Non-Parametric Path Based Model for Taxonomy Induction in Knowledge Graphs

Marcin Pietrasik$^{1,2[0000-0001-7559-8658]}$, Marek Reformat$^{2,3[0000-0003-4783-0717]}$, and Anna Wilbik$^{1[0000-0002-1989-0301]}$

[1] Department of Advanced Computing Sciences, Maastricht University
Maastricht, The Netherlands
[2] Department of Electrical and Computer Engineering, University of Alberta
Edmonton, Canada
[3] Institute of Information Technology, University of Social Sciences, Łódź, Poland
`marcin.pietrasik@maastrichtuniversity.nl`

Abstract. The proliferation of large-scale knowledge graphs has spurred increasing interest in the development of automated tools for reasoning with and extracting information from their underlying structures. One piece of information that is critical for understanding a knowledge graph is its class taxonomy, a hierarchical representation of the knowledge graph's type information. As such, a research area has arisen in recent years around the automated induction of class taxonomies from the otherwise flat knowledge graph. In this paper, we add to this area by proposing a non-parametric path based model that leverages frequency and co-occurrence to guide an optimization process that learns a hierarchical representation of the knowledge graph's classes. As the name suggests, it is non-parametric and thus assumes no prior structure over the taxonomy. Furthermore, our model does not require the explicit calculation of class generality nor does it rely on outside domain knowledge. The approach is closely related to work done in the field of hierarchical topic modelling, specifically hierarchical Latent Dirichlet Allocation which our model may be viewed as a discrete adaptation of. We evaluate our model on three real-world datasets and find that it tends to outperform existing methods in the literature. Finally, we discuss the drawbacks and limitations of our model as well as avenues for future work and extensions.

Keywords: knowledge graph · taxonomy induction · ontology.

1 Introduction

Knowledge graphs are an approach to knowledge representation that leverage principles of graph theory to structure information. At the backbone of the knowledge graph is the triple, which connects two entities together via a predicate. This structure is analogous to directed graphs in which two vertices are connected via an edge. As such, a set of triples defines a knowledge graph like the sets of vertices and edges define a directed graph. Recent years have demonstrated the widespread use of knowledge graphs with large-scale knowledge bases such as DBpedia [13], YAGO [22], and WikiData [33] all using knowledge graphs as their underlying representation structure. The core of these knowledge graphs is the class taxonomy. The class taxonomy organizes the knowledge graph's type information hierarchically such that more general classes appear at the top of the hierarchy and less general classes appear lower down. These relationships are captured in the taxonomy's subsumption axioms. For instance, the class `dog` may be subsumed by the class `mammal` which in turn may be subsumed by the class `animal`. As we can see, this structure usually takes the form of a rooted tree, although some knowledge graphs relax this to any directed acyclic graph.

One of the challenges in working with large-scale knowledge graphs is constructing the class taxonomy. Manual construction is a laborious process that requires expert curation [21,31]. As such, automated methods have received considerable research attention in recent years [9,29,35,34,23]. However, the existing methods are not infallible, and induce taxonomies that are not reliable for practical purposes. In this paper, we aim to advance this research area by proposing a non-parametric path based model for inducing a taxonomy of class subsumption axioms. Our model, in addition to being non-parametric, does not make any assumptions about the content of the classes. In this regard, it is language-independent, context-independent, and non-reliant on external domain knowledge. The intuition behind our model is that classes are assigned to a node on the taxonomy tree and subject entities travel down a path through this tree, associating with themselves all the classes they come across. This reduces the problem to finding the optimal entity paths and class allocation on the taxonomy. Such a formulation is similar in spirit to hierarchical topic modelling, specifically hierarchical Latent Dirichlet Allocation (hLDA) [4]. The main difference between the two models stems from the discrete nature of our proposed model in contrast to the probabilistic graphical model of hLDA. This has implications for the way the generative processes are set up and optimized; namely, discrete optimization for our model and Bayesian inference for hLDA. We demonstrate the viability of our model on three real-world datasets and compare the performance against existing methods. The results indicate that our model tends to outperform existing methods in the quality of induced taxonomies.

2 Related Work

Although our proposed model is applied for the purpose of class taxonomy induction in knowledge graphs, its formulation is that of a tag hierarchy induction model. Tag hierarchy induction is a related problem in which a hierarchy of tags is constructed based on the documents that they annotate. These documents are oftentimes posts from social media platforms and the tags are annotations provided by the users. For instance, the social networking services X and Instagram allow users to annotate their posts with hashtags. As can be seen, this format of documents and tags is not equivalent to the triple structure of knowledge graphs.

In the subsequent section, we demonstrate how we adapt the triple format to be compatible with classical tag hierarchy induction methods. First, however, we discuss the existing work on tag hierarchy induction models that operate on document-tag pairs and class taxonomy induction models that operate on knowledge graphs.

2.1 Tag Hierarchy Induction Methods

Previous work addressing the problem of tag hierarchy induction can be broadly divided into two categories: those that utilize domain knowledge in their approach and those that do not. Domain knowledge in this task refers to any information other than the documents and the tags that describe them. Approaches that do not utilize domain knowledge – which we refer to as frequency-based models – rely on the relative frequencies that tags appear at and the co-occurrences between them. Our model belongs to the latter of these two categories. As such, we focus on frequency-based models in our overview of the related works.

In a canonical approach, Heymann and Garcia-Molina [9] proposed an approach that uses cosine similarity to calculate tag generality. Specifically, tags are assigned a pairwise similarity score based on the frequency with which they annotate the same documents. This score is used to build a tag similarity graph. The distance between tags in this graph is used as the basis for assigning tag generality. A hierarchy is constructed by greedily adding tags as children to the node in the tree that is most similar. In a competing approach, Schmitz [29] proposed a method inspired by Sanderson and Croft [28] that leverages subsumption rules for identifying the class connections in the induced tree. Similar to Heyman and Garcia-Molina, tag frequencies and co-occurrences are used to calculate these rules which are subsequently filtered to account for "idiosyncratic vocabulary". These rules form a directed graph which is then pruned to create a tree. The method proposed by Li et al. [14] avoids explicitly computing tag generality by employing agglomerative clustering for hierarchy generation. In another clustering-based approach, Wang et al. [35] induce a hierarchy by repeated application of the k-medoids algorithm with a custom distance metric based on textual similarities. Similarly, Wetzel et al. [37] generated embeddings for tags to serve as the basis for a similarity metric for hierarchical clustering. Gu et al. [8] and Wang et al. [34] both use a two phase approach in which a tag hierarchy is first induced using a strictly frequency-based approach and then optimized using domain knowledge in the form of an existing hierarchy.

2.2 Class Taxonomy Induction

In an early method, Völker and Niepert [32] use association rule mining on a knowledge graph's transaction table to induce class subsumption axioms in a greedy approach. Transaction tables describe, among other information, the class membership of entities. Nickel et al. [19] extend RESCAL [20], a knowledge graph representation model based on factorization, and apply OPTICS

[3], a density based hierarchical clustering algorithm. Ristoski et al. [26] leverage the spatial information of entities to learn hierarchical representations between classes by applying the intuition that you can infer class relationships through the neighbourhoods of their constituent entities' embeddings. More recently, Pietrasik and Reformat [23,24] achieved state-of-the-art results with a bottom-up approach that leverages class frequencies and co-occurrences to build a taxonomy by adding leaf nodes to an existing tree. Despite being approaches to hierarchical clustering at their core, the models proposed by Zhang et al. [38] and Pietrasik et al. [25] can induce a class taxonomy through selective sampling of their tag distributions. Martel and Zouaq [16] first embed a knowledge graph before applying hierarchical agglomeration clustering to obtain a hierarchical tree structure. Classes are then mapped to this hierarchy to obtain a class taxonomy. Most recently, Horta et al. [10] described a neural approach based on recent advances in computer vision.

3 Problem Formulation

A knowledge graph is expressed as a set of triples that relate a subject entity, s, to an object entity, o, via a predicate, p. We thus define a knowledge graph $\mathcal{KG}$ as the set $\mathcal{KG} := \{\langle s, r, o\rangle \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}\}$ where s and o are entities belonging to the set of entities, $s, o \in \mathcal{E}$, and r is the predicate belonging to the set of predicates, $r \in \mathcal{R}$. When interpreted as a whole, this set forms a directed graph with entities corresponding to vertices and predicates corresponding to edges. Recall that our proposed model is formulated as a tag hierarchy induction method and thus takes in document-tag pairs as input. As such, it is necessary to transform a knowledge graph from its triple representation into one that is compatible with our model. To do this, we adopt the tuple representation proposed by Pietrasik and Reformat [23] in which predicate-object pairs, $\langle r, o\rangle$, are interpreted as tags that annotate a subject entity in the knowledge graph. Specifically, we define a tag as a predicate-object pair $t := \langle r, o\rangle$ such that t is an element of the set of tags, $t \in \mathcal{T}$. Tags annotate subjects, thus we define the set of subjects, namely $s \in \mathcal{S}$, as the subset of entities that take on subject roles in $\mathcal{KG}$, $\mathcal{S} \subseteq \mathcal{E}$. Thus the tuple representation of the knowledge graph can be formulated as $\mathcal{KG}^* := \langle s, t\rangle$.

As mentioned in the introduction, a knowledge graph's class taxonomy is captured as a set of subsumption axioms. Formally, we define the taxonomy, $\mathcal{A}$, as the set of axioms, $\mathcal{A} := \{\langle c_i \rightarrow c_j\rangle \in \mathcal{E} \rightarrow \mathcal{E}\}$, where $\langle c_i \rightarrow c_j\rangle$ is an axiom indicating that class c_i subsumes class c_j. We note that classes are special types of entities in the knowledge graph that capture the type information of a subject entity, thus $c_i, c_j \in \mathcal{E}$. Classes are generally identifiable in a knowledge graph as the objects in a triple that feature the type identifying predicate for the knowledge graph. Different knowledge graphs will use a different predicate to capture this. DBpedia, for instance, uses `rdf:type`. The problem of taxonomy induction involves the generation of subsumption axioms that form a taxonomy that relates a knowledge graph's class entities. In other words, this work aims to derive $\mathcal{A}$ given $\mathcal{KG}^*$.

4 Model Description

In this section, we introduce our proposed model by positioning it in the context of hLDA, the model from which it draws inspiration. Specifically, we first outline the foundation of hLDA before describing our departures from it. We then provide the optimization scheme and algorithm used to train our model.

4.1 Hierarchical Latent Dirichlet Allocation

Hierarchical Latent Dirichlet Allocation is a topic model used in the field of natural language processing to hierarchically organize documents and their corresponding words or tags. The backbone of hLDA is composed of two stochastic processes that are used as non-parametric priors for generating the hierarchy and allocating tags along it: the nested Chinese restaurant process (nCRP) [7,4] and the stick-breaking process [30]. To understand the former, it is first necessary to understand the process which it extends, the Chinese restaurant process (CRP) [1]. The analogy used to describe the CRP is that of sitting patrons at a Chinese restaurant. Specifically, consider a restaurant with an infinite number of tables, each capable of potentially seating an infinite number of patrons. Patrons are seated sequentially such that the first patron is seated at the first table and all subsequent patrons are either seated at an occupied table or the first unoccupied table. The probability that a patron sits at a given table is proportional to the number of patrons that have already been seated at that table. Specifically, the probability of the i^{th} patron, p_i, being seated at table m given M occupied tables is:

$$\mathbb{P}(p_i = m | p_0, ..., p_{i-1}) = \begin{cases} \dfrac{\#_i^m}{i-1+\gamma} & m \leq M \\ \dfrac{\gamma}{i-1+\gamma} & m = M+1 \\ 0 & m > M+1 \end{cases} \tag{1}$$

Where $\#_i^m$ indicates the number of patrons seated at table m when patron p_i arrives and $\gamma > 0$ is a hyperparameter of the process that controls the probability that the patron will be seated at an unoccupied table. The nCRP extends the CRP to the hierarchical setting through nesting. Continuing with the analogy, consider the same restaurant described earlier but instead of being served food, patrons are served a reference to another restaurant they must go to. At this new restaurant, they are once again seated in the fashion described in Equation 1 and given a reference to another restaurant. This process continues for an infinite number of times. It is important to note that all patrons start at the same restaurant and that all restaurants have only one referring restaurant. Under these conditions, the paths taken by the patrons generate a tree of infinite depth and a potentially infinite number of branches. Each restaurant visited corresponds to a node in the tree such that the l^{th} restaurant visited is the tree node at level l. Formally, the probability that upon arriving at node n_k, patron

p_i takes path through n_c having already visited $l-1$ nodes is:

$$\mathbb{P}(p_i^l = n_c \mid \mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_{i-1}, \mathbf{p}_i[1\colon l-1], \gamma) = \begin{cases} \dfrac{\#_i^{n_c}}{\#_i^{n_k} + \gamma} & n_c \in \mathcal{N}_{n_k} \\ \dfrac{\gamma}{\#_i^{n_k} + \gamma} & n_c \notin \mathcal{N}_{n_k} \end{cases} \tag{2}$$

Where p_i^l indicates the path of patron p_i at the l^{th} level of the tree, $\mathcal{N}_{n_k}$ is the set of all previously taken paths from n_k, $\mathbf{p}_i$ is the path taken by patron p_i with the brackets $[a : b]$ denoting the partial path from level a to level b, and $\#_i^{n_k}$ and $\#_i^{n_c}$ are the number of patrons that have passed through communities n_k and n_c at the time patron p_i arrived, respectively. While the nCRP is a non-parametric prior over the tree structure of the hierarchy, the stick-breaking process is a non-parametric prior over the infinite number of levels in the hierarchy. As with the previous two processes, the stick-breaking process is explained by an analogy from which it derives its name. The analogy starts with taking a stick of a length of one and breaking it at a point sampled from the Beta distribution. Then, one piece of the stick is discarded and the other is broken again as per the Beta distribution. This process repeats infinitely, leaving a progressively smaller remainder of the stick. Formally, the length of the discarded stick at the l^{th} break, a^l, is defined as follows:

$$a^l = v^l \prod_{k=1}^{l-1} (1 - v^k) \tag{3}$$

Where v^l is a draw from the Beta distribution at the l^{th} iteration, namely $v^l \sim \text{Beta}(\mu\sigma, (1-\mu)\sigma)$. Here $1 > \mu > 0$ and $\sigma > 0$ are hyperparameters of the process and control the mean and variance, respectively. As can be seen, the resulting a^ls constitute a probability mass function that may be defined as follows:

$$\begin{aligned} \text{Stick}(\mu, \sigma) &= \sum_{l=1}^{\infty} a^l \\ &= \sum_{l=1}^{\infty} v^l \prod_{k=1}^{l-1} (1 - v^k) \end{aligned} \tag{4}$$

This distribution is leveraged by hLDA to allocate tags on the hierarchy. Specifically, hLDA first generates a tree using nCRP by sampling a path for each document. We note that in practice the tree must be bounded and thus the path selection stops after a predetermined depth of L. Each node in this tree is associated with a topic distribution, $\mathbf{b}$, drawn from the Dirichlet distribution. For each tag t_j in the document d_i, a level distribution is sampled from a stick-breaking distribution truncated to a depth of L, denoted as $\mathbf{l}$. The level distribution is sampled by parameterizing the Multinomial distribution to give the level indicator z_j. Tags are generated for each document by sampling the topic corresponding to the previously drawn paths and level indicators. The process is summarized as follows:

– for each node in the tree n_k
  - $\mathbf{b}_k \sim \text{Dirichlet}(\eta)$
– for each document d_i
  - $\mathbf{p}_i \sim \text{nCRP}(\gamma)$
  - $\mathbf{l}_i \sim \text{Stick}(\mu, \sigma)$
  - for each tag t_j in document d_i
    * $z_j \sim \text{Multinomial}(\mathbf{l}_i)$
    * $t_j \sim \mathbf{b}_{\mathbf{p}_i[z_j]}$

This generative model serves as the foundation of our proposed model. With this foundation, we can now outline our discretization changes.

4.2 Proposed Model

In introducing our proposed model, we leverage the notation used in the previous section, adapting it to match the tuple structure of the knowledge graph. Specifically, we make a distinction between subject entity paths, $\mathbf{p}_i^S \in \mathbf{P}^S$, and tag paths, $\mathbf{p}_j^T \in \mathbf{P}^T$. Each subject entity s_i has a corresponding path through the tree $\mathbf{p}_i^S$. As in hLDA, the length of path is equal to the tree depth, L. Similarly, each tag t_j has a corresponding path $\mathbf{p}_j^T$ as well as a level indicator $z_j \in \mathbf{Z}$ which is a one-hot vector of size L indicating the level on $\mathbf{p}_j^T$ that t_j belongs to. Thus, the node that t_j belongs to is the ${z_j}^{\text{th}}$ node on path $\mathbf{p}_j^T$, written as $\mathbf{p}_j^T[z_j]$. This formulation allows the state of the tree to be described by $\{\mathbf{P}^S, \mathbf{P}^T, \mathbf{Z}\}$. The primary difference between our model and hLDA is the discrete nature of tag assignment. Each tag belongs to only one node in the tree, thus each node is a set of tags as opposed to a distribution as is the case in hLDA. As such, subjects passing through node n_k are associated with all the tags that belong to the node, denoted as $\mathcal{T}_{n_k}$. We cannot, therefore, set up the generative model as a problem to be solved by Bayesian inference. Instead, we define an objective function and learn the model through discrete optimization.

Objective Function The objective function, J, defines the goal in the optimization process and acts as a prior for what the induced class taxonomy will look like. For instance, the objective function may favour narrow trees with larger nodes. This adds a degree of flexibility as different objective functions may be used on different datasets if there is a prior expectation about the structure of the taxonomy. In our evaluation procedures, we use the following objective function:

$$J = \sum_{s_i \in \mathcal{S}} \sum_{n_k \in \mathbf{p_i^S}} \sum_{t_j \in \mathcal{T}_{n_k}} \begin{cases} \dfrac{1}{|\mathcal{T}_{n_k}|} & \text{if } t_j \in \mathcal{T}_{s_i} \\ 0 & \text{otherwise} \end{cases}$$
$$- \sum_{s_i \in \mathcal{T}} \sum_{n_k \in \mathbf{p}_i^S} \begin{cases} |\mathcal{T}_{n_k}| + \alpha & \text{if } \mathcal{T}_{n_k} \cap \mathcal{T}_{s_i} = \emptyset \\ 0 & \text{otherwise} \end{cases}$$
$$- \sum_{s_i \in \mathbf{T}} \sum_{n_l \in \mathbf{p_i^S}} \begin{cases} |\mathcal{T}_{n_k}|(L - n_k^{level}) + \alpha & \text{if } |\mathcal{T}_{n_k}| > 1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Where $\mathcal{T}_{s_i}$ is the set of tags that are associated with subject s_i. To increase readability, we group the objective function into terms that capture a structural aspect the model is selecting for. Specifically:

- The first term adds all of the subject tags that appear in its path by an amount inversely proportional to the size of the node the tag is a part of.
- The second term is a penalty for a subject passing through a node that does not contain any tags that describe the subject. The penalty is proportional to the size of the node plus a penalty constant $\alpha > 0$.
- The third term penalizes nodes that contain more than one tag such that the penalty is larger for nodes higher in the tree. n_k^{level} is node n_k's level in the tree.

Broadly speaking, this objective function favours trees that have few tags at each node, with greater penalties for multi-tagged nodes that are higher in the hierarchy.

4.3 Model Optimization

Learning the best model is a problem of local search, wherein the space of possible states is searched by randomly initializing the tree variables, $\{\mathbf{P}^S, \mathbf{P}^T, \mathbf{Z}\}$, and generating candidate solutions through small adjustments. These candidates are evaluated using the objective function and chosen as the new solution according to a heuristic. There have been many methods proposed for traversing the solution space which may be applied to our model. In this paper, we focus on optimization by simulated annealing.

Simulated Annealing Simulated annealing [12,5], is a probabilistic approach to discrete optimization based on a process from metallurgy used to change the properties of a material by heating and then slowly cooling it. Applied to our problem, this translates to searching the state space in a way such that lower scoring states are less likely to be accepted as time progresses and the system cools. More concretely, the space of potential solutions is traversed by generating neighbouring states and moving to them according to the following rule: if the neighbour state is an improvement as per the objective function then move to the state, otherwise move to it with the following probability:

$$\mathbb{P}(x = x') = \exp(-\frac{J_x - J_{x'}}{\theta_{iter}})$$

Where x and x' are the current and neighbour states, and J_x and $J_{x'}$ are the objective functions of the current and neighbour states, respectively. We notice that the lower scoring the neighbouring state is relative to the current state, the less likely it is to be taken. The temperature hyperparameter, θ_{iter}, controls the probability of moving to lower neighbour states. Over time, the temperature decreases, thereby decreasing the probability of moving to lower states until the process is reduced to a basic hill-climbing algorithm. The annealing schedule controls the rate at which the temperature decreases. In our algorithm, we use the following annealing schedule:

$$\theta_{iter} = \theta_0 * \frac{\theta_{final}}{\theta_0}^{\frac{iter}{iter_{max}}}$$

Where θ_0 and θ_{final} are hyperparameters of the annealing schedule representing the initial and final temperatures, respectively, such that $0 < \theta_{final} < \theta_0$. $iter$ and $iter_{max}$ represent the current and maximum number of iterations, respectively. Algorithm 1 outlines the training procedure for our model. In line 5, neighbour generation is performed by updating subjects and tags alternately such that subjects and tags receive the same number of updates in the training process. We note, however, that for datasets with skewed proportions of subjects to tags, it may be advantageous to perform multiple subject path updates before updating a tag, or vice-versa. In line 13, rand(0,1) is a random number greater than or equal to zero and less than one.

Updating Subjects To update the path for s_i, the current path $\mathbf{p}_i^S$ is first removed from the tree. Any nodes in $\mathbf{p}_i^S$ that have no constituent subjects, tags, or descendants with tags are removed from the tree. This ensures that unused tree branches are pruned before sampling a new path. To generate a new $\mathbf{p}_i^S$, a terminal node, $\hat{p}_i^S$, is sampled from the Multinomial distribution:

$$\hat{p}_i^S \sim \text{Multinomial}(\pi_k^S)$$

Where π_k^S is the probability of selecting node n_k calcualted as:

$$P(\hat{p}_i^S = n_k) \propto \begin{cases} \frac{n_k^{level}}{L} & \text{if } n_k \text{ is an internal node} \\ \gamma & \text{if } n_k \text{ is a leaf node} \end{cases}$$

Where $\gamma > 0$ is a hyperparameter controlling how often new paths are chosen. Once $\hat{p}_i^S$ is sampled, the full path $\mathbf{p}_i^S$ can be derived by moving up through

Algorithm 1 Training Procedure Using Simulated Annealing

```
Input: $\mathcal{S}$, $\mathcal{T}$, $\theta_0$, $\theta_{final}$, $\alpha$, $\eta$, $iter_{max}$
Output: $\mathbf{P}^S$, $\mathbf{P}^T$, $\mathbf{Z}$
 1: Randomly initialize tree state $x = \{\mathbf{P}^S, \mathbf{P}^T, \mathbf{Z}\}$
 2: Calculate objective function for current state $J_x$
 3: for $iter$ in 1, 2, ..., $iter_{max}$ do
 4:     Update $\theta_{iter}$ according to annealing schedule
 5:     if $iter \% 2 = 0$ then
 6:         Randomly choose $s_i \in \mathcal{S}$ to update
 7:         Obtain $x'$ by updating $\mathbf{p}_i^S$
 8:     else
 9:         Randomly choose $t_j \in \mathcal{T}$ to update
10:         Obtain $x'$ by updating $\mathbf{p}_j^T$ and $z_j$
11:     end if
12:     Calculate objective function for neighbour state $J_{x'}$
13:     if rand(0,1) $< \exp(-\frac{J_x - J_{x'}}{\theta_{iter}})$ then
14:         Update $x = x'$
15:         Update $J_x = J_{x'}$
16:     end if
17: end for
18: Return tree at final state $x = \{\mathbf{P}^S, \mathbf{P}^T, \mathbf{Z}\}$
```

the parent nodes until the root. If $\hat{p}_i^S$ is an internal node, an empty branch is appended to $\hat{p}_i^S$ such that the path has L nodes. This process is similar in spirit to path sampling in hLDA, with three modifications: paths are not sampled at each level; existing paths are sampled uniformly; and the probability of creating a new path is dependent on the level of the node in the hierarchy.

Updating Tags To update the location of tag t_i in the hierarchy, both $\mathbf{p}_j^T$ and z_j need to be sampled. Similarly as to when updating subjects, t_j is first removed from the hierarchy and unused branches are pruned. To update the tag path, $\mathbf{p}_j^T$, we uniformly sample from the set of subject paths of subjects described by t_j. Formally:

$$\mathbf{p}_j^T \sim \text{Uniform}(\{\mathbf{p}_i^S \in \mathbf{P}_{-j}^S : t_j \in \mathcal{T}_{s_i}\})$$

Where the subscript $_{-j}$ indicates the preceding variable without the inclusion of j. Having sampled $\mathbf{p}_j^T$, t_j has to be assigned to a node along its new path. We sample tag level indicator z_j from the Multinomial distribution.

$$z_j \sim \text{Multinomial}(\pi_k^T)$$

Where π_l^T is the probability distribution that $z_j = l$, calculated as:

$$P(z_j = l) \propto \frac{\eta + |\{s_i \in \mathcal{S}_{-j} : \mathbf{p}_j^T[l] = \mathbf{p}_i^S[l] \wedge t_j \in \mathcal{T}_{s_i}\}|}{\eta + |\{s_i \in \mathcal{S}_{-j} : \mathbf{p}_j^T[l] = \mathbf{p}_i^S[l]\}|}$$

The numerator counts the number of subjects that are described by t_j and pass through the node at $\mathbf{p}_j^T[l]$. The denominator counts the number of subjects that pass through $\mathbf{p}_j^T[l]$. $\eta > 0$ is a small number that controls the probability of choosing nodes with no subjects described by t_j. Once the level has been sampled, t_j is assigned to the node at $\mathbf{p}_j^T[z_j]$.

5 Evaluation

Evaluation of taxonomy induction models is challenging due to the subjective nature of evaluating the correctness of the results. Authors such as Gu et al. [8] and Wang et al. [36] employed domain experts to manually evaluate the correctness of the induced axioms whereas Wang et al. [35] used a ranking approach. Others, such as Liu et al. [15] and Almoqhim et al. [2], leverage a gold standard taxonomy against which performance metrics are calculated. Although this method is less time consuming, expensive, and prone to bias than manual evaluation when a gold standard can be established, it is prone to inaccuracies and tends to misclassify valid subsumptions [23]. Nevertheless, this is the evaluation method we employ in our work as gold standards can be derived for our three datasets. Specifically, we use the F_1 score [6] as our measure of correctness. Hyperparameter selection is performed via a grid search of possible hyperparameter combinations and our best performing model is compared with models from the literature. Our source code along with the datasets used in our work is provided on GitHub[4] [17].

5.1 Datasets

We evaluated our model on three real-world datasets: Carnivora, DBpedia, and WordNet. What follows is a brief description of the datasets.

Carnivora The Carnivora dataset consists of the scientific classification of 253 animals that belong to the Carnivora order as described by Hunter [11]. The results were obtained by querying the Catalogue of Life: 2019 Annual Checklist [27] for the classification of each animal by its common name. We notice that since all of the animals are of the same order, it follows that they are also of the same kingdom, phylum, and class. We remove this redundant information and only include the lower three levels: order, family, genus. This results in a three-level taxonomy of 120 nodes distributed as follows: one node in the first level, twelve nodes in the second level, and 107 nodes in the third level. Hunter's classification provides the gold standard for our dataset.

DBpedia The DBpedia dataset was generated by querying for entities that belong on the top three levels of the gold standard DBpedia ontology[5]. Specifically,

[4] `https://github.com/mpietrasik/nppb`

[5] `http://mappings.dbpedia.org/server/ontology/classes/`

Table 1. Comparison of results obtained by our model and benchmarks on our datasets. (F_1 score mean ± standard deviation.)

Model	Carnivora	DBpedia	WordNet
Heymann and Garcia-Molina	0.9765 ± 0.0092	0.8673 ± 0.0230	0.5447 ± 0.0149
Schmitz	0.9831 ± 0.0000	0.8502 ± 0.0000	$\mathbf{0.6988 \pm 0.0000}$
Wang et al. (2012)	0.8571 ± 0.0279	0.6481 ± 0.0854	0.4462 ± 0.0135
Wang et al. (2018)	0.6908 ± 0.1851	0.5318 ± 0.1099	0.4286 ± 0.0949
Pietrasik and Reformat	0.9731 ± 0.0092	0.8788 ± 0.0086	0.6171 ± 0.0135
Our Model	$\mathbf{0.9866 \pm 0.0141}$	$\mathbf{0.8981 \pm 0.0370}$	0.5508 ± 0.0128

for each tag in the first three levels, all the entities that are described by the tag through the predicate `rdf:type` were queried. If this set contained fifteen or less entities, the tag was discarded. Otherwise, fifteen entities were selected randomly and queried for all the tags in the first three levels that describe them. At the time of querying, this process eliminated 87 out of 192 tags, meaning that at the three topmost levels 45% of DBpedia ontology terms have fifteen or fewer entities they describe. The majority of these have zero. The final tag hierarchy has 105 nodes with one root node, 27 nodes in the second level, and 77 nodes in the third level.

WordNet The WordNet dataset is a subset of DBpedia queried to contain subjects of types that exist in the WordNet lexicon [18]. The gold standard was obtained as the DBpedia relations between WordNet classes through the `rdfs:subClassOf` predicate. Specifically, `yago:PhysicalEntity100001930` was set as the root class and the taxonomy is built by recursively querying for subclasses using the `rdfs:subClassOf` predicate. The derived WordNet class taxonomy contained 100 tags split over four levels with one root node, five nodes in the second level, 23 in the third level, and 71 in the fourth level. We note that it was necessary to collapse the taxonomy by removing tags that were missing in the dataset and adopting orphaned tags with the nearest ancestor.

5.2 Results

Our model was applied to each dataset five times to account for stochasticity in the training process. Hyperparameters were chosen by a grid search exploration of the hyperparameter space. We compare our model with benchmarks obtained from implementations of five methods found in the literature: Heymann and Garcia-Molina, Schmitz, Wang et al . (2012), Wang et al. (2018), and Pietrasik and Reformat. The methods were chosen based on the quality and thoroughness of the details necessary for implementation as well as the lack of reliance on domain knowledge. We note, therefore, that our implementation of Wang et al. (2018) only considers the unsupervised model, without domain knowledge optimization. As with our model, we performed hyperparameter exploration on the benchmark methods and applied each dataset five times. The F_1 means and standard deviations of all the methods tested are reported in Table 1.

For the Carnivora dataset, we used half a million simulated annealing steps to train our model. We found that the hyperparameter values of $\theta_0 = 2.5$, $\theta_{final} = 0.01$, $\alpha = 10$, $\gamma = 0.01$, and $\eta = 0.01$ produce the best results. The threshold values for Heymann and Garcia-Molina and Schmitz were 0.3 and 0.9, respectively. Wang et al. (2012) had both k and minimum cluster size set to 22. Pietrasik and Reformat was tested with an α value of 0.7. The Carnivora dataset is considered easy for two reasons: all entities have exactly three classes, each belonging to a different level in the hierarchy; and all entities are classified correctly. We see this reflected in the high F_1 scores for all models. We note that the errors in our model are due to the inability to infer the correct order of parent-child relations when parent and child occur at the same frequencies. The poor performance of Wang et al. (2018) is explained by the model's lack of ability to differentiate between parent nodes that have the same co-occurrence rates. The parent is therefore chosen based on which one came first in the vocabulary which is random in our implementation.

To account for dataset complexity, ten million simulated annealing steps were performed to train our model on the DBpedia dataset finding that the same hyperparameter combination as that of Carnivora produced the best results. We use the same threshold value for Heymann and Garcia-Molina as before and 0.65 for Schmitz. Wang et al. (2012) used 17 for k and minimum cluster size and Pietrasik and Reformat had an α value of 0.5. We notice that this dataset is more difficult as reflected by the lower average F_1 scores. In contrast to Carnivora, the DBpedia dataset is inconsistent in the amount of tags for each document and there are errors in the assigning of classes to subjects. In light of this, the higher difference between our model and the benchmark models suggests that our model is more robust to less structured datasets.

Our model performed comparatively worse on the WordNet dataset. Similar to DBpedia, the dataset presents challenges resulting from its incongruities between class labels and the gold standard taxonomy. Moreover, the dataset generation process included the collapsing of taxonomy nodes, further adding to its complexity. As with DBpedia, we performed ten million simulated annealing steps to train our model. We found, however, that a different hyperparameter combination was optimal on the dataset. Specifically, we used $\theta_0 = 2.5$, $\theta_{final} = 0.01$, $\alpha = 1$, $\gamma = 0.1$, and $\eta = 0.01$ in our results. We used thresholds value of 0.5 and 0.9 for Heymann and Garcia-Molina and Schmitz, respectively The same hyperparameters were used for Wang (2012) as in DBpedia and an α value of 0.9 for Pietrasik and Reformat. Despite trailing two benchmark methods on the WordNet dataset, our model, on the whole, outperforms existing methods in our experiments.

5.3 Model Limitations

The primary limitation of our model is that of efficient model optimization. Currently, our model cannot train fast enough to handle very large datasets. The principal cause of this is the size of the search space when combined with the optimization technique used to traverse it. We can approximate the upper bound of the search space using perfect m-ary trees which are defined as rooted trees where each node has exactly m children and all leaf nodes are at a depth of L. We can arrive at this bound by assigning subject and tag paths to the leaf nodes of the maximally branching m-ary tree for our model. The maximum branching factor is $|\mathcal{S}| + |\mathcal{A}|$, meaning there are $(|\mathcal{S}| + |\mathcal{A}|)^L$ nodes at level L. We approximate the upper bound of the size of the state space as follows:

$$L|\mathbf{Z}|(((|\mathcal{S}| + |\mathcal{A}|)^L))^{|\mathcal{S}|+|\mathcal{A}|} \tag{6}$$

This value is derived by counting all possible path allocations on the leaf nodes of the maximum branching m-ary tree at all possible level allocations. As it is an upper bound, it overestimates the true size of the state space. We see that the state space scales too quickly with respect to the input for simulated annealing. Another limitation of our model is that of inferring the correct order of parent-child relations when parent and child classes co-occur at the same frequencies. To our model, these two classes are identical and their relative assignment in the hierarchy will be determined randomly. We note that this is a problem inherent to all frequency-based models.

6 Conclusion and Future Work

In this paper, we proposed a non-parametric path based model for inducing a taxonomy of classes from a knowledge graph. Our model draws inspiration from hLDA and transfers aspects from it to the discrete space. In order to apply these ideas, we first transformed the knowledge graph from its original triple structure to a tuple structure. Our model was evaluated by inducing a class taxonomy for three real-world datasets and comparing the resulting subsumption axioms to their respective gold standards. Our model is competitive with or outperforms benchmarks obtained from similar models in the literature. Future work can include a manual, qualitative evaluation of the induced taxonomies along the attributes identified by Nickerson et al. [21] that define a useful taxonomy: conciseness, robustness, comprehensiveness, extendability, and explainability. In addition to this, we also discussed the limitations currently present in our model, namely its inability to scale to large datasets. With this in mind we propose two directions for future work on the problem of scalability: decrease the size of the search space by modifying subject and tag update procedures; and explore different optimization methods for traversing the search space. Another avenue for future research is exploring our model's capacity for hierarchical clustering of subject entities. We can consider the set of subjects that travel through the same node as belonging to the same cluster at the level of that node. Thus, at the lowest level in the hierarchy, subjects are clustered together only if they have identical paths. As the clustering moves up the taxonomy, the clusters are merged as the branches in the tree join together. The classes associated with nodes may be viewed as descriptors of the clusters.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Aldous, D.J.: Exchangeability and related topics. In: École d'Été de Probabilités de Saint-Flour XIII—1983, pp. 1–198. Springer (1985)
2. Almoqhim, F., Millard, D.E., Shadbolt, N.: Improving on popularity as a proxy for generality when building tag hierarchies from folksonomies. In: International Conference on Social Informatics. pp. 95–111. Springer (2014)
3. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: Optics: ordering points to identify the clustering structure. In: ACM Sigmod record. vol. 28, pp. 49–60. ACM (1999)
4. Blei, D.M., Griffiths, T.L., Jordan, M.I.: The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. Journal of the ACM (JACM) **57**(2), 7 (2010)
5. Černỳ, V.: Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. Journal of optimization theory and applications **45**(1), 41–51 (1985)
6. Chinchor, N.: Muc-4 evaluation metrics. In: Proceedings of the 4th conference on Message understanding. pp. 22–29. Association for Computational Linguistics (1992)
7. Griffiths, T., Jordan, M., Tenenbaum, J., Blei, D.: Hierarchical topic models and the nested chinese restaurant process. Advances in neural information processing systems **16** (2003)
8. Gu, C., Yin, G., Wang, T., Yang, C., Wang, H.: A supervised approach for tag hierarchy construction in open source communities. In: Proceedings of the 7th Asia-Pacific Symposium on Internetware. pp. 148–152. ACM (2015)
9. Heymann, P., Garcia-Molina, H.: Collaborative creation of communal hierarchical taxonomies in social tagging systems. Tech. rep. (2006)
10. Horta, V.A., Sobczyk, R., Stol, M.C., Mileo, A.: Semantic interpretability of convolutional neural networks by taxonomy extraction. In: NeSy. pp. 118–127 (2023)
11. Hunter, L.: Carnivores of the world. Princeton University Press (2011)
12. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. science **220**(4598), 671–680 (1983)
13. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al.: Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. Semantic web **6**(2), 167–195 (2015)
14. Li, X., Wang, H., Yin, G., Wang, T., Yang, C., Yu, Y., Tang, D.: Inducing taxonomy from tags: An agglomerative hierarchical clustering framework. In: International Conference on Advanced Data Mining and Applications. pp. 64–77. Springer (2012)
15. Liu, K., Fang, B., Zhang, W.: Ontology emergence from folksonomies. In: Proceedings of the 19th ACM international conference on Information and knowledge management. pp. 1109–1118. ACM (2010)
16. Martel, F., Zouaq, A.: Taxonomy extraction using knowledge graph embeddings and hierarchical clustering. In: Proceedings of the 36th Annual ACM Symposium on Applied Computing. pp. 836–844 (2021)
17. McCallum, A.K.: Mallet: A machine learning for languagetoolkit. http://mallet. cs. umass. edu (2002)
18. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)

19. Nickel, M., Tresp, V., Kriegel, H.P.: Factorizing yago: scalable machine learning for linked data. In: Proceedings of the 21st international conference on World Wide Web. pp. 271–280. ACM (2012)
20. Nickel, M., Tresp, V., Kriegel, H.P., et al.: A three-way model for collective learning on multi-relational data. In: Icml. vol. 11, pp. 3104482–3104584 (2011)
21. Nickerson, R.C., Varshney, U., Muntermann, J.: A method for taxonomy development and its application in information systems. European Journal of Information Systems **22**(3), 336–359 (2013)
22. Pellissier Tanon, T., Weikum, G., Suchanek, F.: Yago 4: A reason-able knowledge base. In: European Semantic Web Conference. pp. 583–596. Springer (2020)
23. Pietrasik, M., Reformat, M.: A simple method for inducing class taxonomies in knowledge graphs. In: European Semantic Web Conference. pp. 53–68. Springer (2020)
24. Pietrasik, M., Reformat, M.: Path based hierarchical clustering on knowledge graphs. arXiv preprint arXiv:2109.13178 (2021)
25. Pietrasik, M., Reformat, M., Wilbik, A.: Hierarchical blockmodelling for knowledge graphs. arXiv preprint arXiv:2408.15649 (2024)
26. Ristoski, P., Faralli, S., Ponzetto, S.P., Paulheim, H.: Large-scale taxonomy induction using entity and word embeddings. In: Proceedings of the International Conference on Web Intelligence. pp. 81–87. ACM (2017)
27. Roskov Y., Ower G., O.T.N.D.B.N.K.P.B.T.D.R.D.W.N.E.v.Z.J.P.L.e.: Species 2000 itis catalogue of life, 2019 annual checklist. (2019)
28. Sanderson, M., Croft, B.: Deriving concept hierarchies from text. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. pp. 206–213. ACM (1999)
29. Schmitz, P.: Inducing ontology from flickr tags. In: Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland. vol. 50, p. 39 (2006)
30. Sethuraman, J.: A constructive definition of dirichlet priors. Statistica sinica pp. 639–650 (1994)
31. Soetekouw, T., Grefen, P., Vanderfeesten, I., Turetken, O.: Developing taxonomies for business process engineering. In: International Conference on Enterprise Design, Operations, and Computing. pp. 169–186. Springer (2023)
32. Völker, J., Niepert, M.: Statistical schema induction. In: Extended Semantic Web Conference. pp. 124–138. Springer (2011)
33. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)
34. Wang, S., Wang, T., Mao, X., Yin, G., Yu, Y.: A hybrid approach for tag hierarchy construction. In: International Conference on Software Reuse. pp. 59–75. Springer (2018)
35. Wang, S., Lo, D., Jiang, L.: Inferring semantically related software terms and their taxonomy by leveraging collaborative tagging. In: 2012 28th IEEE International Conference on Software Maintenance (ICSM). pp. 604–607. IEEE (2012)
36. Wang, W., Barnaghi, P.M., Bargiela, A.: Probabilistic topic models for learning terminological ontologies. IEEE Transactions on Knowledge and Data Engineering **22**(7), 1028–1040 (2009)
37. Wetzel, M., Mathioulakis, S., Hummel, J., Vasilevich, A.: Collaborative-ai knowledge graph generation: Taxonomization of iate, the eu terminology. In: KGCW@ ESWC (2021)
38. Zhang, Y., Pietrasik, M., Xu, W., Reformat, M.: Hierarchical topic modelling for knowledge graphs. In: European Semantic Web Conference. pp. 270–286. Springer (2022)