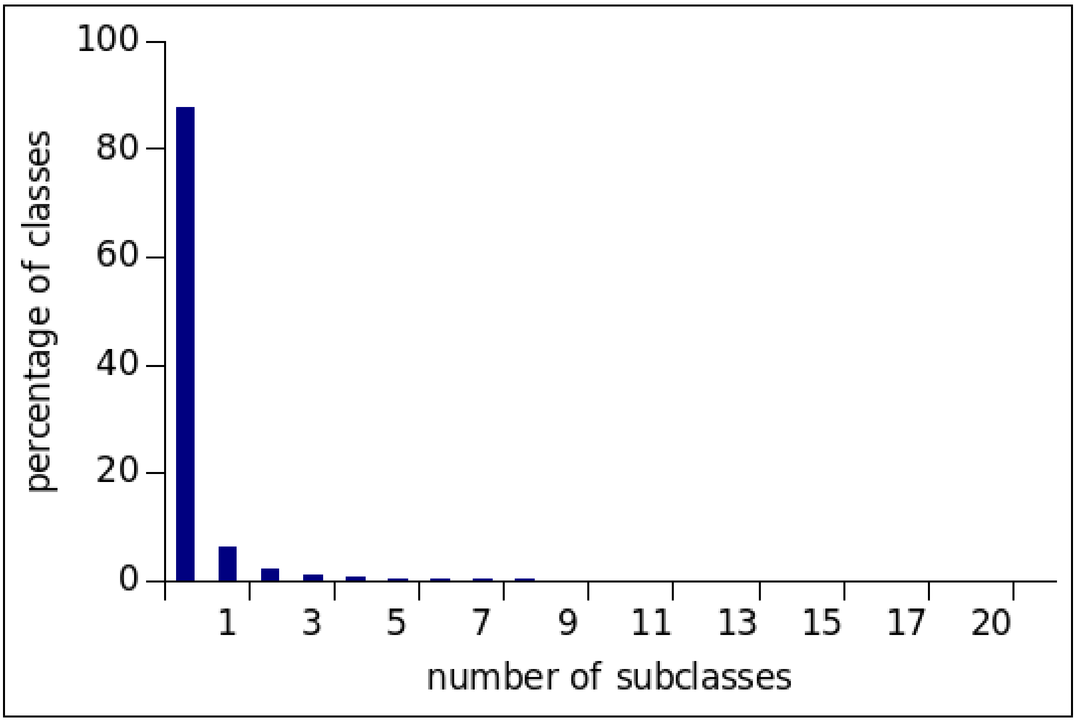

Figure 12: Percentage of unlinked, labeled, instantiated classes with a specific amount of subclasses. Wikidata (2016/11/07)

Petrucci et al. [20] neural network for ontology learning
Fu et al. [11] word embeddings for word hierarchies
Hazman et al. [14] Survey on ontology learning
Cimiano et al. [6] Ontology learning intro
Maedche and Staab [17] OntoEdit
Wong et al. [25] Ontology learning from text

Manually building ontologies is an expensive, tedious and error-prone process [14]. Maedche and Staab [17] recognize that the manual construction of ontologies results in a *knowledge acquisition bottleneck*. The field of *ontology learning (OL)* supports ontology engineer in this process by providing ontology learning techniques in the form of tools like *OntoEdit* [17]. The process of OL can be divided into different subtasks. OL tools consist of different components, which automatically or semi-automatically support this process. The process of ontology learning and the components of a generic OL architecture are summarized and the task of the thesis is categorized. Following, basic algorithms for learning taxonomic relations are summarized. Both subsections are based on work by Cimiano et al. [6], Maedche and Staab [17], and Hazman et al. [14]. Finally, related work, which exploit neural networks are analyzed and compared to the thesis' task. The novelty and additional benefits of this work are justified.

4.1 Process and architecture for ontology learning

4.2 Approaches for learning taxonomic relations

A subgroup of algorithms for ontology learning is concerned with learning taxonomic relations. The following approaches, categorized by Cimiano et al. [6], use text as input.

Lexico-syntactic patterns are word patterns in patterns, which are used to identify hypernym-hyponym pairs (superclass-subclass pairs) in natural text. For example, such a pattern is

$$NP_{hyper} \text{ such as } \{NP_{hypo},\}^* \ \{(\text{and} \mid \text{or})\} \ NP_{hypo}$$

, where NP stands for noun phrase, and NP_{hyper} is a hypernym or superclass, while NP_{hypo} are hyponyms or subclasses. These patterns provide reasonable results, but the manual creation of patterns is involve "high"**not really high, but it could be automatized** cost and time investments [25].

Clustering uses some measure of similarity to organize objects into groups. This can be achieved by representing the words or terms as vectors [8], on which different distance or similarity measures can be applied. Clustering methods can be categorized to three different types. Agglomerative clustering initializes each term as its own cluster and merges in each step the most similar terms into one cluster. Divisive clustering approaches the problem the opposite way by starting with a single cluster, containing all words, and then dividing them into smaller groups. Both of these approaches generate hierarchies, agglomerative bottom-up and divisive top-down **maybe not say that or formulate it another way**.

Phrase analysis analyses noun phrases directly. It is assumed that nouns, which have additional modifiers are subclasses of the noun without modifiers. For example, this could be applied on the labeled unlinked classes of Wikidata. For example the classes *Men's Junior European Volleyball Championship (Q169359)* and *Women's Junior European Volleyball Championship (Q169956)* could be subclasses of *European Volleyball Championship (Q6834)*. In this case, phrase analysis interprets *Men's Junior* and *Women's Junior* as modifiers, which denote these classes as specialization to *Q6834*.

Classification-based approaches can be used, when a taxonomy is already present. In this case, classification can be used to add unclassified concepts to the existing hierarchy. Challenging with this task is that a taxonomy typically contains a large amount of classes and therefore classification methods like SVM are not suited for the task. Specific algorithms, which only consider, a subset of relevant classes are necessary to carry out an efficient classification. For example, Pekar and Staab [19] solves this problem by exploiting the taxonomy's tree structure using tree-ascending or tree-descending algorithms.

The algorithm developed by this thesis uses a classification-based approach, since a large taxonomy is already given. Instead of exploiting the hierarchic structure of the taxonomy, a variant of k-nearest-neighbors is used to address the "high number of labels" problem (mentioned in Section 2.4).

4.3 Ontology learning using neural networks

5 Neural networks

Notion of neural networks will be introduced.

5.1 Recursive neural networks for graph representation

Scarselli et al. [23]

5.2 Deep neural networks for graph representation

Cao et al. [4]
Raghu et al. [21]

5.3 Continuous Bag-of-Words

Mikolov et al. [18]

5.4 Skip-gram with negative sampling

Mikolov et al. [18]
Levy et al. [15]
Goldberg and Levy [13]
graph representations are not really relevant as there is only one big component and the input classes are not part of that component.

5.5 Comparison

6 Algorithm

6.1 Baseline

- Hyper parameters
- Training data

6.2 Supplementing with other resources

e.g. Wikipedia

7 Evaluation

7.1 Method

Dellschaft and Staab [9]

7.2 Generation of gold standard

7.3 Results

8 Conclusion and future work

References

[1] Wikidata: Create a new item. https://www.wikidata.org/wiki/Special:NewItem, . Accessed: 2017-02-05.

[2] Property talk:p279. https://www.wikidata.org/wiki/Property_talk:P279, . Accessed: 2017-01-29.

[3] Talk:q35120. https://www.wikidata.org/wiki/Talk:Q35120, . Accessed: 2017-01-29.

[4] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Deep neural networks for learning graph representations. In Dale Schuurmans and Michael P. Wellman, editors, AAAI, pages 1145–1152. AAAI Press, 2016. URL http://dblp.uni-trier.de/db/conf/aaai/aaai2016.html#CaoLX16.

[5] Yihua Chen, Eric K. Garcia, Maya R. Gupta, Ali Rahimi, and Luca Cazzanti. Similarity-based classification: Concepts and algorithms. J. Mach. Learn. Res., 10:747–776, June 2009. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1577069.1577096.

[6] P. Cimiano, A. Mädche, S. Staab, and J. Völker. Ontology learning. In S. Staab and R. Studer, editors, Handbook on Ontologies, International Handbooks on Information Systems, pages 245–267. Springer, 2nd revised edition edition, 2009. URL http://www.uni-koblenz.de/~staab/Research/Publications/2009/handbookEdition2/ontology-learning-handbook2.pdf.

[7] Philipp Cimiano. Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387306323.

[8] Philipp Cimiano, Andreas Hotho, and Steffen Staab. Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. Journal of Artificial Intelligence Research, 24:305–339, 2005. ISSN 10769757. doi: 10.1.1.60.228.

[9] Klaas Dellschaft and Steffen Staab. On how to perform a gold standard based evaluation of ontology learning. In Proceedings of the 5th International Conference on The Semantic Web, ISWC'06, pages 228–241, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-49029-9, 978-3-540-49029-6. doi: 10.1007/11926078_17. URL http://dx.doi.org/10.1007/11926078_17.

[10] AnHai Doan, Jayant Madhavan, Pedro Domingos, and Alon Halevy. Learning to map between ontologies on the semantic web. In Proceedings of the 11th International Conference on World Wide Web, WWW '02, pages 662–673, New York, NY, USA, 2002. ACM. ISBN 1-58113-449-5. doi: 10.1145/511446.511532. URL http://doi.acm.org/10.1145/511446.511532.

[11] Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. Learning Semantic Hierarchies via Word Embeddings. Acl, pages 1199–1209, 2014.

[12] Luis Galárraga. Rule Mining in Knowledge Bases. PhD thesis, Telecom ParisTech, 2016.

[13] Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. CoRR, abs/1402.3722, 2014. URL http://arxiv.org/abs/1402.3722.

[14] Maryam Hazman, Samhaa R El-Beltagy, and Ahmed Rafea. A Survey of Ontology Learning Approaches. International Journal of Computer Applications, 22(9):975–8887, 2011.

[15] Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. Transactions of the Association for Computational Linguistics, 3:211–225, 2015. ISSN 2307-387X. URL https://transacl.org/ojs/index.php/tacl/article/view/570.

[16] Dekang Lin. An information-theoretic definition of similarity. In Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-556-8. URL http://dl.acm.org/citation.cfm?id=645527.657297.

[17] Alexander Maedche and Steffen Staab. Ontology learning for the semantic web. IEEE Intelligent Systems, 16(2):72–79, March 2001. ISSN 1541-1672. doi: 10.1109/5254.920602. URL http://dx.doi.org/10.1109/5254.920602.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. CoRR, abs/1301.3781, 2013. URL http://arxiv.org/abs/1301.3781.

[19] Viktor Pekar and Steffen Staab. Taxonomy learning - factoring the structure of a taxonomy into a semantic classification decision. In Proceedings of the 19th Conference on Computational Linguistics, COLING-2002, August 24 - September 1, 2002 , Taipei, Taiwan, 2002, 2002.

[20] Giulio Petrucci, Chiara Ghidini, and Marco Rospocher. Using recurrent neural network for learning expressive ontologies. CoRR, abs/1607.04110, 2016. URL http://arxiv.org/abs/1607.04110.

[21] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein. On the expressive power of deep neural networks. ArXiv e-prints, June 2016.

[22] M. Andrea Rodríguez and Max J. Egenhofer. Determining semantic similarity among entity classes from different ontologies. IEEE Trans. on Knowl. and Data Eng., 15(2):442–456, February 2003. ISSN 1041-4347. doi: 10.1109/TKDE.2003.1185844. URL http://dx.doi.org/10.1109/TKDE.2003.1185844.

[23] F. Scarselli, M. Gori, Ah Chung Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. IEEE Transactions on Neural Networks, 20(1):61–80, jan 2009. ISSN 1045-9227. doi: 10.1109/TNN.2008.2005605. URL http://ieeexplore.ieee.org/document/4700287/.

[24] Roger Weber. Similarity Search in High-Dimensional Vector Spaces. PhD thesis, SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH, 2000.

[25] Wilson Wong, Wei Liu, and Mohammed Bennamoun. Ontology learning from text: A look back and into the future. ACM Comput. Surv., 44(4):20:1–20:36, September 2012. ISSN 0360-0300. doi: 10.1145/2333112.2333115. URL `http://doi.acm.org/10.1145/2333112.2333115`.

[26] Min-Ling Zhang and Zhi-Hua Zhou. A k-Nearest Neighbor Based Algorithm for Multi-label Classification. volume 2, pages 718–721 Vol. 2. The IEEE Computational Intelligence Society, 2005. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1547385`.

Appendices

A Appendix: Graphs

Definition 10 (Directed graph)**.** A directed graph G is an ordered pair $G = (V, E)$, where V is a set of vertices, and $E = \{(v_1, v_2) \mid v_1, v_2 \in V\}$ is a set of ordered pairs called directed edges, connecting the the vertices.

Definition 11. Subgraph Let $G = (V, E)$ and $H = (W, F)$ be directed graphs. H is called subgraph of G, if $W \subseteq V$ and $F \subseteq E$.

Definition 12 (Predecessor)**.** Let $G = (V, E)$ be a directed graph. $v_1 \in V$ is a predecessor of $v_2 \in V$, if there exists an edge so that $(v_1, v_2) \in E$. Let $v \in V$ be a vertice of G, then $pred_G(v) = \{w \mid (w, v) \in E\}$ is the set of predecessors of v.

Definition 13 (Successor)**.** $v_1 \in V$ is a successor of $v_2 \in V$, if there exists an edge so that $(v_2, v_1) \in E$. Let $v \in V$ be a vertice of G, then $succ_G(v) = \{w \mid (v, w) \in E\}$ is the set of successors of v.

Definition 14 (Walk)**.** Let $G = (V, E)$ be a directed graph. A walk W of length $n \in \mathbb{N}$ is a sequence of vertices $W = (v_1, \dots, v_n)$ with $v_1, \dots, v_n \in V$, so that $(v_i, v_{i+1}) \in E \; \forall i = 1, \dots, n-1$.

Definition 15 (Connected (Vertice))**.** Let $G = (V, E)$ be a directed graph. Vertices $u, v \in V$ are connected, if $u = v$, or $u \neq v$ and there is walk between u and v or v and u.

Definition 16 (Connected (Directed graph))**.** Let G be a directed graph. G is connected, if every pair of vertices in G are connected.

Definition 17 (Component)**.** Let G and H be directed graphs. H is a component of G, if H is a connected subgraph of G and H is not subgraph of another connected subgraph of G, which has more vertices or edges than H.

Definition 18 (Cycle)**.** A walk $W = (v_1, \dots, v_n)$ of length n is called a cycle, if $v_1 = v_n$.

Definition 19 (Directed acyclic graph)**.** A directed graph G is called directed acyclic graph, if there are no cycles in G.