Entity Classification with AnyBURL

Bachelor Thesis

presented by
Jonathan Max Vetter
Matriculation Number 1695327

submitted to the
Data and Web Science Group
Prof. Dr. Gemulla
University of Mannheim

June 2022

Contents

List of Tables

Chapter 1

Introduction

In the last ten years, knowledge graphs have become very important in research. Through them, many real-world relationships can be represented. These relationships can be used as a basis to make further statements / predictions (e.g. classes of entities [13]). In this thesis, the focus will be on entity classification with AnyBURL. AnyBURL (Anytime Bottom-Up Rule Learning) is a bottom-up technique for learning rules from a large knowledge graph [10]. One can think of rule learning from a knowledge graph as capturing certain patterns that occur in a knowledge graph as a rule [10]. These learned rules captured by AnyBURL can be used for many purposes. The most studied application for AnyBURL so far is the knowledge graph completion task (see [10]). Another application of AnyBURL in combination with knowledge graphs that has received little attention so far is entity classification. This is intended to enable prediction of which nodes in the knowledge graph will receive which labels. Since AnyBURL has already achieved very good results in knowledge graph completion compared to other models, we now want to test whether AnyBURL can also deliver similarly good results in the area of entity classification or whether more work on AnyBURL is required for this.

To find this out, we want to test AnyBURL on different entity classification datasets and subsequently compare it with other applications. In order to obtain optimal results with AnyBURL, we also want to look at whether and how the standard implementation of AnyBURL can be tuned in this thesis. Another task of this thesis is to look at the obtained entity classification results and also discuss them. By doing so, we should be able to identify possible incorrect predictions that occur across multiple datasets.

This bachelor thesis starts with the introduction of the underlying concepts, which should help to follow the topic of entity classification with AnyBURL (Chapter 2). Then, in Chapter 3, we provide a short explanation of what AnyBURL is and how it works. The next Chapter 4 is the main part of this thesis, in which we show how we constructed the experiments with AnyBURL and compare and discuss the results afterwards. At the end of this thesis, it should be determined whether the standard implementation of AnyBURL is a good alternative to the other applications in entity classification, or whether it needs more work. This will be done in Chapter 5.

1.1 Related Work

The field of entity classification in knowledge graphs, is currently not as strongly represented in the literature as other areas related to knowledge graphs (e.g. the Knowledge Graph Completion task). Finding papers in this already smaller area that is also based on rule learning like AnyBURL is difficult.

The paper ”A Comparative Study of Distributional and Symbolic Paradigms for Relational Learning” [2] deals with entity classification of knowledge graphs and knowledge graph completion. Their work is also based on a rule learner called *TILDE*. This allows a comparison of AnyBURL's results to another application, which is also based on the concept of learning rules. To ensure comparability, we can take the datasets of this work and generate our results based on the same datasets.

Finding more scientific papers on rule learning in entity classification was complicated. But to better assess AnyBURL's performance, it takes more than a single source. Therefore, we included the better studied sub-field of embeddings to show the comparability between different approaches as well as to conduct more experiments.

In the paper ”Do Embeddings Actually Capture Knowledge Graph Semantics?” [7] the authors discuss whether embeddings are also useful for semantic tasks in a knowledge graph. They examine the areas of categorization and clustering of entities. Another part of this paper deals with entity classification. This is the part that is relevant for us. Because from this part, as with the paper above, we can calculate results with AnyBURL based on the same datasets and then compare them afterwards. This makes it possible to classify AnyBURL's performance also in the context of embedding models.

Chapter 2

Theoretical Framework

In this chapter, the theoretical framework relevant to this thesis is explained in order to create a uniform basis on which the following chapters can be built. The following topics are covered in the theoretical framework: First, in section 2.1 we give a brief explanation of the term knowledge graph, including various definitions. This is followed by a summary of how knowledge graphs are to be understood in the context of this work. After that, in section 2.2 we deal with link prediction. Again, a definition is given and link prediction is discussed in general (see 2.2.1). Then we show how link prediction works for knowledge graphs (see 2.2.1). This is followed by entity classification (section 2.3), which can be seen as a special form of link prediction. As in the previous topics, after the general introduction to the topic, we will cover entity classification for knowledge graphs in more detail. Finally, in section 2.4 we deal with rule learning. Two different approaches to rule learning are presented, as well as the general structure of a rule.

2.1 Knowledge Graph

The term knowledge graph was introduced by Google in 2012 as a semantic extension of their search function, the aim being to search for things (real-world objects) instead of comparing results with strings [16].

After this informal introduction of the term knowledge graph, it also received more and more attention in research. However, this led to the fact that there was no uniform introduction of the knowledge graph and thus many different approaches / definitions emerged [3]. In order to form a uniform basis of how knowledge graphs are to be understood in the context of this work, we will take a closer look at the following two definitions and draw conclusions about what a knowledge graph is:

Definition 1 *"A knowledge graph (i) mainly describes real world entities and their interrelations, organized in a graph, (ii) defines possible classes and relations of entities in a schema, (iii) allows for potentially interrelating arbitrary entities with each other an (iv) covers various topical domains." [13]*

Definition 2 *"We define a Knowledge Graph as an RDF (Resource Description Framework) graph. An RDF graph consists of a set of RDF triples where each RDF triple* (s, p, o) *is an ordered set of the following RDF terms: a subject* $s \in U \cup B$*, a predicate* $p \in U$*, and an object* $U \cup B \cup L$*. An RDF term is either a URI* $u \in U$*, a blank node* $b \in B$*, or a literal* $l \in L$*." [4]*

In our scope, as stated in Definition 1, knowledge graphs mainly reflect a network of real-world objects and the relationship between these objects. Examples of real-world objects would be people, languages or animals. The information about objects and their relationships are stored in a set of triples.

These triples are formed in the same way as described in the Definition 2. This means that a triple always consists of a subject, a predicate and an object, i.e. (*subject*, *predicate*, *object*). To illustrate this, here are examples of two triples: (*Peter*, *live*, *Mannheim*), (*Peter*, *own*, *Dog*) The meaning of the two triples is easy to understand even for humans. The statements from the two triples are that Peter lives in Mannheim and that Peter owns a dog.

To make a connection to a classical graph, one can say that the predicate of a single triple can be seen as an edge between the subject and the object from the same triple [8]. If this were done with every triple of the knowledge graph, one would end up with a huge graph containing the knowledge of all triples.

2.2 Link Prediction

2.2.1 Introduction to Link Prediction

Generally speaking, link prediction is about making predictions about relationships within a network [19]. Normally, link prediction is used to predict new links based on existing nodes and relationships [19]. Some possible applications of link prediction are movie recommendations, knowledege graph completion or entity classification (which is explained in more detail in section 2.3), which is particularly relevant for this work [21]. Since the above statements about link prediction were rather general and informal, we look at Definition 3 that formally summarises the statements.

Definition 3 *"In a network $G = (V, E)$, V is the set of nodes and E is the set of edges. The link prediction task is to predict whether there is or will be a link $e(u, v)$ between a pair of nodes u and v, where $u, v \in V$ and $e(u, v) \notin E$." [19]*

Two categories can be derived from the link prediction problem:

- In a given network, missing links are inferred using link prediction, which is intended to detect possibly lost and / or hidden links [19]. An example would be to detect medication interactions.
- In a given network, future links are inferred using link prediction. These inferred links are links that could be added to the network in the future [19]. This would be the case, for example, to give certain people a prediction of which movie they might like based on their past.

2.2.2 Link Prediction for Knowledge Graphs

The task of link prediction is to derive missing or new facts from the existing information (= set of triples) of the knowledge graph. This means that the correct subject or object is to be predicted in a triple [14]. A distinction is made between two different types:

- $\langle ?, predicate, object \rangle$ is called head prediction [14]. Here the intention is to predict the head / subject of the triple (symbolized by the ?).
- $\langle subject, predicate, ? \rangle$ is called tail prediction [14]. Here the intention is to predict the tail / object of the triple (symbolized by the ?).

To state this more generally, for each prediction, the entity already known can be called the *source entity* and the entity to be predicted can be called the *target entity* [14]. The target entity is symbolised by the ? in the two cases above. There are different approaches to perform link prediction in a knowledge graph, for example embeddings or path ranking algorithm [14]. However, in this paper we focus on link prediction using rule learners, which will be explained further in section 2.4.

2.3 Entity Classification

The task of entity classification can be seen as a more specific task of link prediction (see section 2.2), since links must also be predicted here. To be more precise, a link between entities and types must be made. In order to perform entity classification, there must be entities that have associated labels [2]. These labels contain the type information, which means that a certain label reflects a type, for example a label called ”person” would be assigned to all entities that are persons.

For the problem of entity classification, it is worth mentioning that the number of labels is much smaller than the number of entities [20]. This is also logical, since many different entities can have the same label, so for example all people would get the same label person and not everyone a specific one.

The goal of entity classification is to predict these labels for previously unclassified entities and possibly assign other appropriate labels to already classified entities. In the end, all entities should be assigned all possible matching labels [2, 18]. For example, after a successful entity classification with the labels (*person*, *animal*, *male*) and the entities (*Peter*, *Bob*, *Lisa*), the result could be as follows:

- *Peter* has been assigned the following labels: *person*, *male*
- *Bob* has been assigned the following labels: *animal*, *male*
- *Lisa* has been assigned the following labels: *person*

To narrow this down a bit more for the application of entity classification in the area of knowledge graphs, one can say that entity classification is a special variant of the tail prediction task from the section 2.2.2. In general, the structure of tail prediction ($\langle subject, predicate, ?\rangle$) in the entity classification task follows this pattern:

- The *subject* is the entity to which a type should be assigned.
- The *predicate* is usually a type assignment, for example *typeOf* or *hasType*.
- The *?* is the entity type / label sought for the given entity.

2.4 Rule Learning

An important role in machine learning and data mining is the ability to learn rules [5]. In the process of rule learning, regularities are to be found in the underlying data sets, which are then to be derived into an IF-THEN rule [6]. There are two different approaches, depending on what kind of rule is to be derived. The two approaches are *Descriptive Rule Discovery* on the one hand and *Predictive Rule Learning* on the other. [6]. In the following sections (2.4.2 and 2.4.3), the two approaches are fundamentally explained and distinguished from each other. But first, section 2.4.1 explains how rules are constructed and how they can be understood.

2.4.1 Presentation of Rules

A rule is usually composed of a condition part and a conclusion [5]:

$$\mathbf{f_1} \wedge \mathbf{f_2} \wedge \mathbf{f_3} \wedge \ldots \wedge \mathbf{f_L} \longrightarrow \mathbf{c_j}$$

One can also see the condition part of a rule as a logical conjunction of different features $\mathbf{f_i}$ [5]. A feature $\mathbf{f_i}$ is a kind of query about whether the currently considered example fulfils it or not. For example, feature $\mathbf{f_i}$ says that you have to be a dog. The *rule length* is defined as the number $\mathbf{L}$ of features [5]. Often the condition part is also called the *body* of a rule.

The conclusion of a rule consists of a value $\mathbf{c_j}$ belonging to a class. The statement of the conclusion is basically only that if an example fulfils all features $\mathbf{f_i}$, the value $\mathbf{c_j}$ is predicted for this example [5]. Another name for the conclusion is the *head* of a rule.

2.4.2 Descriptive Rule Discovery

The main task of *Descriptive Rule Discovery* is to discover rules in the underlying dataset that can represent salience and certain patterns [6]. A difference to *Predictive Rule Learning* is that in *Descriptive Rule Discovery* the emphasis is on learning individual rules. The evaluation of the rules does not refer to the predictive performance of the rule, as in *Predictive Rule Learning*, but rather to the statistical validity [6].

Supervised learning and *unsupervised learning* stand out as the main application areas of *descriptive rule discovery* in the context of machine learning [6].

2.4.3 Predictive Rule Learning

The goal of *predictive rule learning* is to generalize the given training data in such a way that one can make predictions about new examples based on the learned rules [6]. Normally, individual rules cover only part of the entire training data, so completeness must be enforced by learning an unordered rule set [6]. This unordered rule set is a set of individual rules that, when combined, form a classifier [6].

Chapter 3

AnyBURL

This chapter is about the algorithm AnyBURL (**Any**time **B**ottom-**U**p **R**ule **L**earning) used for this thesis.

In Section 3.1 we briefly summarize what AnyBURL is, what it was developed for and how well the algorithm performs compared to others. After that, in section 3.2, we look at the language bias of AnyBURL. We explain what the language bias is and then show a short example of how to understand it. This is followed by a brief summary of how entity classification with AnyBURL is carried out in practice. This is done in section 3.3. Finally, in Section 3.4 we present how the algorithm works and show which parameters are available. We also give a pseudo code for AnyBURL in this section.

3.1 Introduction to AnyBURL

AnyBURL (**Any**time **B**ottom-**U**p **R**ule **L**earning) is a rule learner that has been developed for efficient learning of logical rules from a large knowledge graph [12]. The rule learning of AnyBURL is based on a bottom-up technique. This technique is built on the concept that an example is a concise representation of a particular rule that, by generalization, captures a broad subset of all positive examples. [10].

The main task for which AnyBURL was developed is knowledge graph completion. In this context, the focus of current research is usually on the concept of embedding a knowledge graph in a vector space. But as a result of this focus, symbolic approaches (such as AnyBURL) have received less attention [17].

However, compared to approaches based on vector spaces, AnyBURL yields similar results and in some cases even outperforms most models [12]. Furthermore, AnyBURL uses significantly less resources (in terms of runtime and memory) compared to the vector space based approaches [12]. Another advantage of

AnyBURL is that it continues to achieve very good results even if the algorithm is stopped after only a short periode of time [10]. In addition, symbolic approaches have the advantage that they provide an explanation in a form of rules that triggers a prediction [12].

3.2 Language Bias

In the context of AnyBURL, a rule $h(c_0, c_1) \leftarrow b_1(c_1, c_2), ..., b_n(c_n, c_{n+1})$ is called a *ground path rule* of length *n* [10]. The c_i in a rule stand for constants or entities of a knowledge graph (subject / object) and $h()$, $b()$ stand for a certain predicate of the knowledge graph. In the following description of the language bias, capital letters are used for variables and lower case letters for constants.

A rule that has no cycles in its body, i.e. formally expressed: $c_i \neq c_j$ for $i, j \in \{1, ..., n\}$ with $i \neq j$ and if $c_0 \neq c_i$ with $0 < i \leq n$, is called straight [10]. The straight ground path rules can be divided into two further categories, namely *cyclic rules* with $c_0 = c_{n+1}$ and *acyclic rules* with $c_0 \neq c_{n+1}$ [10]. The three types defined below, **C**, $\mathbf{AC_1}$ or $\mathbf{AC_2}$, include any reasonable generalization of a straight path rule of length n that is either a generalization of a path rule that is not straight or that is not also a generalization of a shorter path rule [10]. The following three definitions for the types are all taken from [10]. Furthermore, for the definitions, *X* and *Y* are used for variables that occur in the head of the rule and A_i for variables that occur in the body of the rule [10].

- **C:** $h(Y, X) \leftarrow b_1(X, A_2), ..., b_n(A_n, Y)$ [10]
- $\mathbf{AC_1}$**:** $h(c_0, X) \leftarrow b_1(X, A_2), ..., b_n(A_n, c_{n+1})$ [10]
- $\mathbf{AC_2}$**:** $h(c_0, X) \leftarrow b_1(X, A_2), ..., b_n(A_n, A_{n+1})$ [10]

All rules belonging to type **C** are generalizations of cyclic ground path rules, the rules belonging to type $\mathbf{AC_1}$ are generalizations of cyclic and acyclic ground path rules and all rules from $\mathbf{AC_2}$ are generalizations of acyclic ground path rules [10]. A rule that is more specific than rules that can be assigned to these three types has, instead of the variable A_k, a constant c_k with $k \leq n$ [10].

To illustrate this briefly with an example, Figure 3.1 shows a small part of a knowledge graph. The example is mostly inspired by the example from [10]:

Suppose the goal is to find out which rules describe why Peter speaks German (= $speaks(peter, g)$). In order to construct suitable rules, all paths with length *n* and starting at *peter* or *g* are considered [10]. A path can be created by following edges that enter or leave a node. In Figure 3.1, three different paths are highlighted

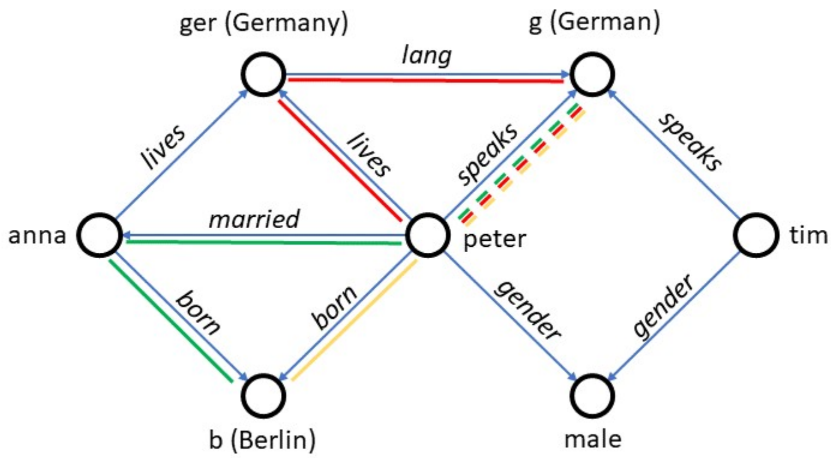

Figure 3.1: A small knowledge graph, which is intended for illustration purposes. Rule 3.1 is shown in Red, Rule 3.2 in Green and Rule 3.3 in Yellow. The solid line reflects the body of the rule and the dashed line the head of the rule. The graph was inspired by [10].

(solid line), all of which start at the node *peter*. The red path is cyclic in combination with $speaks(peter, g)$, the other two paths are acyclic paths. The following Rules 3.1, 3.2 and 3.3 are the bottom rules and need to be generalized [10].

$$speaks(peter, g) \leftarrow lives(peter, ger), lang(ger, g) \tag{3.1}$$

$$speaks(peter, g) \leftarrow married(peter, anna), born(anna, b) \tag{3.2}$$

$$speaks(peter, g) \leftarrow born(peter, b) \tag{3.3}$$

The following rules are the result of the generalization of Rule 3.1 and Rule 3.2 [10]:

$$speaks(X, g) \leftarrow married(X, A_2), born(A_2, b) \tag{3.4}$$

$$speaks(X, g) \leftarrow married(X, A_2), born(A_2, A_3) \tag{3.5}$$

$$speaks(X, Y) \leftarrow lives(X, A_2), lang(A_2, Y) \tag{3.6}$$

$$speaks(X, g) \leftarrow lives(X, A_2), lang(A_2, g) \tag{3.7}$$

$$speaks(peter, Y) \leftarrow lives(peter, A_2), lang(A_2, Y) \tag{3.8}$$

It is not difficult to see that whenever Rule 3.7 is true, Rule 3.6 is automatically true. Nevertheless, Rule 3.7 is not simply discarded, because it may be that both rules have different *Confidence* scores and it might therefore make sense to use these two rules in the prediction process [10].

The *Confidence* of a rule in this context is the number of body groundings that make the head of the rule true divided by the number of all body groundings for this rule.

3.3 Working Principle of AnyBURL for Entity Classification

To obtain results for entity classification with AnyBURL, one has to perform the following working principle:

- First let AnyBURL learn rules from a dataset (training set). To do this, one must specify the path to the dataset from which the rules are to be learned in the configuration file. In addition, one must specify a path where the learned rules are to be stored. At this point one can also insert the settings as described in section 4.2.3. It is important that the dataset used as input contains relations that indicate a type relation, otherwise no rules can be learned for the types.

- After AnyBURL is done learning rules from the given dataset, one can find the rules under the path given above. AnyBURL creates three files with rules. These have the suffixes *-10, -50, -100*, which indicate for how many seconds AnyBURL has learned rules.

- When making predictions for entities in terms of their type from a test set, one must give AnyBURL a path to this data. In addition, a path must be given that specifies the location where the predictions are to be stored. Now, if AnyBURL is supposed to make predictions, they will be stored under the path mentioned above. It is important to note that only predictions about type assignment are relevant for entity classification.

- When AnyBURL is done predicting, there are also three files with the suffixes *-10, -50, -100*. These indicate on which basis of rules the predictions were made. Looking at one of the files, one can see that there are always three lines belonging to one prediction. The first line indicates the triple of the test set for which the current prediction was calculated. The second line starts with "*Head:*", this can be neglected, as it only predicts the head of the triple, which is irrelevant for the given task. The third line, which starts with "*Tail:*" is the relevant line for entity classification, as it predicts the end of a triple, i.e. the type of the entity. This line is followed by an enumeration of the possible types in combination with their confidence. The type that appears first in this line is the type that AnyBURL would predict for this entity.

By following these steps, entity classification with AnyBURL is possible. The commands to run AnyBURL as well as the default configuration files can be found at this URL: *https://web.informatik.uni-mannheim.de/AnyBURL/*.

3.4 Algorithm

The basic concept of AnyBURL is to extract paths of length *n* from a given knowledge graph $\mathbb{G}$ [10]. The starting point is $n = 2$. When the algorithm considers a path of length *n*, it learns rules of length $n - 1$ until a certain saturation *sat* is reached [10]. If the specified saturation is reached, *n* is incremented by one and AnyBURL then learns rules of greater length [10]. The quality criteria *Q*, which is responsible for when a rule is stored (for example, a threshold for the confidence of a rule) and the saturation *sat* must be specified in advance [10]. The parameter *s* specifies the size of the sample and the parameter *ts* is the length of a period of time to be allowed for a learning process [10]. The set *R* contains all rules from the previous time spans, the set R_s contains all rules from the current time span and the set R'_s contains all rules found in the current time span as well as in one of the previous time spans [10].

In Algorithm 1, AnyBURL is listed again in the form of a pseudo code to provide additional clarity on how AnyBURL works.

Algorithm 1 Anytime Bottom-up Rule Learning (taken from [10])

ANYBURL($\mathbb{G}$, s, sat, Q, ts)

```
n = 2
R = ∅
loop
    R_s = ∅
    start = currentTime()
    repeat
        p = samplePath(𝔾, n)
        R_p = generateRules(p)
        for r ∈ R_s do
            score(r, s)
            if Q(r) then
                R_s = R_s ∪ {r}
            end if
        end for
    until currentTime() > start + ts
    R'_s = R_s ∩ R
    if |R'_s|/|R_s > sat then
        n = n + 1
    end if
    R = R_s ∪ R
end loop
return R
```

Chapter 4

Experimental Evaluation

In this chapter we deal with the experiments for classifying entities with AnyBURL. This includes everything that has to do with the experiments, from the datasets to the creation of the different experiments and the discussion of the results.

First, in Section 4.1, we look at the different datasets themselves. This includes a look at the different types that should be assigned to the entities. We also look at the different sizes of the datasets. The next Section 4.2 is about the actual experiments. We start with a short list of the goals that are to be achieved by the experiments. Furthermore, we give an explanation of how the datasets were used in the context of AnyBURL. We also take a look at how the results could possibly be improved. Finally, Section 4.3 presents the results of the entity classification and discusses the reasons for the results.

4.1 Datasets

In the following section we present the used datasets, as well as the entities that are to be identified. For more information about the size of the datasets, see the Table 4.1.
First, we take a look at the datasets from the paper "A Comparative Study of Distributional and Symbolic Paradigms for Relational Learning." [2]:

The *Hepatitis* dataset provides information from a set of patients who have hepatitis *Type B* and *Type C* [1]. Six different types of terrorist attacks (*Arson*, *Bombing*, *Kidnapping*, *NBCR Attack*, *Other Attacks* and *Weapon Attack*) are mapped to terrorist attacks in the *Terrorists* dataset [1]. The *Mutagenesis* dataset contains information about chemical compounds and their atoms [1]. The labels in this dataset are *Mutagenic_Yes* and *Mutagenic_No*. Last but not least, the *WebKB* dataset was used. This dataset consists of links and pages collected from the Cornell University web page [1]. *WebKB* consists of seven different labels (*Course*, *Department*, *Faculty*, *Person*, *Staff*, *Student* and *Research Project*).

For these datasets, it is worth mentioning that, for the purpose of cross-validation, the entirety of the training and test sets are split into five (for *WebKB* only four) different experiments, but each of them contains the whole data.

Next, the *FB15k-237* dataset from the paper "Do Embeddings Actually Capture Knowledge Graph Semantics?" [7] was used. In order to use it in the context of entity classification, different experiments for different levels were designed in the above mentioned paper [7]. Four of these experiments were tested for entity classification with AnyBURL. These four experiments are:

Level-1 (here: *FB15k-237 – Task 1*), with the types *Person*, *Organization* and *Product*; *Level-2-Organizations* (here: *FB15k-237 – Task 2*), with the types *Musical Organization*, *Party*, *Enterprises* and *Non-Governmental Institution*; *Level-2-Persons* (here: *FB15k-237 – Task 3*), with the types *Artist*, *Politician*, *Scientist*, *Officeholder* and *Writer*; and *Level-3-Artists* (here: *FB15k-237 – Task 4*), with the types *Photographer* and *Painter* [7]. In the brackets (here: ...) is the name of the experiments in Table 4.1, this serves to make the table easier to read.

It should be added that the original paper [7] is somewhat inaccurate in the description of the classes and also contains a small error. For the experiments shown in their table, the number of classes does not match those that are actually used. Most likely, the author discards classes represented by less than 40 examples, but does not mention this. Furthermore, the classes and experiment names for experiment *Level-3-Artists* in the table do not match those from their source code. The names from the source code should be the correct ones, because they match those from the dataset.
Finally, it should be mentioned that in the two papers [2, 7] training sets and test sets were created, but both lack of a validation set.

4.2 Experiments

In this section, we go into more detail about the entity classification experiments performed on the datasets from section 4.1.

To begin with, we look at the goals of the experiments in section 4.2.1. Then, in section 4.2.2, we take a look at the creation of the data. This refers to how the underlying datasets are constructed and how they need to be modified / extended in order to be able to use them with AnyBURL. Finally, in section 4.2.3 we show what known settings AnyBURL has and how these can be changed to achieve optimal results.

Datasets	Number of Labels	Training Set Size	*hasType* in Training	Test Set Size
Hepatitis	2	57.268	400	100
Terrorists	5	16.906	810	203
Mutagenesis	2	24.839	184	46
WebKB	7	249.963	1.492	497
FB15k-237 – Task 1	3	316.835	6.719	1.680
FB15k-237 – Task 2	4	310.461	345	86
FB15k-237 – Task 3	5	313.530	3.414	865
FB15k-237 – Task 4	2	310.257	141	38

Table 4.1: Listing of information about the different datasets. ”Number of Labels” indicates the number of different labels/entity types. The number of triples of the training set and the test set are listed in the column ”Training Set Size” and ”Test Set Size” respectively. The column ”*hasType* in Training” indicates how many triples have a *hasType* relation in the respective training set.

4.2.1 Goal of the Experiments

One goal of the experiments is to obtain a ranking of the quality of AnyBURL in the field of entity classification. To achieve this, we compare the results of AnyBURL with other algorithms that also perform entity classification. Another goal is to take a look at the reasons why AnyBURL performs the way it does. In doing so, we want to find any patterns that might help explain the results. Our final goal is to see if AnyBURL can be tuned by changing certain settings to improve the results.

4.2.2 Creation of the Data

In this part we consider how the datasets from section 4.1 have been modified and composed so that AnyBURL can work with them.

First, we look at the datasets from the paper [2]:

A dataset is split into five parts, namely (1) *training.db*, (2) *training.labels*, (3) *test.db*, (4) *test.labels* and (5) *predicate.defs*.

training.db contains the training set of the underlying knowledge graph. The individual entries are triples in functional form, i.e. *predicate(subject, object)*.

training.labels contains the set of all labels representing certain types / classes. The labels are always assigned to an entity from the training set. The form in which the labels are present is *type(entity)*.

The test set of the knowledge graph is stored in *test.db*, also in functional form. But we can neglect *test.db* for entity classification, as we are only interested in predictions that assign entities to types / classes.

test.labels is the set of labels relevant to the evaluation of the entity classification problem. This is used to check the correctness of the predictions and to calculate certain metrics for evaluation. Again, the labels are stored in the form *label(entity)*.

predicate.defs contains information about the predicates and how they are composed. These are also stored in the functional form. This means that, for example, two definitions look like this: *pred1(X, Y)*; *pred2(Y, Z)*. From this, one can deduce that objects of *pred1* serve as subjects in the function *pred2*.

AnyBURL requires inputs in the form *subject predicate object*, which means that the above data formats must be slightly modified. One can easily adapt the functional form, one translates *pred1(sub1, obj1)* to *sub1 pred1 obj1*. For the labels it is not much more difficult, one only has to insert an own predicate here, which specifies the assignment and is not yet present in the dataset (here *hasType* was used). So *label(entity)* becomes *entity hasType label*.

The training set used for AnyBURL is a concatenation of the modified (adapted to AnyBURL) data *training.db*, *training.labels* and *predicate.defs*. The test set for AnyBURL consists only of *test.labels*, because, as described above, only this set is relevant for the entity classification task.

Next we look at the datasets from the paper [7]:

Here we have two sets obtained by modifying their code and dumping the exact same splits as them. A training set and a test set. Both sets store information about entities and their associated types / classes. In this dataset, a special feature is that several types / classes are assigned to individual entities. An entry in the two sets looks like this: *Entity 0 0 1*. An *0* means that the entity does not belong to a type / class, whereas an *1* means the opposite. To put this into a form that can be used by AnyBURL, the auxiliary predicate *hasType* is also used. So *Entity 0 1 1* changes to *Entity hasType Type_2, Entity hasType Type_3*.

In order for AnyBURL to create meaningful entity classification rules, it also needs the entire Knowledge Graph in the training set. Therefore, an additional concatenation of the knowledge graph *FB15k-237* (taken from [10] and the original training set was done to obtain a training set for AnyBURL.

In section 4.1 it was already noted that in both sources of the datasets [2, 7] there are no validation sets. In order to be able to tune AnyBURL nevertheless, validation sets were created. About 10% of the *hasType* relations were randomly removed from the corresponding training set and added to the validation set.

4.2.3 Tuning the Settings

To possibly improve AnyBURL, it makes sense to change various settings in order to achieve better results. It is important to mention that the search for better settings is based on the results of the newly created validation sets presented in section 4.2.2.

The possible settings of AnyBURL can be found on the website *https://web.informatik.uni-mannheim.de/AnyBURL/* (= [11]). Now follows a short presentation of all adjustable parameters of AnyBURL:

- *Policy* – The two possible settings are *greedy policy* (=1 in the settings) on the one hand and *weighted policy* (=2 in the settings) on the other hand [11].
- *Reward* – There are three different options here. First there is *correct predictions* (=1 in the settings), then there is *correct predictions weighted by confidence with laplace smoothing* (=3 in the settings) and the last option is *correct predictions weighted by confidence with laplace smoothing divided by* $(RuleLength - 1)^2$) (=5 in the settings) [11].
- *Epsilon* – randomly assigns a nucleus with a probability of *Epsilon*; the value for *Epsilon* can be between 0,0 and 1,0 [11].
- *Threshold_Correct_Predictions* – is a value that sets the minimum limit of correct predictions of a rule for which the rule is stored.
- *Threshold_Confidence* – is a value that sets the minimum confidence limit of a rule for which the rule is stored.

Three further settings to improve AnyBURL in the area of entity classification were worked out in a dialogue with the author of the program (Dr. Christian Meilicke). These are:

- *Max_Length_Cyclic* – This specifies the maximum length of the cyclic rules to be learned. The author's recommendation was to set this value to five or seven.
- *Max_Length_Acyclic* – This specifies the maximum length of the acyclic rules to be learned. The author's recommendation was to set this value to two or three.
- *Single_Relations* – This parameter specifies that only rules with exactly this predicate should be stored. The recommendation was to set the value to *SINGLE_RELATIONS = hasType*, as only this predicate is responsible for type assignments.

Since the combination of all settings would be a multitude of different possibilities, a random search was applied instead of a grid search to find the optimal parameters [9]. For each dataset 149 random settings of the parameters were tested and always once the default settings. The threshold values for the parameters were set as follows:

- *Policy*: Either 1 or 2
- *Reward*: Either 1, 3 or 5
- *Epsilon*: Values in the range of [0, 1]
- *Threshold_Correct_Predictions*: Integer values in the range [1, 15]
- *Threshold_Confidence*: Values in the range of [0.00001, 0.3]
- *Max_Length_Cyclic*: Either 3, 5 or 7
- *Max_Length_Acyclic*: Either 1, 2 or 3
- *Single_Relations*: Either *hasType* or nothing at all

Looking at the results of the tuning (Table B.1), one has to conclude that with the settings and their threshold values known at that time, no improvement was possible compared to the default settings. Therefore, for the rest of this work, all results will be obtained with the default settings of AnyBURL.

4.3 Results

This part is about the results we have obtained with AnyBURL in the context of entity classification. A deeper analysis of why the results turn out the way they do is provided in section 4.4, looking in particular at the rules on which AnyBURL based its decisions.

The presentation of the results is divided into two parts, on the one hand the results for the datasets *Hepatitis*, *Terrorists*, *Mutagenesis* and *WebKB* (see 4.3.1) and on the other hand, for the different experiments of the dataset *FB15k-237* (see 4.3.2).

4.3.1 Results for Hepatitis, Terrorists, Mutagenesis, WebKB

For the datasets in this section, the results shown in the Table 4.2 are the results after cross validation is done. This means that this result is the average of the different cross validation splits. The results of the individual splits are shown in Table B.2. The evaluation metric in these two tables is the accuracy of the entity classification predictions.

The results show that AnyBURL is almost perfect in the *Hepatits* dataset and thus clearly beats the other applications. However, AnyBURL cannot reproduce this good result in the other datasets. In the other three datasets AnyBURL clearly lags behind, although in the dataset *WebKB* only the *TILDE* model is clearly ahead.

For these datasets, it is important to mention that the embedding baselines are most likely weak, as the models were probably not trained well enough [15]. In this case, this may mean that the results of the embedding models here are worse than they could be if the models were trained well enough.

	Datasets			
	Hepatitis	Terrorists	Mutagenesis	WebKB
TransE	0,88	0,76	0,69	0,51
DistMult	0,90	0,76	<u>0,77</u>	0,56
ComplEx	0,90	0,74	<u>0,77</u>	0,64
TILDE	0,81	<u>0,83</u>	0,75	<u>0,74</u>
AnyBURL	<u>0,99</u>	0,60	0,60	0,66

Table 4.2: Entity classification results for the datasets *Hepatitis*, *Terrorists*, *Mutagenesis* and *WebKB*. The results are the accuracies (range [0,1]) of the algorithms. The best value(s) per dataset is underlined. The values from the other algorithms are taken from [2].

4.3.2 Results for the Experiments of FB15k-237

Table 4.3 contains the results of the entity classification of AnyBURL with respect to the four different experiments (see 4.1) on the *FB15k-237* dataset. The metrics used here are the Weighted F1 Measures. The individual F1 scores as well as Precision and Recall for the individual types of experiments can be seen in Tables B.3, B.4, B.5 and B.6.

Looking at the results, we see that in the first two experiments (*Level-1, Level-2-Organizations*) AnyBURL is only slightly worse than the other applications. In experiment *Level-2-Persons* AnyBURL achieves the best value, even by a small margin. However, in experiment *Level-3-Persons* AnyBURL loses significantly.

For these experiments, strong embedding baselines were used. This can be seen from the fact that the pre-trained models were taken from [15]. However, it must be noted that the authors of [7] have an error in their results for the experiment

Level-3-Artists. However, this error was confirmed by the authors and therefore we use for experiment *Level-3-Artists* results obtained by the authors of [15].

	FB15k-237			
	Level-1	Level-2-Organizations	Level-2-Persons	Level-3-Artists
ComplEx	0,986	0,926	0,803	0,776
DistMult	0,986	0,955	0,799	0,792
ConvE	0,988	0,932	0,804	0,707
RESCAL	0,988	0,919	0,796	0,788
AnyBURL	0,971	0,920	0,868	0,671

Table 4.3: Entity classification results for the different experiments of the *FB15k-237* dataset. The results are the Weighted F1 Measures of the algorithms. The best value per experiment is underlined. The values of the other algorithms are taken from [7], except the values for *Level-3-Artists*, we got these from the authors of [15].

4.4 Discussion of the Results

To get an idea of why the results from 4.3 turn out the way they do, it makes sense to look at the basis on which AnyBURL makes certain predictions. Since AnyBURL's predictions are based on the learned rules, this is relatively easy to do, because it is possible to display which rules were used for which prediction. (This works by inserting these three values into the configuration file: *PATH_EXPLANATION = path*, *MAX_EXPLANATIONS = 10*, *AGGREGATION_TYPE = maxplus-explanations*). The Tables B.7 - B.14 contain all rules used for predictions, for the individual datasets / experiments. The individual columns in the tables mean:

- **Rule:** Contains all rules which have been used for the prediction of at least one *hasType* relation.
- **True:** The number of correct predictions made by the associated rule.
- **False:** The number of wrong predictions made by the associated rule.
- **Frequency:** Indicates how often a rule has been used for predictions (whether right or wrong). The tables are sorted in descending order according to this value.
- **Rank:** Indicates the rank of the particular rule compared to all other learned rules. It should be mentioned that all rules with the same confidence value have the same rank.

Furthermore, rows marked in light grey indicate that more wrong than right predictions were made on the basis of this rule.

Discussion of the individual Datasets

In the following, we list some anomalies per dataset/experiment and try to establish a connection between the learned rules and the result:

Hepatitis (Table B.7) The results for this dataset are very good. This can also be seen when looking at the rules used. The frequently used rules have a very good rank (best 2%) and produce no errors at all. The only error is achieved by a rule that has no body. Such a rule can rarely be useful when it comes to classification.

Terrorists (Table B.8) A look at the rules used here quickly shows why the result is not good. Almost 30% of the wrong decisions are based on a rule without a body. Furthermore, it is noticeable that some relatively specific rules are responsible for many errors. Here one could assume that either AnyBURL has learned something ”wrong” or that the underlying dataset is not sufficiently well constructed.

Mutagenesis (Table B.9) If one looks at the rules used for this dataset, it quickly becomes clear why the result did not turn out well. The only rule used for classification is a rule without a body. So one might say that AnyBURL has not been able to learn any (useful) rules for the classification task here.

WebKB (Table B.10) For this dataset, too, a major reason for AnyBURL's poor performance is the use of rules consisting of only a head. Almost 96% of all predictions here are based on this type of rule. Logically, this also results in many incorrect predictions.

FB15k-237 – Level-1 (Table B.11) The result for this experiment is the worst compared to the others, but still a very good result. This is also reflected in the rules used. Here, only very specific patterns were learned and used for the classification. The big problem of the datasets above is only a small problem here, since rules without bodies only account for about 1% of the predictions.

FB15k-237 – Level-2-Organizations (Table B.12) AnyBURL also performs well in this experiment. This is also reflected in the rules, because only one rule causes more wrong than right predictions. Furthermore, it can also be observed that almost all rules are very specific and therefore good predictions are possible.

FB15k-237 – Level-2-Persons (Table B.13) In this experiment, AnyBURL performs best, but it is striking here that about 15 % rules produce more wrong than right predictions. Most of these 15% produce significantly more wrong predictions than right ones. This could be due to incorrectly observed patterns of AnyBURL or to a possibly not so good combination of data.

FB15k-237 – Level-3-Artists (Table B.14) Comparing the results with the other algorithms we see that AnyBURL is significantly worse than the rest. This could be due to the fact that only three different rules were used for prediction and the most frequently used rule often leads to incorrect predictions (approx. 29% of its predictions are incorrect).

General Observations

In general, two main groups of errors can be identified, which most likely play a role in why AnyBURL sometimes receives poor results.

The first major problem is that AnyBURL often uses rules for prediction that have no body. This means rules of the form: $hasType(X, Type) \Leftarrow$. As a result, any prediction made using this rule does not check any properties of the entity at all, but naively assigns it this type, and this can lead to many wrong predictions.

The other problem is that AnyBURL learns "wrong" rules and then uses them for prediction. By "wrong" rule, we mean rules that for the most part produce wrong predictions and are therefore not a suitable criterion for classification. It is difficult to say whether the fault lies solely with AnyBURL or whether the composition of the datasets is not optimal. An indication that it is AnyBURL's fault would be that other algorithms (compare 4.3.1 and 4.3.2) achieve good results.

Fixing these problems by changing the settings does not seem to work here (see 4.2.3). However, there might be other, at this point unknown, settings that could improve the results significantly.

Chapter 5

Conclusion

The goal of this bachelor thesis was to classify the quality of AnyBURL in terms of entity classification by conducting various experiments. To get a good assessment, we tested AnyBURL on different entity classification datasets and discussed the results. It is important that we tested AnyBURL on different datasets because only then a reasonable classification is possible. If AnyBURL had only been tested on one dataset, the results would not be really representative of the actual performance of AnyBURL.

The results of the different experiments show that AnyBURL does not match other approaches in the area of entity classification in most datasets. Although AnyBURL performed best in some datasets (*Hepatitis*, *FB15k-237; Level-2-Persons*, the results are generally worse. This difference is particularly evident in the *Terrorists, Mutagenesis and WebKB* datasets, where AnyBURL's predictions are up to 23% worse, which is quite a big difference.

Looking more closely at AnyBURL's predictions, one can often directly see the reasons why AnyBURL performs the way it does in these datasets. In the datasets where AnyBURL performs poorly, it is often possible to identify incorrectly learned rules from the dataset that lead to incorrect predictions. Furthermore, a big problem that has also led to many wrong predictions is that AnyBURL learns rules without a body and then takes them for the prediction. This misbehaviour could not be corrected by adjusting the parameters in the settings and thus no improvement of the results was possible at that time.

Nevertheless, it is worth mentioning that AnyBURL was not developed for entity classification, but for the task of knowledge graph completion. Therefore, one cannot expect AnyBURL to work as well for entity classification, as this task is different. However, it might be possible to achieve better results in the future if AnyBURL is adapted for this task. However, at this stage and with the default implementation of AnyBURL, it is often not possible to get close to the other approaches. For this reason, AnyBURL is not yet a really good alternative in the field of entity classification.

5.1 Future Work

A task for the future could be to identify out more settings for the standard implementation of AnyBURL and test if they have a positive influence on the results. One idea would be to identify settings that prevent the learning of rules without a body. This alone would probably lead to better results in entity classification.

Furthermore, one could consider how to change the inside of AnyBURL in order to better focus AnyBURL on the task of entity classification. This would most likely be the biggest change and AnyBURL would most likely be able to achieve similar results in this area as in knowledge graph completion.

Lastly, a task for the future would be to test AnyBURL in terms of entity classification on more datasets. This would provide further insight into why exactly AnyBURL behaves the way it does. This could also be helpful for the two tasks mentioned above, as it would provide even deeper insights into the behaviour of AnyBURL. This would make it possible to search for settings in a more targeted way and to adjust AnyBURL's inside even more precisely in order to achieve optimal results.

In summary, the main task for the future is to bring AnyBURL's results closer to the results of the other approaches, so that AnyBURL is not only a good rule learner for knowledge graph completion but also for entity classification.

Bibliography

[1] Sebastijan Dumancic and Hendrik Blockeel. Clustering-based relational unsupervised representation learning with an explicit distributed representation. *arXiv preprint arXiv:1606.08658*, 2016.

[2] Sebastijan Dumancic, Alberto García-Durán, and Mathias Niepert. A comparative study of distributional and symbolic paradigms for relational learning. *arXiv preprint arXiv:1806.11391*, 2018.

[3] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48(1-4):2, 2016.

[4] Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web*, 9(1):77–129, 2018.

[5] Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. *Foundations of rule learning*. Springer Science & Business Media, 2012.

[6] Johannes Fürnkranz and Tomáš Kliegr. A brief overview of rule learning. In *International symposium on rules and rule markup languages for the semantic web*, pages 54–69. Springer, 2015.

[7] Nitisha Jain, Jan-Christoph Kalo, Wolf-Tilo Balke, and Ralf Krestel. Do embeddings actually capture knowledge graph semantics? In *European Semantic Web Conference*, pages 143–159. Springer, 2021.

[8] Mayank Kejriwal. Knowledge graphs. In *Applied Data Science in Tourism*, pages 423–449. Springer, 2022.

[9] Petro Liashchynskyi and Pavlo Liashchynskyi. Grid search, random search, genetic algorithm: a big comparison for nas. *arXiv preprint arXiv:1912.06059*, 2019.

[10] Christian Meilicke. Anytime bottom-up rule learning for knowledge graph completion, 2019.

[11] Christian Meilicke. Anyburl, 2021.

[12] Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. An introduction to anyburl. In *Joint German/Austrian Conference on Artificial Intelligence (Künstliche Intelligenz)*, pages 244–248. Springer, 2019.

[13] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2017.

[14] Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(2):1–49, 2021.

[15] Daniel Ruffinelli, Samuel Broscheit, and Rainer Gemulla. You can teach an old dog new tricks! on training knowledge graph embeddings. In *International Conference on Learning Representations*, 2019.

[16] Amit Singhal. Introducing the knowledge graph: Things, not strings, May 2012.

[17] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

[18] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[19] Yang Yang, Ryan N Lichtenwalter, and Nitesh V Chawla. Evaluating link prediction methods. *Knowledge and Information Systems*, 45(3):751–782, 2015.

[20] Lorenzo Zangari, Roberto Interdonato, Antonio Calió, and Andrea Tagarelli. Graph convolutional and attention models for entity classification in multilayer networks. *Applied Network Science*, 6(1):1–36, 2021.

[21] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018.

Appendix A

Program Code / Resources

The source code, documentation, datasets and additional test results are available at *https://github.com/jovetter/Entity-Classification-AnyBURL*

They as well as a PDF version of this thesis are also contained on the memory stick attached to this thesis.

Appendix B

Further Experimental Results

Further experimental results are listed below. The table B.1 gives a brief overview of the results of tuning AnyBURL. In the tables B.2 to B.6 the results that AnyBURL has achieved for the different datasets / experiments are listed in a more detailed form. The tables B.7 to B.14 contain rules that were used for the predictions. These tables also store information about the accuracy and frequency of these rules.

	Parameters								
Dataset	Policy	Reward	Epsilon	TH Correct Pred.	TH Confidence	Max L. C.	Max L. A.	Single Rel.	**Result**
Hepatitis	2	5	0,1	2	0,0001	3	1	no	1,0
	1	1	0,9608	13	0,0484	5	2	yes	1,0
	2	1	0,0730	12	0,0004	3	3	no	1,0
Terrorists	2	5	0,1	2	0,0001	3	1	no	0,8375
	2	5	0,0656	14	0,0059	3	1	yes	0,8375
	1	1	0,8990	5	0,0051	5	1	no	0,8375
Mutagenesis	2	5	0,1	2	0,0001	3	1	no	0,7778
	1	3	0,9797	11	0,0160	7	1	yes	0,7778
	2	5	0,5359	9	0,01504	7	2	no	0,7778
WebKB	2	5	0,1	2	0,0001	3	1	no	0,8092
	1	3	0,7642	15	0,0045	3	1	no	0,8026
	1	3	0,6604	10	0,0054	3	1	no	0,7961
Level-1	2	5	0,1	2	0,0001	3	1	no	0,9702
	2	1	0,7255	4	0,0357	3	2	no	0,9675
	1	3	0,0188	5	0,02307	3	1	yes	0,9675
Level-2-Organizations	2	5	0,1	2	0,0001	3	1	no	0,9002
	2	3	0,4208	11	0,0912	7	2	yes	0,9002
	2	3	0,8135	10	0,0917	3	2	no	0,9002
Level-2-Persons	2	5	0,1	2	0,0001	3	1	no	0,8666
	2	5	0,0197	4	0,2759	5	2	no	0,8541
	1	1	0,8944	6	0,1591	5	1	yes	0,8494
Level-3-Artists	2	5	0,1	2	0,0001	3	1	no	0,8645
	1	5	0,5055	2	0,0307	5	3	no	0,8645
	2	3	0,7309	8	0,0306	7	2	yes	0,8645

Table B.1: The best three tuning results per dataset are shown here. If several datasets achieve the same result as the default setting, the default setting is placed first. In the column *Result* the value is always interpreted as Accuracy for the first four datasets and as Weighted F1 Measure for the last four. The abbreviations of the column names mean: TH = Threshold; Max L. C. = Max Length Cyclic; Max L. A. = Max Length Acyclic; Rel. = Relation. The individual tables for each dataset with all values can be found here:
https://github.com/jovetter/Entity-Classification-AnyBURL.

	Datasets			
	Hepatitis	Terrorists	Mutagenesis	WebKB
Split 1	0,99	0,549019608	0,630434783	0,668831169
Split 2	1	0,593301435	0,586956522	0,70
Split 3	1	0,63546798	0,565217391	0,630669546
Split 4	1	0,592783505	0,695652174	0,62369338
Split 5	1	0,60591133	0,52173913	-
Result	0,998	0,595296772	0,60	0,655798524

Table B.2: Full entity classification results for the datasets *Hepatitis*, *Terrorists*, *Mutagenesis* and *WebKB*. [2].

	FB15k-237; Level-1		
	Person	Organization	Product
Precision	0,9529	0,9907	0,9968
Recall	0,9989	0,9120	0,9783
F1-Score	0,9754	0,9497	0,9875
Weighted-F1		**0,9706**	

Table B.3: Precision, Recall and F1-Score for the individual types of the *Level-1* experiment, as well as the Weighted F1 Measure for the whole experiment.

	FB15k-237; Level-2-Organizations			
	Musical Organization	Party	Enterprises	Non-Governmental Institution
Precision	1,0	1,0	0,9167	0,7333
Recall	1,0	0,8889	0,8919	0,8462
F1-Score	1,0	0,9412	0,9041	0,7857
Weighted-F1			**0,9202**	

Table B.4: Precision, Recall and F1-Score for the individual types of the *Level-2-Organizations* experiment, as well as the Weighted F1 Measure for the whole experiment.

	FB15k-237; Level-2-Persons				
	Artist	Officeholder	Writer	Scientist	Politician
Precision	0,9240	0,9474	0,8265	0,8889	0,9118
Recall	0,8360	0,75	0,9719	0,5333	0,5536
F1-Score	0,8778	0,8372	0,8933	0,6667	0,6889
Weighted-F1			**0,8678**		

Table B.5: Precision, Recall and F1-Score for the individual types of the *Level-2-Persons* experiment, as well as the Weighted F1 Measure for the whole experiment.

	FB15k-237; Level-3-Artists	
	Painter	Photographer
Precision	1,0	0,6563
Recall	0,3529	1,0
F1-Score	0,5217	0,7925
Weighted-F1	**0,6713**	

Table B.6: Precision, Recall and F1-Score for the individual types of the *Level-3-Artists* experiment, as well as the Weighted F1 Measure for the whole experiment.

Rules	True	False	Frequency	Rank
hasType(X,Y) ⇐ B_rel11(A,X), B_rel11(A,B), hasType(B,Y)	81	0	81/100	97/5602
hasType(X,TYPE_C) ⇐ B_rel11(B_ID_5,X)	18	0	18/100	82/5602
hasType(X,TYPE_C) ⇐	0	1	1/100	273/5602

Table B.7: The table shows all rules, sorted in descending order by frequency, that were used to predict the types in the *Hepatitis* dataset. The accuracy of AnyBURL in this dataset was 0,998.

Rules	True	False	Frequency	Rank
hasType(X,Bombing) ⇐	28	60	88/204	639/4667
hasType(X,Bombing) ⇐ HasEventFeature(X,EventFeature29)	17	1	18/204	266/4667
hasType(X,Y) ⇐ PerformedBySameOrg(X,A), PerformedBySameOrg(B,A),hasType(B,Y)	9	8	17/204	377/4667
hasType(X,Y) ⇐ CoLocatedEvent(X,A), CoLocatedEvent(B,A), hasType(B,Y)	2	9	11/204	647/4667
hasType(X,Y) ⇐ PerformedBySameOrg(A,X), hasType(A,Y)	5	2	7/204	518/4667
hasType(X,Y) ⇐ CoLocatedEvent(A,X), CoLocatedEvent(A,B), hasType(B,Y)	4	2	6/204	687/4667
hasType(X,Bombing) ⇐ CoLocatedEvent(X,Http_counterterror_mindswap_org_2005_ict _events_owl_Hamas_20011201)	5	0	5/204	213/4667
hasType(X,Weapon_Attack) ⇐ HasEventFeature(X,EventFeature37)	5	0	5/204	268/4667
hasType(X,Y) ⇐ CoLocatedEvent(X,A), PerformedBySameOrg(B,A), hasType(B,Y)	2	3	5/204	533/4667
hasType(X,Y) ⇐ CoLocatedEvent(A,X), PerformedBySameOrg(A,B), hasType(B,Y)	1	4	5/204	579/4667
hasType(X,Y) ⇐ PerformedBySameOrg(X,A), hasType(A,Y)	3	1	4/204	526/4667
hasType(X,Y) ⇐ CoLocatedEvent(A,X), hasType(A,Y)	3	0	3/204	702/4667
hasType(X,Bombing) ⇐ HasEventFeature(X,EventFeature49)	2	1	3/204	200/4667
hasType(X,Y) ⇐ PerformedBySameOrg(A,X), PerformedBySameOrg(A,B), hasType(B,Y)	2	1	3/204	455/4667
hasType(X,Bombing) ⇐ CoLocatedEvent(X,Http_counterterror_mindswap_org_2005_ict _events_owl_Hamas_19941225)	2	0	2/204	131/4667
hasType(X,Weapon_Attack) ⇐ HasEventFeature(X,EventFeature66)	2	0	2/204	285/4667
hasType(X,Kidnapping) ⇐ HasEventFeature(X,EventFeature69)	2	0	2/204	302/4667
hasType(X,Bombing) ⇐ HasEventFeature(X,EventFeature48)	2	0	2/204	429/4667
hasType(X,Weapon_Attack) ⇐ CoLocatedEvent(Http_counterterror_mindswap_org_2005_ict _events_owl_Unknown_19980328,X)	1	0	1/204	115/4667
hasType(X,Bombing) ⇐ CoLocatedEvent(X,Http_counterterror_mindswap_org_2005_ict _events_owl_Sendero_Luminoso_20020320)	1	0	1/204	227/4667
hasType(X,Bombing) ⇐ CoLocatedEvent(X,Http_counterterror_mindswap_org_2005_ict _events_owl_Unknown_19900804)	1	0	1/204	227/4667
hasType(X,Weapon_Attack) ⇐ HasEventFeature(X,EventFeature3)	1	0	1/204	274/4667
hasType(X,Weapon_Attack) ⇐ CoLocatedEvent(Http_counterterror_mindswap_org_2005_ict _events_owl_No_Affiliation_20020917,X)	1	0	1/204	286/4667
hasType(X,Weapon_Attack) ⇐ HasEventFeature(X,EventFeature57)	1	0	1/204	325/4667
hasType(X,Weapon_Attack) ⇐ HasEventFeature(X,EventFeature39)	1	0	1/204	337/4667
hasType(X,Bombing) ⇐ HasEventFeature(X,EventFeature0)	1	0	1/204	510/4667
hasType(X,Weapon_Attack) ⇐ HasEventFeature(X,EventFeature72)	1	0	1/204	578/4667
hasType(X,Weapon_Attack) ⇐ HasEventFeature(X,EventFeature86)	1	0	1/204	690/4667
hasType(X,Y) ⇐ CoLocatedEvent(X,A), hasType(A,Y)	1	0	1/204	727/4667
hasType(X,Bombing) ⇐ CoLocatedEvent(X,Http_counterterror_mindswap_org_2005_ict _events_owl_Unknown_20010801)	0	1	1/204	195/4667
hasType(X,Bombing) ⇐ HasEventFeature(X,EventFeature36)	0	1	1/204	374/4667
hasType(X,Weapon_Attack) ⇐ HasEventFeature(X,EventFeature46)	0	1	1/204	453/4667
hasType(X,Bombing) ⇐ HasEventFeature(X,EventFeature53)	0	1	1/204	462/4667
hasType(X,Bombing) ⇐ HasEventFeature(X,EventFeature62)	0	1	1/204	525/4667

Table B.8: The table shows all rules, sorted in descending order by frequency, that were used to predict the types in the *Terrorists* dataset. The accuracy of AnyBURL in this dataset was 0,595.

Rules	**True**	**False**	**Frequency**	**Rank**
hasType(X,Mutagenic_yes) $\Leftarrow$	29	17	46/46	69/975

Table B.9: The table shows all rules, sorted in descending order by frequency, that were used to predict the types in the *Mutagenesis* dataset. The accuracy of AnyBURL in this dataset was 0,60.

Rules	**True**	**False**	**Frequency**	**Rank**
hasType(X,Person) $\Leftarrow$	174	90	264/462	7919/36026
hasType(X,Student) $\Leftarrow$	126	50	176/462	11225/36026
hasType(X,Person) $\Leftarrow$ HasWord(X,WordWashington)	7	1	8/462	17151/36026
hasType(X,Student) $\Leftarrow$ HasWord(X,WordWashington)	1	6	7/462	20692/36026
hasType(X,Student) $\Leftarrow$ HasWord(X,WordWisconsin)	1	2	3/462	14602/36026
hasType(X,Course) $\Leftarrow$	0	2	2/462	21044/36026
hasType(X,Person) $\Leftarrow$ HasWord(X,WordWisc)	0	1	1/462	12842/36026
hasType(X,Y) $\Leftarrow$ HasWord(X,A), HasWord(B,A), hasType(B,Y)	0	1	1/462	23789/36026

Table B.10: The table shows all rules, sorted in descending order by frequency, that were used to predict the types in the *WebKB* dataset. The accuracy of AnyBURL in this dataset was 0,656.

Rules	True	False	Frequency	Rank
hasType(X,Y) ⇐ /people/person/spouse_s./people/marriage/type_of_union(X,A), /people/person/spouse_s./people/marriage/type_of_union(B,A), hasType(B,Y)	215	0	215/1680	1645/34272
hasType(X,Y) ⇐ /film/actor/film./film/performance/film(X,A), /film/actor/film./film/performance/film(B,A), hasType(B,Y)	138	0	138/1680	1617/34272
hasType(X,Y) ⇐ /base/popstra/celebrity/friendship./base/popstra/friendship/participant(A,X), /award/award_nominee/award_nominations./award/award_nomination/award_nominee(A,B), hasType(B,Y)	72	3	75/1680	1273/34272
hasType(X,Y) ⇐ /film/actor/film./film/performance/film(A,X), /film/film/produced_by(B,A), hasType(B,Y)	61	0	61/1680	2842/34272
hasType(X,Person) ⇐ /people/person/profession(X,Voice Actor)	50	0	50/1680	1453/34272
hasType(X,Y) ⇐ /media_common/netflix_genre/titles(A,X), /film/film/genre(B,A), hasType(B,Y)	26	22	48/1680	3070/34272
hasType(X,Y) ⇐ /soccer/football_team/current_roster./soccer/football_roster_position/position(X,A), /soccer/football_team/current_roster./sports/sports_team_roster/position(B,A), hasType(B,Y)	43	0	43/1680	2115/34272
hasType(X,Organization) ⇐ /sports/sports_team/sport(X,Football)	41	0	41/1680	1698/34272
hasType(X,Y) ⇐ /base/popstra/celebrity/dated./base/popstra/dated/participant(A,X), hasType(A,Y)	41	0	41/1680	1280/34272
hasType(X,Person) ⇐ /people/person/religion(X,Catholicism)	39	0	39/1680	1405/34272
hasType(X,Y) ⇐ /people/person/profession(X,A), /people/person/profession(B,A), hasType(B,Y)	37	0	37/1680	1766/34272
hasType(X,Organization) ⇐ /education/educational_institution/school_type(X,Public university)	34	0	34/1680	1681/34272
hasType(X,Y) ⇐ /film/film/story_by(A,X), /film/actor/film./film/performance/film(B,A), hasType(B,Y)	31	0	31/1680	1107/34272
hasType(X,Y) ⇐ /award/ranked_item/appears_in_ranked_lists./award/ranking/list(X,A), /award/ranked_item/appears_in_ranked_lists./award/ranking/list(B,A), hasType(B,Y)	31	0	31/1680	2256/34272
hasType(X,Y) ⇐ /sports/pro_athlete/teams./sports/sports_team_roster/team(A,X), /sports/pro_athlete/teams./sports/sports_team_roster/team(A,B), hasType(B,Y)	30	0	30/1680	2006/34272
hasType(X,Person) ⇐ /people/person/profession(X,Screenwriter)	29	0	29/1680	1625/34272
hasType(X,Y) ⇐ /film/actor/film./film/performance/film(A,X), /award/award_winning_work/awards_won./award/award_honor/award_winner(B,A), hasType(B,Y)	29	0	29/1680	3080/34272
hasType(X,Y) ⇐ /education/educational_institution/students_graduates./education/education/student(A,X), /education/educational_institution/students_graduates./education/education/student(A,B), hasType(B,Y)	28	0	28/1680	1658/34272
hasType(X,Y) ⇐ /organization/role/leaders./organization/leadership/organization(A,X), /organization/role/leaders./organization/leadership/organization(A,B), hasType(B,Y)	27	0	27/1680	2195/34272
hasType(X,Product) ⇐ /film/film/genre(X,Period piece)	25	0	25/1680	2746/34272
hasType(X,Person) ⇐ /people/person/profession(X,Comedian)	22	0	22/1680	1509/34272
hasType(X,Y) ⇐ /music/record_label/artist(X,A), /music/record_label/artist(B,A), hasType(B,Y)	22	0	22/1680	1164/34272
hasType(X,Product) ⇐ /film/film/genre(X,Crime Fiction)	21	0	21/1680	2928/34272
hasType(X,Y) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award(X,A), /award/award_category/winners./award/award_honor/award_winner(A,B), hasType(B,Y)	8	13	21/1680	2422/34272
hasType(X,Y) ⇐ /base/americancomedy/celebrity_impressionist/celebrities_impersonated(A,X), hasType(A,Y)	20	0	20/1680	844/34272
hasType(X,Product) ⇐ /film/film_distributor/films_distributed./film/film_film_distributor_relationship/film(Universal Studios,X)	19	0	19/1680	1002/34272
hasType(X,Person) ⇐ /common/topic/webpage./common/webpage/category(X,Official Website)	1	18	19/1680	9828/34272
hasType(X,Product) ⇐ /film/film/genre(X,Comedy-drama)	19	0	19/1680	2470/34272
hasType(X,Person) ⇐ /people/person/religion(X,Atheism)	19	0	19/1680	1285/34272
hasType(X,Person) ⇐	12	6	18/1680	4851/34272
hasType(X,Person) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award(X,Primetime Emmy Award for Outstanding Lead Actor - Miniseries or a Movie)	17	0	17/1680	1013/34272
hasType(X,Y) ⇐ /influence/influence_node/peers./influence/peer_relationship/peers(X,A), hasType(A,Y)	15	0	15/1680	1016/34272
hasType(X,Organization) ⇐ /sports/professional_sports_team/draft_picks./sports/sports_league_draft_pick/school(Golden State Warriors,X)	15	0	15/1680	1481/34272
hasType(X,Y) ⇐ /people/person/sibling_s./people/sibling_relationship/sibling(A,X), hasType(A,Y)	14	0	14/1680	887/34272
hasType(X,Person) ⇐ /base/eating/practicer_of_diet/diet(X,Vegetarianism)	14	0	14/1680	1057/34272
hasType(X,Y) ⇐ /base/x2010fifaworldcupsouthafrica/world_cup_squad/current_world_cup_squad./base/x2010fifaworldcupsouthafrica/current_world_cup_squad/current_club(A,X), hasType(A,Y)	14	0	14/1680	1693/34272
hasType(X,Organization) ⇐ /education/educational_institution/students_graduates./education/education/major_field_of_study(X,Business Administration)	12	0	12/1680	792/34272
hasType(X,Y) ⇐ /sports/sports_team/roster./american_football/football_historical_roster_position/position_s(X,A), /american_football/football_team/current_roster./sports/sports_team_roster/position(B,A), hasType(B,Y)	11	0	11/1680	1240/34272
hasType(X,Organization) ⇐ /sports/sports_team/colors(X,Black (Color) #269)	10	0	10/1680	1240/34272
hasType(X,Person) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award(X,Primetime Emmy Award for Outstanding Lead Actress - Miniseries or a Movie)	10	0	10/1680	798/34272

Table B.11: The table shows all rules, sorted in descending order by frequency, that were used to predict the types in the *FB15k-237; Level-1* dataset. The Weighted F1 Measure of AnyBURL in this dataset was 0,9706. The table was shortened to 40 entries. The whole table can be found at *https://github.com/jovetter/Entity-Classification-AnyBURL.*

Rules	**True**	**False**	**Frequency**	**Rank**
hasType(X,Party) ⇐ /music/performance_role/regular_performances./music/group_membership/group(Percussion,X)	14	0	14/86	3459/36548
hasType(X,Musical_Organization) ⇐ /music/performance_role/regular_performances./music/group_membership/group(Bass guitar,X)	9	0	9/86	3565/36548
hasType(X,Party) ⇐ /government/political_party/politicians_in_this_party./government/political_party_tenure/politician(X,A)	8	0	8/86	3492/36548
hasType(X,Enterprises) ⇐ /business/job_title/people_with_this_title./business/employment_tenure/company(Chief Financial Officer,X)	8	0	8/86	6678/36548
hasType(X,Y) ⇐ /business/business_operation/industry(X,A), hasType(A,Y)	7	0	7/86	4050/36548
hasType(X,Musical_Organization) ⇐ /music/performance_role/regular_performances./music/group_membership/group(Drums (Musical instrument),X)	4	0	4/86	3967/36548
hasType(X,Non_Governmental_Institution) ⇐ /education/educational_institution/school_type(X,Private university)	3	1	4/86	12071/36548
hasType(X,Enterprises) ⇐ /common/topic/webpage./common/webpage/category(X,Official Website)	2	2	4/86	25269/36548
hasType(X,Non_Governmental_Institution) ⇐ /sports/professional_sports_team/draft_picks./sports/sports_league_draft_pick/school(Golden State Warriors,X)	2	2	4/86	13526/36548
hasType(X,Y) ⇐ /organization/role/leaders./organization/leadership/organization(A,X), /business/job_title/people_with_this_title./business/employment_tenure/company(A,B), hasType(B,Y)	1	3	4/86	13764/36548
hasType(X,Enterprises) ⇐ /business/business_operation/industry(X,Video game)	3	0	3/86	4721/36548
hasType(X,Enterprises) ⇐ /base/schemastaging/organization_extra/phone_number./base/schemastaging/phone_sandbox/service_location(X,United Kingdom)	2	0	2/86	4050/36548
hasType(X,Enterprises) ⇐ /base/schemastaging/organization_extra/phone_number./base/schemastaging/phone_sandbox/service_location(X,Canada)	2	0	2/86	3418/36548
hasType(X,Non_Governmental_Institution) ⇐ /location/statistical_region/religions./location/religion_percentage/religion(North Carolina,X)	2	0	2/86	4766/36548
hasType(X,Non_Governmental_Institution) ⇐ /people/person/religion(A,X)	2	0	2/86	6712/36548
hasType(X,Enterprises) ⇐ /organization/organization/headquarters./location/mailing_address/state_province_region(X,California)	1	1	2/86	13411/36548
hasType(X,Enterprises) ⇐	1	0	1/86	7070/36548
hasType(X,Enterprises) ⇐ /award/award_category/winners./award/award_honor/award_winner(Peabody Award,X)	1	0	1/86	18827/36548
hasType(X,Enterprises) ⇐ /business/business_operation/industry(X,Airline (Industry))	1	0	1/86	814/36548
hasType(X,Enterprises) ⇐ /business/job_title/people_with_this_title./business/employment_tenure/company(President,X)	1	0	1/86	12101/36548
hasType(X,Enterprises) ⇐ /business/job_title/people_with_this_title./business/employment_tenure/company(A,X)	1	0	1/86	13504/36548
hasType(X,Enterprises) ⇐ /organization/organization/headquarters./location/mailing_address/state_province_region(X,Washington)	1	0	1/86	5940/36548
hasType(X,Enterprises) ⇐ /organization/role/leaders./organization/leadership/organization(Managing Director,X)	1	0	1/86	7826/36548

Table B.12: The table shows all rules, sorted in descending order by frequency, that were used to predict the types in the *FB15k-237; Level-2-Organizations* dataset. The Weighted F1 Measure of AnyBURL in this dataset was 0,9202.

Rule	True	False	Frequency	Rank
hasType(X,Writer) ⇐ /people/person/profession(X,Screenwriter)	109	5	114/865	2882/36108
hasType(X,Writer) ⇐ /people/person/profession(X,Writer)	91	1	92/865	2285/36108
hasType(X,Y) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award(X,A), /award/award_category/winners./award/award_honor/award_winner(A,B), hasType(B,Y)	10	31	41/865	6424/36108
hasType(X,Artist) ⇐ /people/person/profession(X,Songwriter)	38	0	38/865	1892/36108
hasType(X,Artist) ⇐ /people/person/profession(X,Musician (Profession))	34	1	35/865	2068/36108
hasType(X,Artist) ⇐ /music/instrument/instrumentalists(Guitar,X)	31	0	31/865	1738/36108
hasType(X,Writer) ⇐ /people/person/profession(X,Author (Profession))	25	4	29/865	2643/36108
hasType(X,Artist) ⇐ /people/person/profession(X,Guitarist)	25	3	28/865	1617/36108
hasType(X,Y) ⇐ /award/award_winner/awards_won./award/award_honor/award_winner(X,A), hasType(A,Y)	21	5	26/865	5573/36108
hasType(X,Y) ⇐ /government/politician/government_positions_held./government/government_position_held/basic_title(X,A), /government/politician/government_positions_held./government/government_position_held/basic_title(B,A), hasType(B,Y)	5	20	25/865	3284/36108
hasType(X,Politician) ⇐ /people/person/profession(X,Politician)	24	0	24/865	2527/36108
hasType(X,Artist) ⇐ /people/person/profession(X,Singer-songwriter)	21	1	22/865	1803/36108
hasType(X,Y) ⇐ /people/person/profession(X,A), /tv/non_character_role/tv_regular_personal_appearances./tv/tv_regular_personal_appearance/person(A,B), hasType(B,Y)	20	2	22/865	3426/36108
hasType(X,Y) ⇐ /influence/influence_node/influenced_by(X,A), hasType(A,Y)	11	10	21/865	4409/36108
hasType(X,Y) ⇐ /people/person/profession(X,A), /people/person/profession(B,A), hasType(B,Y)	1	19	20/865	6699/36108
hasType(X,Writer) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award(X,Golden Globe Award for Best Screenplay - Motion Picture)	18	0	18/865	1802/36108
hasType(X,Writer) ⇐ /people/person/profession(X,Film Producer)	5	13	18/865	5536/36108
hasType(X,Artist) ⇐ /people/person/profession(X,Composer)	18	0	18/865	2155/36108
hasType(X,Writer) ⇐ /people/person/profession(X,Film Director)	13	4	17/865	4203/36108
hasType(X,Artist) ⇐ /people/person/profession(X,Artist)	12	2	14/865	2433/36108
hasType(X,Y) ⇐ /music/group_member/membership./music/group_membership/role(X,A), hasType(A,Y)	13	0	13/865	914/36108
hasType(X,Y) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award_nominee(A,X), hasType(A,Y)	7	5	12/865	5986/36108
hasType(X,Writer) ⇐ /people/person/profession(X,Television producer)	7	4	11/865	4606/36108
hasType(X,Artist) ⇐ /people/person/profession(X,Cinematographer)	9	0	9/865	2660/36108
hasType(X,Writer) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award(X,Academy Award for Best Original Screenplay)	8	0	8/865	2293/36108
hasType(X,Artist) ⇐ /people/person/profession(X,Record producer)	6	1	7/865	2054/36108
hasType(X,Y) ⇐ /music/instrument/instrumentalists(A,X), /music/instrument/instrumentalists(A,B), hasType(B,Y)	0	7	7/865	3564/36108
hasType(X,Writer) ⇐ /people/person/profession(X,Playwright)	6	0	6/865	2129/36108
hasType(X,Officeholder) ⇐ /people/person/profession(X,Politician)	5	1	6/865	4281/36108
hasType(X,Writer) ⇐ /people/person/profession(X,Lyricist (Profession))	6	0	6/865	4821/36108
hasType(X,Writer) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award(X,Golden Globe Award for Best Director - Motion Picture)	4	1	5/865	3566/36108
hasType(X,Artist) ⇐ /music/genre/artists(Pop music,X)	5	0	5/865	3056/36108
hasType(X,Y) ⇐ /influence/influence_node/influenced_by(A,X), hasType(A,Y)	4	1	5/865	4426/36108
hasType(X,Y) ⇐ /people/person/employment_history./business/employment_tenure/company(X,A), /education/educational_institution/students_graduates./education/education/student(A,B), hasType(B,Y)	2	2	4/865	5474/36108
hasType(X,Writer) ⇐ /people/person/gender(X,Male)	3	1	4/865	7525/36108
hasType(X,Writer) ⇐	0	4	4/865	5953/36108
hasType(X,Y) ⇐ /influence/influence_node/influenced_by(X,A), /influence/influence_node/influenced_by(B,A), hasType(B,Y)	2	2	4/865	5109/36108
hasType(X,Writer) ⇐ /people/person/religion(X,Judaism)	0	4	4/865	5038/36108
hasType(X,Writer) ⇐ /people/ethnicity/people(Jewish people,X)	2	2	4/865	5706/36108
hasType(X,Writer) ⇐ /people/person/profession(X,Comedian)	2	2	4/865	4758/36108

Table B.13: The table shows all rules, sorted in descending order by frequency, that were used to predict the types in the *FB15k-237; Level-2-Persons* dataset. The Weighted F1 Measure of AnyBURL in this dataset was 0,8678. The table was shortened to 40 entries. The whole table can be found at *https://github.com/jovetter/Entity-Classification-AnyBURL*.

Rule	True	False	Frequency	Rank
hasType(X,Y) ⇐ /people/person/profession(X,A), /people/person/profession(B,A), hasType(B,Y)	17	7	24/38	15981/35707
hasType(X,Photographer) ⇐ /people/person/profession(X,Cinematographer)	10	1	11/38	3302/3507
hasType(X,Photographer) ⇐ /award/award_nominee/award_nominations./award/award_nomination/award(X,Satellite Award for Best Cinematography)	3	0	3/38	1471/35707

Table B.14: The table shows all rules, sorted in descending order by frequency, that were used to predict the types in the *FB15k-237; Level-3-Artists* dataset. The Weighted F1 Measure of AnyBURL in this dataset was 0,6713.

Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Master-/Bachelorarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

J. Vetter

Mannheim, den 27.06.2022 Unterschrift