Unsupervised Tree Extraction in Embedding Spaces for Taxonomy Induction

François Torregrossa, Robin Allesiardo, Vincent Claveau, Guillaume Gravier

Unsupervised Tree Extraction in Embedding Spaces for Taxonomy Induction

François Torregrossa
ftorregrossa@solocal.com
francois.torregrossa@irisa.fr
Solocal and IRISA
Rennes, France

Robin Allesiardo
rallesiardo@solocal.com
Solocal
Rennes, France

Vincent Claveau
vincent.claveau@irisa.fr
IRISA, CNRS
Rennes, France

Guillaume Gravier
guig@irisa.fr
IRISA, CNRS
Rennes, France

ABSTRACT

Exposing latent structure (graph, tree...) of data is a major challenge to deal with the web of data. Today's embedding techniques incorporate any data source (noisy graphs, item similarities, plain text) into continuous vector spaces that are typically used as input to classifier. In this work, we are dealing with the opposite task: finding structures (taxonomies) from embedded data. We provide an original unsupervised methodology for taxonomy induction by directly searching for graph structures preserving pairwise distances between items. Contrary to the state-of-the-art (SOTA), our approach does not require to train classifiers; it is also more versatile as it can be applied to any embedding (eg. word embedding, similarity embedding like space-time local embedding...). On standard benchmarks and metrics, our approach yields SOTA performance. As another contribution, we propose better evaluation metrics for taxonomy induction, leveraging graph kernel similarities and edit distance, showing that the structures of our predicted taxonomies are significantly closer to the ground-truth than SOTA solutions.

KEYWORDS

Unstructured/Structured Data, Hyperbolic Embedding, Tree extraction, Taxonomy Induction

1 INTRODUCTION

Nowadays, structured knowledge sources, such as knowledge graphs or taxonomies, can be integrated in numerous AI systems. Question answering, information extraction/retrieval are examples of domains where taxonomies are found advantageous, as they enhance the generalization ability of automatic systems [6, 18, 31]. Yet, taxonomies are often designed by human experts which implies time and financial costs. Moreover, those handcrafted taxonomies are configured with regard to a topic which may not match every usage. Others even claim that structured knowledge inferred from Gargantuan online sources such as WikiData are not suited for specific domain [31].

An early idea to address these issues is to automatically induce taxonomies from text sources [13]. Indeed, automatic knowledge discovery ultimately leads to a tailored taxonomy closing the gap between general taxonomies and specific applications. Provided taxonomical terms and a sufficiently large text corpus covering those, the process consists in collecting morphological, syntactic, and semantic features to classify or extract *is-a* relations between these terms [2, 6, 18, 23]. This taxonomy is then employed in real-world applications (eg. e-commerce, biomedecine) [11, ch. 3, for examples], which help at judging its quantitative quality [31].

Generally, collected features are used to produce continuous or handcrafted term embeddings used in classifiers or with heuristics helping at predicting the taxonomy [2, 18, 23]. For instance, noisy *is-a* relations are projected into a hyperbolic space in order to remove noise and obtain cleaner features [2]. In our opinion, these additional supervised steps seem disposable, as some intermediate embedding proposed in the literature perform well at hypernym detection [16]. Provided a mapping from embedding to taxonomies, preserving pairwise distances between term vectors, the mapped taxonomy may also preserve the geometrically embedded structural information. Today, there are methods to convert trees or graphs to continuous hyperbolic embeddings with minimal information loss [22, 26]. Here, we propose solutions doing the exact opposite for taxonomy induction: converting continuous embedding spaces to discrete trees. In other words, we propose to address taxonomy induction with a different point of view: instead of using embeddings as filters or as input to classifiers [2], we intend to directly extract the latent structural information from them. We build a taxonomy that tries to preserve as much as possible the information contained in the distance between embedded terms.

The contributions of this paper can be summarized as follows:

(1) we provide a *new unsupervised methodology* for the taxonomy induction problem relying on the distortion between metric spaces. This methodology is flexible and is applicable to any kind of data sources (plain text, graph, similarities).
(2) we propose a *new feature* mixing morphological and syntactic features, used to build term embeddings.
(3) we *explore three different solutions* for our tree extraction problem on various distances—Euclidean and hyperbolic.
(4) we propose *a new evaluation protocol* to quantitatively evaluate how close the induced taxonomies are from the ground-truth. In this respect, we experimentally show that our approach outperforms existing methods.

The paper is organized as follows: Sec. (II) gives an overview of the state-of-the-art (SOTA) and basic notions for taxonomy induction. Sec. (III) describes our whole approach. Experimental results are gathered in Sec. (IV) before conclusion in Sec. (V).

2 RELATED WORK

A taxonomy $\mathcal{T} = (V, E)$ is an acyclic directed connected graph denoted by a set of nodes V and an edge set E. Nodes are labeled by terms which are word phrases representing a concept of the taxonomy. For instance, *«computer science»* may be chosen as a node label to depict the computer science field composed by sciences such as *«software engineering»*, *«robotics»* or *«artificial intelligence»*. In the following, we refer to nodes as terms.

The edge set E is viewed as a set of *is-a* relations, i.e. if a term $x \in V$ *is-a* term $y \in V$, then there is a directed edge from x to y in E. In the following, we denote this relation by $(x, \textit{is-a}, y)$ as in [16]. The set of *is-a* relations is defined by the taxonomy and illustrate that the characteristics or concepts of x are contained in those of y (eg. *«artificial intelligence»* is a field of *«computer science»*).

2.1 Overview

Taxonomy induction is a problem where an edge set of *is-a* relations must be found for a given set of terms. A text corpus may be used as a source to compute features for each term [2, 4]. Several shared tasks were proposed following this scheme, the most recent one being SemEval-2016 Task 13 [4]. Among the participating systems, TAXI [23] obtained the best results by exploiting lexico-syntactic patterns on a general (Wikipedia) and a domain-specific corpus. Other systems participated: USAAR [30] leverages hyponym endocentricity (see next sub-section); JUNLP [17] relies on BabelNet (https://babelnet.org/) and morphological rules.

In the early approaches, the set of relations is constructed leveraging morphological features of the term (seen as a character strings) and syntactic features (their co-occurrences in the plain text corpus following syntactic patterns, known as Hearst patterns) [13, 23, 30, 31]. Extracted relations are then further filtered or pruned using hard-coded rules (USSAR, JUNLP, SubSeq [12]) and/or a supervised system (TAXI). Distributional features (such as word embedding) may be used to help with the inference of the taxonomy, but they are not fully exploited [4, 31]. Indeed, their usage is limited to the refinement of taxonomy constructed by TAXI [2], or as input of supervised systems [18]. Recently, a proposition (CTP) was made to fine-tune contextualised embedding for taxonomy induction [5]. It explores the use of syntactic patterns in pre-trained contextualised language models. While it shows an improvement for induction of small taxonomies (10-50 terms), it did not improve induction of medium to large scale taxonomies (100-1000 terms), which is our main interest here.

Other systems avoid pruning as it appears to be an error-prone process. TaxoRL [18] is an end-to-end supervised system relying on reinforcement learning to construct a taxonomy from scratch. As it is supervised, it requires real-world taxonomy to be trained on: sub-taxonomies of WordNet [19] were used in that purpose. Graph2Taxo [28] is another end-to-end approach which predicts the adjacency matrix of the taxonomy, but also introduces direct cross-domain knowledge integration. In other words, it used a priori known taxonomy to create features to help with the prediction of edges for unknown taxonomies.

The major issue with SOTA techniques is the supervision. TAXI, TaxoRL, Graph2Taxo, CTP require external structured data to train their models or heavy heuristics to work. Instead, we propose an unsupervised taxonomy induction system. Our approach is inspired by HyperCones [16], an unsupervised hyperbolic term embedding based on the co-occurrence counts of Hearst patterns. They leverage hyperbolic geometry which is shown more effective than Euclidean geometry at representing and recovering the hierarchical structure of tree structures [10, 22]. We push these ideas further to produce complete taxonomies from term embeddings instead of the simpler task of hypernym detection proposed by [16]. Indeed, hypernym detection does not have to deal with the directed acyclic and connected nature of taxonomies. We also propose to construct and embed a graph that leverages both morphological and syntactic features, while they mainly focused on the syntactic ones.

In this work, we compare with TAXI [23], USAAR [30], JUNLP [17] and TAXI with refinement presented in [2], an improved version of TAXI. Unfortunately, we will not compare with SubSeq for reproducibility issues. As we do not use cross-domain input or supervision with external taxonomies, other systems (like Graph2Taxo, TaxoRL and CTP) are not considered. To the best of our knowledge, TAXI with refinements is the best taxonomy induction system that is not using cross-domain information for the SemEval-2016 Task 13. We provide, in the following, further details on related studies and important notions helping to understand how we completed this task.

2.2 Feature Engineering

Feature extraction from plain text or term strings is an important step in taxonomy induction since classifiers used in TAXI [23] or more complex models such as TaxoRL or Graph2Taxo [18, 28] rely on them to produce taxonomies. The quality of the resulting taxonomies highly depends on those features.

2.2.1 Morphological features. They simply compare term strings while being particularly accurate. This is why they are widely exploited [2, 18, 23, 30]. Indeed, the appearance of one term in another, known as the hyponym endocentricity [30], often indicates a *is-a* relation (eg. *engineering* and *electrical engineering*). Generally, a set of morphological features are generated by pointing out whether prefix or suffix matches or the length of the longest common substring [3]. Although, morphological features are very precise, they lack recall since they cannot detect *is-a* relations for pairs without common morphological components (eg. *algebra* and *mathematics*).

2.2.2 Syntactic features. They are produced with syntactic patterns, or Hearst patterns [13], which are applied on the corpus to harvest specific co-occurrences between terms of the taxonomy. A set of patterns is pre-defined, such as *"X is a Y"* or *"X such as Y"*, and one counts the number of times patterns are triggered for a term pair *X, Y*. The Hearst graph is a formal representation of these frequency counts. In other words, nodes of the graph are taxonomical terms and an edge linking a term X to another term Y is weighted by the amount of times X and Y matched patterns in the corpus. This approach is widely used [2, 16, 18, 23, 25] because

it is simple, explainable, scalable. *WebIsA* [27] is a free online available database gathering such frequency counts crawled on very large web corpora, used in this work and in [2, 23] to enrich term features.

2.2.3 Distributional features. They rely on vectors from word embeddings (eg. word2vec) or distributional approaches to compute inclusion or semantic proximity of terms. They are finally added either before classification or modeling [18] or after in order to detect co-hyponyms [2] and help to predict the taxonomy. Distributional features may also be used to detect *is-a* relations using the Distributional Inclusion Hypothesis claiming that contexts of a term are included in the contexts of its hypernyms [31, 32].

2.3 Hyperbolic Term Embeddings

Recently, [2, 16] proposed to embed the Hearst graph into a hyperbolic space, following the hyperbolic graph embedding method of [22]. By projecting the graph into a continuous hyperbolic space where the geometry is more suited for taxonomical inference, [2, 16] showed that it solved some of the problem caused by the noise and sparsity of the Hearst graph (patterns are sometimes over-sensitive which might creates cycles or wrong *is-a* relations, or too restrictive which hides true ones). Those relations are condensed and term embedding vectors are interpreted into new relations, using heuristics or specific tuning on the hyperbolic distance.

In [2], hyperbolic embeddings are used to refine the taxonomies predicted with traditional methods explained above. Thus, it adds an additional post-processing on the top of an already highly layered solution, increasing error propagation and reducing the ability to benefit from low layer improvements. In our work, we propose to process these embeddings directly to search for the taxonomical graph that preserves embedding term distances. Also, contrary to [16], we aim at building an entire taxonomy on a specific topic while they only focused on hypernymy detection.

2.4 Distortion

The distortion indicates how well a mapping from a metric space (e.g. an embedding) to another (e.g. a graph / tree) preserves pairwise distances (see [21, 26]). It is mathematically defined by:

$$d_f = \max_{\substack{(x,x')\in X^2 \\ x\neq x'}} \frac{d_Y(f(x),f(x'))}{d_X(x,x')} \cdot \max_{\substack{(y,y')\in f(X)^2 \\ y\neq y'}} \frac{d_X(f^{-1}(y),f^{-1}(y'))}{d_Y(y,y')} \tag{1}$$

where f is a mapping from one metric space (X, d_X) to another (Y, d_Y); in this work, from a graph metric to an embedding space.

[26] used it to prove that their method is able to embed a tree in a hyperbolic space while preserving the distances, thus the information contained in the tree. In this case, the distance in a tree d_T is the sum of the edge weights of the path between nodes. Taxonomy induction can be seen as the opposite problem where one is given an embedding of terms—constructed as described in the previous sub-section—and need to recover a tree from it seen as the taxonomy skeleton: this is exactly what we aim to accomplish. Put differently, we need a method for extracting a tree of terms from term embeddings having the lowest distortion.

An embedding can be considered as a fully connected graph where nodes are embedded elements and edges are weighted by a distance function. If the graph is equipped with the standard graph distance d_G—defined by the path with the smallest weight sum between two nodes—then the graph and d_G form a finite metric space. Then, our problem is known in the network literature as finding tree spanner with lowest expansion / distortion [1]. The lowest expansion tree-spanner is named the Minimum Max-stretch Spanning Tree (MMST).

When distances in the tree perfectly preserve distances in the embedding (such a tree does not always exist), then the MMST matches the Minimum Spanning Tree (MST) [1, 21]. In general, the distortion is not perfect and we must have a strategy to construct the MMST. This problem is known to be NP-hard [1]. Due to the complexity of the problem, we investigate a naive solution (identical to [20]) or approximate solutions produced by some algorithms presented in [7].

3 METHODOLOGY

3.1 Combination of morphological and syntactic features

3.1.1 Overview. Similarly to [16, 23, 25], we use pattern based features (which are turned into a mutual information space) and morphological features. We propose to also turn morphological features into a mutual information space, and to sum pattern-based and morphological mutual information scores. The combination of this morphological and syntactic information leads to a new feature f_{agg}, contribution (2). We detail each element hereafter. In the end, we are able to quantify the strength of *is-a* relations between terms represented as a weighted directed graph (weights being f_{agg}). We expect this graph to be less ambiguous than the Hearst Graph which is usually employed [2, 16].

3.1.2 Corpora. For comparison purposes, we use the same corpus and counts of *is-a* patterns provided by [2][1]. Following previous work [2, 18, 23], this corpus consists of texts from two different sources: *general* and *specific*. The general domain corpus is composed of texts coming from huge online resources such as Wikipedia. The specific corpus gathers texts being correlated with the topic of the taxonomy, collected by querying a Web search engine with random combinations of taxonomical terms.

3.1.3 Syntactic feature. The corpora are processed leveraging syntactic patterns to extract co-occurrences between terms with tools such as *PattaMaika*[2]. Also, frequencies from *WebIsA* [27] are collected to enrich our feature collection. Finally, three kinds of co-occurrences are obtained: general, specific and *WebIsA* ones. They are each independently turned into a Positive Pointwise Mutual Information (PPMI) space, following [25].

3.1.4 morphological feature. The morphological feature $\sigma(t_i, t_j)$ between two terms x and y is an hypernymy substring based score [23] formally defined by $\sigma(x,y) = \frac{\text{length}(x)}{\text{length}(y)}$ if x inside y and 0 otherwise. Before being combined, the hypernymy substring based score is transformed as a PPMI score as above.

[1] https://github.com/uhh-lt/Taxonomy_Refinement_Embeddings
[2] http://ltmaggie.informatik.uni-hamburg.de/jobimtext/components/pattamaika/

3.1.5 Aggregated feature. We introduce a new aggregating feature f_{agg} which is given by assembling for each term pair (x, y) patterns and morphological features as follows:

$$f_{\text{agg}}(x, y) = \text{ppmi}_{\sigma}(x, y) + \frac{1}{3} \sum_{d=\left\{\begin{smallmatrix}\text{General}\\ \text{Specific}\\ \text{WebIsA}\end{smallmatrix}\right\}} \text{ppmi}_{d}(x, y) \quad (2)$$

This feature weights the strength of the relation $(x, \textit{is-a}, y)$ in the three corpora while also taking into account their substring scores. The SOTA methods commonly employ a classifier to combine morphological and syntactic features, or exclusively use the syntactic ones. We advocate for f_{agg} which fuses early both features and, thus, integrates all information into the graph prior to the term embedding step.

To justify this new feature, we show its effectiveness at ranking term neighbors, w.r.t. its components, in Table 1. The data used is a part of the Semeval 2016 Task 13 [4] dataset (see Sec. 4.3). The f_{agg} improvement over any other feature is statistically significant (one-sided t-test with p<0.01). One can remark that taking individual component features of f_{agg} is less efficient at organizing neighborhood, which advocates for the combination of all features as we propose.

We create a matrix using this feature, which is smoothed by removing some of its singular values as done in [25], and values under the mean value. The underlying matrix graph is further pruned such that each term have at most 5 direct neighbors. This value was chosen arbitrarily and further investigation must be carried to observe its impact on the final performance. This graph is then embedded into a Euclidean or a hyperbolic space, following Sec. 3.2.

3.2 Term embeddings

For convenience, the custom graph introduced in the previous section is denoted $G = (V, M)$, V being the terms and M being the smoothed asymmetric matrix of the weights between each terms. We also note $M(x, y)$ for $(x, y) \in V^2$, the weights for the relation $(x, \textit{is-a}, y)$. We now thoroughly explain how it can be embedded into a Euclidean or hyperbolic space [10, 22].

3.2.1 Embedding parameters. Let $\mathcal{D}^N = \{(x, y) | M(x, y) > 0, (x, y) \in V^2\}$ be the set of noisy directed edges estimated by taking the term pairs corresponding to the positive values of the matrix M. We parameterize $|V|$ term embedding vectors with dimension n, denoted by $\Theta = (\theta_x)_{x \in V} \in \mathbb{R}^{n \times |V|}$. We perform optimization using three distances having specific characteristics that alter the optimization procedure:

1) The Euclidean distance $d_{\mathcal{E}}$. It consists of the well known L2-norm, written for two terms $(x, y) \in V^2$, $d_{\mathcal{E}}(x, y) = \|\theta_x - \theta_y\|_2$.

2) The Poincaré distance $d_{\mathcal{P}}$ [22]. It is defined for terms $(x, y) \in V^2$ on a unit sphere $\mathbb{D}^n = \{a | a \in \mathbb{R}^n, \|a\|_2 < 1\}$ as:

$$d_{\mathcal{P}}(x, y) = \text{arcosh}\left(1 + 2\frac{\|\theta_x - \theta_y\|_2^2}{(1 - \|\theta_x\|_2^2)(1 - \|\theta_y\|_2^2)}\right) \quad (3)$$

The parameters Θ must be initialized and optimized in $\mathbb{D}^n$. This distance is expected to encode more easily hierarchies compared with the Euclidean distance.

3) The squared Lorentzian pseudo-distance $d_{\mathcal{L}}$ [15]. It is defined on the hyperboloid $\mathcal{H}^{n,\beta} = \{a = (a_0, ..., a_n) \, | a \in \mathbb{R}^{n+1}, \|a\|_{\mathcal{L}}^2 = -\beta\}$, where $\beta \in \mathbb{R}$ is a hyperparameter. It is based on the squared Lorentzian inner product $\langle a, b \rangle_{\mathcal{L}} = -a_0 b_0 + \sum_{i=1}^{n} a_i b_i$, and the squared Lorentzian norm $\|a\|_{\mathcal{L}}^2 = \langle a, a \rangle_{\mathcal{L}}$. As we already have an Riemannian optimizer for $\mathbb{D}^n$, we propose to initialize the parameter Θ in $\mathbb{D}^n$ and then project them to $\mathcal{H}^{n,\beta}$ with the projector h^{-1} from $\mathbb{D}^n$ to $\mathcal{H}^{n,\beta}$ [15]:

$$h^{-1}(\theta_x) = \left(\sqrt{\left\|\frac{2\theta_x}{1 - \|\theta_x\|_2^2}\right\|_2^2 + \beta}, \frac{2\theta_{x,1}}{1 - \|\theta_x\|_2^2}, ..., \frac{2\theta_{x,1}}{1 - \|\theta_x\|_2^2}\right) \quad (4)$$

with $\theta_x = (\theta_{x,1}, ..., \theta_{x,n})$. The squared Lorentzian pseudo-distance between two terms $(x, y) \in V^2$ is: $d_{\mathcal{L}}(x, y) = -2\beta - 2\langle h^{-1}(\theta_x), h^{-1}(\theta_y) \rangle_{\mathcal{L}}$. The main advantage of $d_{\mathcal{L}}$ is its ability to strongly enforce generic elements to remain in the center of $\mathbb{D}^n$ and the specific ones at the border of $\mathbb{D}^n$. In comparison, the Poincaré distance is less able to do so. This attitude is desirable since it may help at recomposing the taxonomy.

3.2.2 Optimization. We build the term embeddings w.r.t. the chosen distance d such that they incorporate information contained in $\mathcal{D}^N$. The loss function is:

$$\mathcal{L}(\Theta) = \sum_{(x,y) \in \mathcal{D}^N} \log \frac{e^{-d(x,y)}}{\sum_{y' \sim \mathcal{N}(x)} e^{-d(x,y')}} \; ; \mathcal{N}(x) = \{y' | (x, y') \notin \mathcal{D}^N\} \quad (5)$$

where $\mathcal{N}(\S)$ is the set negative edges (terms are not linked to x in $\mathcal{D}^N$). $\mathcal{L}(\Theta)$ is minimized with the following setting: for each batch, we sample 10 negative edges and compute the Euclidean gradients of $\mathcal{L}$ which are then used to realize a Riemannian optimization as presented in [10] or [22]. For both optimizations, we relied on *hyperbolic_cones* and *gensim* projects.

Table 1: MAP of neighborhood produced by each feature on the direct edges of the EN environment taxonomy of Semeval 2016 Task 13 [4].

Features	MAP
$\text{ppmi}_{\text{Specific}}$	0.46
$\text{ppmi}_{\text{General}}$	0.50
$\text{ppmi}_{\text{WebIsA}}$	0.37
ppmi_{σ}	0.51
$\text{avg}(\text{ppmi}_{\text{Specific}}, \text{ppmi}_{\text{General}}, \text{ppmi}_{\text{WebIsA}})$	0.59
f_{agg}	0.71

3.3 Taxonomy extraction

This part of our work (contribution 3) is particularly different from other studies about taxonomy induction as we will search for a taxonomy minimizing information loss with the embedding. Other studies generally feed features and/or embedding into a classifier or a more complex model [2, 18, 23].

Minimizing the distortion gives an undirected graph, so the direction of the edges is recovered by rooting those at the centroid

of the embedding leading to acyclic connected directed graph interpreted as taxonomies. For instance, if we have undirected edges $E = \{(\text{bicycle}, \text{vehicle}), (\text{vehicle}, \text{car})\}$, and if the centroid is vehicle, then the taxonomy is (bicycle, *is-a*, vehicle) and (car, *is-a*, vehicle)

The main drawback of approximating taxonomies by rooted trees is that terms are constrained to have a single parent. Yet, the counterpart is that we can use results from graph theory to find rooted trees close to our embeddings. Three different versions of tree extractor are used, other solutions—aiming at minimizing the distortion—are possible and could be investigated in future work.

Hereafter, we present the three different tree extractors we propose in this work. The tree extractor is applied on the embedding produced in the previous step (see Sec. 3.2) with $\mathcal{D}^N$ obtained by our aggregated feature f_{agg}. At the end of each extraction process, the centroid is considered to be the root of the tree and edges are directed accordingly.

3.3.1 Naive Extraction (NE). This technique is very similar to [20] except that we extract an undirected tree which is then rooted at the centroid of the embedding space. The extraction relies on the simple idea that for hyperbolic embeddings, the hyperbolic distance encodes the strength of *is-a* relations and the direction of those can be determined by the Euclidean norm $\|.\|$. That is, for an embedding Θ with distance d and a term vector x, y^* is considered to be related to x and the edge (x, y^*) is added to the edge set of the tree if:

$$y^* = \arg\min_{y \in V} d(y, x) \text{ with } \|\theta_y\|_2 \leq \|\theta_x\|_2$$

One must notice that edges defined as such always provide a tree since the definition avoid cycles and multiple components.

3.3.2 MST Extraction (MSTE). As mentioned above, the MST corresponds to the tree that perfectly preserves pairwise distances of embedded element when it does exist. This is why we propose to use the MST because in some cases it can be a good spanning tree for the embedding.

3.3.3 Low Average Stretch Extraction (LASE). This stochastic algorithm was initially proposed by [7]. We slightly adapted it to support weighted graphs. This algorithm recursively decomposes the embedding space using spheres with specific radius to preserve the distances from the metric space to the tree. It requires a parameter controlling the scale of radius which was chosen according to experiment on the WordNet *mammal.n.01* sub-taxonomy.

3.4 Taxonomy Refinement

We also investigated taxonomy refinement as proposed in [2] which consists in the addition or deletion of aberrant edges with regard to external features. With the same idea, we only consider edge addition for our extracted taxonomy with pre-trained word embeddings from Spacy for each language [14]. Those preserving the taxonomy structure (no cycle) are added to the tree. Only disconnected terms are reconnected with this procedure.

4 EXPERIMENTS

We present here two series of experiments: the first assesses the extraction performance on embeddings issued from real taxonomies (WordNet [19]), the second evaluates our complete approach for taxonomy induction with embeddings calculated with the graph described in Sec. 3.1. Embeddings are generated as explained in the previous section; three distances are optimized: Euclidean $d_{\mathcal{E}}$ ($L2$ distance), Poincaré $d_{\mathcal{P}}$ and the squared Lorentzian $d_{\mathcal{L}}$. For each, we computed embeddings with different dimensions $D \in \{2, 5, 10, 20, 50, 100\}$ for 1, 500 epochs, and the dimension kept is the one maximizing Directed F1 on a validation set for each experiment. The rest of the hyperparameters are identical to previous work [22].

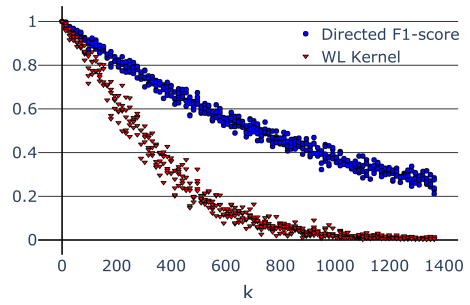

Figure 1: Directed F1-score and WL Kernel explained by the number of corruptions k.

4.1 Evaluation protocol

SOTA benchmarks principally rely on Directed F1-score. Here, we propose a more elaborated evaluation protocol which provides measures on the inner sub-structures of taxonomies (contribution 4), highlighting the similarity between taxonomy geometries. Our implementation relies on the libraries GMatch4py [9] and aproximated_ged [24]. Hereunder, we detail the evaluation scores for comparing a ground-truth taxonomy (with edges E^*) and a predicted taxonomy (edges E^p).

Directed F1-score is the standard combination of the precision $p = \frac{|E^* \cap E^p|}{|E^p|}$ and the recall $r = \frac{|E^* \cap E^p|}{|E^*|}$, which assesses the quality of predicted edges.

Transitive F1-score is an extension of the directed score (found in [10]) which measures the concordance between ancestors by connecting every node to each of its ancestors. Specifically, from an edge set E, we create a new set $T(E) = \{(y, x) | y \text{ is an ancestor of } x \text{ in } E\}$. Then we compute the F1-score as previously except that we replace edge sets with $T(E^*) \setminus E^*$ and $T(E^p) \setminus E^p$.

Weisfeiler Lehman Kernel score (WL Kernel) [29] is a kernel based graph comparison score with kernel focusing on subtree-like pattern. It thus quantifies the similarity between subtrees from the two graphs.

Shortest Path Kernel score (SP Kernel) is also a kernel based graph comparison score where kernels contain shortest path information between nodes.

Hausdorff Edit Distance (HED) [8] is an approximate edit distance for graphs like the Levenshtein distance for strings [8] by counting the number of nodes addition or deletion and edges addition or deletion needed to turn a graph to another one.

In order to emphasize the relevance of kernel-based metrics compared to Directed F1-score, we compare the behavior of both scores on artificially degraded taxonomies. Let $\mathcal{T}_h^n$ be the set of

perfect n-ary trees with height h. We denote by $\widetilde{\mathcal{T}}^n_{h,k}$ the set of trees constructed as follows:

(1) Let $t \in \mathcal{T}^n_h$.
(2) Repeat k times (the number of corruptions):
   - Randomly select a node x from t. If $y = \text{parent}(x)$ is higher up in t, then proceed, else skip.
   - Remove edges from x to any of its children.
   - Connect these new orphan nodes to y.

This process voluntarily introduces errors in trees from $\mathcal{T}^n_h$ that are very similar to real world mistakes for taxonomy induction (mismatch hypernyms with co-hyponyms). On Fig. 1, we compare trees from $\widetilde{\mathcal{T}}^4_{5,k}$ (k is varying) with their original version in $\mathcal{T}^4_5$ and reported Directed F1 and WL Kernel scores. We observe that Directed F1-score linearly decreases (Pearson's coefficient ≥ 0.98) while WL Kernel plummets more. With 600 corruptions, Directed F1-score is still decent whereas WL Kernel value is fairly lower. With so many corruptions, the inner subtree structures are very damaged w.r.t. the original tree and this is not reflected by Directed F1-score. Therefore, we advocate to use kernel metrics to better evaluate the quality of the predicted taxonomies (but we still report standard measures for comparison purposes in the following experiments).

4.2 Inferring WordNet Taxonomies

Before dealing with real taxonomy induction, we first evaluate our methodology on distinct WordNet [19] sub-taxonomies. Instead of using noisy directed edges, we directly embed its ground-truth edges. We work on sub-taxonomies because LASE does not scale on the whole WordNet taxonomy. Spearman's correlation coefficients between the Direct F1-score and the distortion for the distances $d_{\mathcal{E}}$, $d_{\mathcal{P}}$, $d_{\mathcal{L}}$ are respectively: -0.79 ($p < 0.01$), -0.89 ($p < 0.01$), 0.24 ($p < 0.17$). It shows that, for $d_{\mathcal{P}}$ and $d_{\mathcal{E}}$, high Direct F1-score are obtained when distortion is low, confirming our approach which tries to find low distortion tree for embeddings. Concerning $d_{\mathcal{L}}$, we do not observe any particular trend maybe due to the fact that it is not a proper distance, thus, the distortion seems less meaningful in this case.

Results for two sub-taxonomies—similar trends are observed for others—are presented in Table 2 (SP Kernel and HED metrics are omitted for computational reasons). For each pair of extractor / distance, we selected the dimension corresponding to the best Directed F1-score on the sub-taxonomy *communication.n.02*, which acted as a validation set.

For all experiments, embeddings using $d_{\mathcal{E}}$ are significantly less effective than those using hyperbolic distances $d_{\mathcal{P}}$ or $d_{\mathcal{L}}$. This is expected since hyperbolic spaces are known to better embed hierarchical relations than Euclidean space. MSTE is the best extractor for Euclidean embedding since NE uses hyperbolic heuristics which are not usable for Euclidean distance and LASE is hard to tune for this distance. The distance $d_{\mathcal{L}}$ is best-performing for any metrics and extractor except for transitive F1-score on *plant.n.02*. This is due to our rooting policy which sets the root of the tree at the centroid. When the centroid does not match the true root, then the transitive F1-score is highly affected since errors in high levels of the taxonomy have more impact than errors in low levels. This is why it is important to also look at the WL Kernel score as it is less affected by these errors and reflects structural differences. Given that, it appears that $d_{\mathcal{L}}$ is the best to preserve the taxonomy structure with the considered extractors and distances.

Concerning tree extractors, there are noticeable differences. The taxonomy structure (directed F1-score and WL Kernel) is best reconstructed by LASE for $d_{\mathcal{L}}$, NE for $d_{\mathcal{P}}$ and MSTE for $d_{\mathcal{E}}$. This crucial result highlights that one must adapt the tree extraction to the embedding space to yield the best reconstruction performance achievable. Overall, the best combination for WordNet reconstruction is LASE and $d_{\mathcal{L}}$. However, on the transitive F1-score this combination is less effective than others due to two reasons: the tree is ill-rooted and LASE decomposes the embedding space in several spheres which can potentially break high-level relations.

4.3 Taxonomy Induction

We highlight the versatility of our approach by applying it to the taxonomy induction task: SemEval-2016 task 13 [4]. This task is composed of multilingual taxonomies: 4 languages—English (EN), French (FR), Italian (IT), Dutch (NL)—and 3 domains—Science (Sc.), Food (Fo.), Environment (En.)— produce 12 taxonomies, one for each arrangement. For this problem, we compare with TAXI, USAAR, JUNLP (taxonomies for USAAR and JUNLP are found on the official SemEval'16 page) and TAXI with refinements (TAXI (ref.)) proposed in [2] (their source code is used to compute taxonomies with their method). Given their poor results for English, missing taxonomies for JUNLP and USAAR are not recalculated.

It is worth noting that the roots of taxonomies needs to be given to all methods except ours. In comparison to others, we are able to provide a candidate for the root—being the centroid—that is shown to be close to the truth: we reported in the table the depth of the centroid to indicate how far the centroid is from the real root. In most cases, we observe that the centroid matches the real root of the taxonomy (centroid depth = 0). Yet, we decided to root the tree at the true root for a fair comparison between models.

The English taxonomy covering environment is used to tune hyper-parameters: the distance, the tree extractor and the dimension. We found that the best combination is $d_{\mathcal{P}}$ with NE and 100 dimensions. Results are reported in Table 3. When method is followed by (ref.), it indicates whether the system uses refinements posterior to the taxonomy prediction.

We observe our propositions (refined or not) to be slightly less effective than TAXI (ref.) at predicting directed edges of the ground truth taxonomy. Indeed, our method has a slightly lower recall but is more precise than TAXI (ref.) potentially due to the tree approximation: we prevent nodes from having multiple parents which limit the number of predicted edges. Nevertheless, our propositions are more effective on other metrics. This is significantly observed on WL Kernel and SP Kernel for which our propositions conduct to fairly higher scores. HED results are not significant but show an identical trend. For Transitive F1-score, which highlights the ability at providing stereotypical features, our proposition with refinement gives best results, while TAXI (ref.) and our method without refinements are comparable. Overall, our propositions, with or without refinements, predict taxonomies with a graph structure that is closer to the target taxonomy, without greatly degrading the performance of TAXI (ref.) on directed F1-score. It also gives higher performance than other methods. Two main reasons can

Table 2: Reconstruction results on two sub-taxonomies of WordNet. Dimensions are tuned on the sub-taxonomy *communication.n.02* with *Directed F1-score*. Best results are bolded.

Sub-Taxonomy ($\|V\|$, $\|E\|$)	Tree Extraction	Distance	Dimension	Centroid	Directed F1-score	Transitive F1-score	WL Kernel
cognition.n.01 (3999, 4033)	NE	$d_{\mathcal{E}}$	5	basic_cognitive_process.n.01	0.2146	0.1183	0.0477
		$d_{\mathcal{P}}$	100	cognition.n.01	0.391	0.6129	0.2494
		$d_{\mathcal{L}}$	100	cognition.n.01	**0.5466**	**0.7615**	**0.7341**
	MSTE	$d_{\mathcal{E}}$	100	cognition.n.01	0.249	0.2754	0.0402
		$d_{\mathcal{P}}$	100	cognition.n.01	0.373	0.6048	0.2436
		$d_{\mathcal{L}}$	100	cognition.n.01	**0.6074**	**0.9098**	**0.7263**
	LASE	$d_{\mathcal{E}}$	5	basic_cognitive_process.n.01	0.0853	0.0596	0.0658
		$d_{\mathcal{P}}$	10	concept.n.01	0.3073	0.2037	0.1593
		$d_{\mathcal{L}}$	50	cognition.n.01	**0.7805**	**0.5961**	**0.8954**
plant.n.02 (4487, 4493)	NE	$d_{\mathcal{E}}$	5	vascular_plant.n.01	0.1568	0.1397	0.0147
		$d_{\mathcal{P}}$	100	vascular_plant.n.01	0.5439	**0.5908**	0.8565
		$d_{\mathcal{L}}$	100	herb.n.01	**0.6243**	0.4102	**0.9256**
	MSTE	$d_{\mathcal{E}}$	100	vascular_plant.n.01	0.1795	0.2467	0.0153
		$d_{\mathcal{P}}$	100	vascular_plant.n.01	0.517	**0.5751**	0.852
		$d_{\mathcal{L}}$	100	herb.n.01	**0.6341**	0.5387	**0.924**
	LASE	$d_{\mathcal{E}}$	5	vascular_plant.n.01	0.09	0.231	0.0345
		$d_{\mathcal{P}}$	10	vascular_plant.n.01	0.4633	**0.4984**	0.4603
		$d_{\mathcal{L}}$	50	herb.n.01	**0.9032**	0.349	**0.9789**

explain these differences: (1) the new feature is more effective than previously used features, (2) the graph pruning step of TAXI is brutal, as it uses individual edge classification, while we extract a coherent tree structure from our embedding space in a single step.

5 CONCLUSION

In this work, we proposed a new approach to extract tree directly from an embedding space. This method reduces the need for supervision in taxonomy induction system. It can be applied to any embedding computed with any kind of data source—similarities, relations, raw text—which can extend taxonomy induction to other data sources. We quantitatively show that it produces taxonomy with a better structure. This methodology was applied to WordNet term embeddings to extract the latent hierarchical representations with effective performance. Our code will be made available for reproducibility purposes at https://github.com/pagesjaunes/dembedder/. In addition, we extended the commonly used evaluation protocol—mainly composed by the directed F1-score—which gives quantitative insights on the inner structure of the taxonomy. We hope that this new benchmark will provide a more detailed comparison between forthcoming methods.

In the future, we propose to search for other tree extractor algorithms, as well as graph extractors (removing the tree constraint of our proposition). It may involve neural networks similar to [28] or may require to jointly optimize the tree/graph structure with the embedding, adapting the proposition of [20] to any extractor. Ideally, our method can be applied to any embedding space; it would be interesting to investigate which type of taxonomies it can produce with embeddings learned with different data sources than graph, or noisy graph and different data types.

REFERENCES

[1] Reyan Ahmed, Greg Bodwin, Faryad Darabi Sahneh, Keaton Hamm, Mohammad Javad [Latifi Jebelli], Stephen Kobourov, and Richard Spence. 2020. Graph spanners: A tutorial review. *Computer Science Review* 37 (2020).

[2] Rami Aly, Shantanu Acharya, Alexander Ossa, Arne Köhn, Chris Biemann, and Alexander Panchenko. 2019. Every Child Should Have Parents: A Taxonomy Refinement Algorithm Based on Hyperbolic Term Embeddings. In *ACL*. ACL, 4811–4817.

[3] Mohit Bansal, David Burkett, Gerard de Melo, and Dan Klein. 2014. Structured Learning for Taxonomy Induction with Belief Propagation. In *ACL*. ACL, 1041–1051.

[4] Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. Semeval-2016 task 13: Taxonomy Extraction Evaluation (TExEval-2). In *SemEval*. ACL.

[5] Catherine Chen, Kevin Lin, and Dan Klein. 2021. Constructing Taxonomies from Pretrained Language Models. In *NAACL*. ACL, 4687–4700.

[6] Guillaume Cleuziou, Davide Buscaldi, Gael Dias, Vincent Levorato, and Christine Largeron. 2015. QASSIT: A Pretopological Framework for the Automatic Construction of Lexical Taxonomies from Raw Texts. In *SemEval*. ACL, 955–959.

[7] Kedar Dhamdhere, Anupam Gupta, and Harald Räcke. 2006. Improved Embeddings of Graph Metrics into Random Trees. In *SODA*. SIAM, 61–69.

[8] Andreas Fischer, Ching Y. Suen, Volkmar Frinken, Kaspar Riesen, and Horst Bunke. 2015. Approximation of graph edit distance based on Hausdorff matching. *Pattern Recognition* 48, 2 (2015), 331–343.

[9] Jacques Fize. 2018. GMatch4py. https://github.com/Jacobe2169/GMatch4py.

[10] Octavian Ganea, Gary Becigneul, and Thomas Hofmann. 2018. Hyperbolic Entailment Cones for Learning Hierarchical Embeddings. In *ICML*, Vol. 80. PMLR, 1646–1655.

[11] Amit Gupta. [n.d.]. Automated Taxonomy Induction and its Applications.

[12] Amit Gupta, Rémi Lebret, Hamza Harkous, and Karl Aberer. 2017. Taxonomy Induction Using Hypernym Subsequences *(CIKM)*. ACM, 1329–1338.

[13] Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *COLING*.

[14] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. *spaCy: Industrial-strength Natural Language Processing in Python.*

[15] Marc Law, Renjie Liao, Jake Snell, and Richard Zemel. 2019. Lorentzian Distance Learning for Hyperbolic Representations. In *ICML*, Vol. 97. PMLR, 3672–3681.

[16] Matthew Le, Stephen Roller, Laetitia Papaxanthos, Douwe Kiela, and Maximilian Nickel. 2019. Inferring Concept Hierarchies from Text Corpora via Hyperbolic Embeddings. In *ACL*. ACL, 3231–3241.

[17] Promita Maitra and Dipankar Das. 2016. JUNLP at SemEval-2016 Task 13: A Language Independent Approach for Hypernym Identification. In *SemEval*. ACL, 1310–1314.

[18] Yuning Mao, Xiang Ren, Jiaming Shen, Xiaotao Gu, and Jiawei Han. 2018. End-to-End Reinforcement Learning for Automatic Taxonomy Induction. In *ACL*. ACL, 2462–2472.

[19] George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38, 11 (Nov. 1995), 39–41.

Table 3: Taxonomy Induction on SemEval 2016 Task 13 [4]. Significance of experiments *: p < 0.1, **: p < 0.05, ***: p < 0.01, with a one-sided paired t-test. Best results are bolded.

Score	Method	EN		FR			IT			NL		
		Sc.	Fo.	Sc.	Fo.	En.	Sc.	Fo.	En.	Sc.	Fo.	En.
Centroid depth	Ours	0	0	0	1	0	2	2	1	0	3	3
Directed F1-score	Ours**	0.396	0.342	**0.348**	0.252	0.261	0.338	0.195	0.24	0.322	0.215	0.183
	Ours (ref.)***	0.40	0.337	0.319	0.25	0.249	0.329	**0.197**	0.225	0.293	0.211	0.173
	TAXI	0.369	0.307	0.321	0.232	0.244	0.323	0.172	0.31	0.298	0.197	0.285
	TAXI (ref.)	**0.417**	**0.374**	0.334	**0.295**	**0.29**	**0.346**	0.185	**0.309**	0.305	**0.218**	**0.266**
	JUNLP	0.3232	0.253	0.324	0.231	-	0.336	0.187	-	**0.325**	0.225	-
	USAAR	0.373	0.268	-	-	-	-	-	-	-	-	-
Transitive F1-score	Ours	0.489	0.408	0.407	0.335	0.507	0.316	0.088	0.38	0.364	0.267	0.399
	Ours (ref.)**	0.497	0.42	**0.453**	**0.354**	**0.547**	0.335	**0.271**	**0.434**	**0.396**	**0.279**	**0.434**
	TAXI	0.501	0.367	0.376	0.218	0.316	0.367	0.214	0.288	0.365	0.226	0.305
	TAXI (ref.)	**0.559**	**0.452**	0.391	0.301	0.375	**0.383**	0.233	0.304	0.379	0.258	0.304
	JUNLP	0.357	0.253	0.269	0.12	-	0.181	0.093	-	0.20	0.114	-
	USAAR	0.191	0.115	-	-	-	-	-	-	-	-	-
WL Kernel	Ours***	0.646	**0.397**	0.517	0.293	0.423	0.484	0.271	**0.598**	0.473	0.303	**0.518**
	Ours (ref.)*	**0.647**	0.393	**0.583**	**0.306**	0.309	**0.559**	**0.326**	0.405	**0.597**	**0.309**	0.393
	TAXI	0.396	0.205	0.362	0.14	0.366	0.42	0.144	0.41	0.366	0.122	0.349
	TAXI (ref.)	0.528	0.348	0.419	0.26	**0.503**	0.465	0.150	0.4849	0.393	0.166	0.436
	JUNLP	0.439	0.173	0.261	0.16	-	0.415	0.169	-	0.398	0.203	-
	USAAR	0.470	0.239	-	-	-	-	-	-	-	-	-
SP Kernel	Ours***	0.762	0.777	0.881	0.986	**0.741**	**0.911**	0.952	**0.895**	**0.809**	**0.981**	**0.957**
	Ours (ref.)***	**0.787**	**0.810**	**0.901**	**0.992**	0.732	0.91	**0.962**	0.87	0.778	0.968	0.952
	TAXI	0.466	0.501	0.569	0.895	0.455	0.383	0.509	0.656	0.405	0.573	0.719
	TAXI (ref.)	0.526	0.643	0.573	0.982	0.592	0.418	0.557	0.664	0.406	0.572	0.922
	JUNLP	0.714	0.779	0.587	0.849	-	0.506	0.743	-	0.309	0.782	-
	USAAR	0.06	0.116	-	-	-	-	-	-	-	-	-
HED	Ours	**11**	69	**6**	79	**11**	7	72	**10**	**5**	87	6
	Ours (ref.)	**11**	68	11	**76**	22	8	**55**	23	7	**72**	15
	TAXI	17	67	8	87	16	9	92	7	18	85	7
	TAXI (ref.)	20	**57**	11	82	15	8	78	11	21	83	**4**
	JUNLP	17	123	22	83	-	**6**	101	-	10	85	-
	USAAR	12	111	-	-	-	-	-	-	-	-	-

[20] Nicholas Monath, Manzil Zaheer, Daniel Silva, Andrew McCallum, and Amr Ahmed. 2019. Gradient-based Hierarchical Clustering Using Continuous Representations of Trees in Hyperbolic Space. In *KDD*. ACM, 714–722.

[21] Amir Nayyeri and Benjamin Raichel. [n.d.]. Viewing the Rings of a Tree: Minimum Distortion Embeddings into Trees. In *SODA*. 2380–2399.

[22] Maximillian Nickel and Douwe Kiela. 2017. Poincaré Embeddings for Learning Hierarchical Representations. In *NIPS*. Curran Associates, Inc., 6338–6347.

[23] Alexander Panchenko, Stefano Faralli, Eugen Ruppert, Steffen Remus, Hubert Naets, Cédrick Fairon, Simone Paolo Ponzetto, and Chris Biemann. 2016. TAXI at SemEval-2016 Task 13: a Taxonomy Induction Method based on Lexico-Syntactic Patterns, Substrings and Focused Crawling. In *SemEval*. ACL, 1320–1327.

[24] Paul Riba and Anjan Dutta. 2017. Aproximated Graph Edit Distance. https://github.com/priba/aproximated_ged.

[25] Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst Patterns Revisited: Automatic Hypernym Detection from Large Text Corpora. In *ACL*. ACL, 358–363.

[26] Rik Sarkar. 2012. Low Distortion Delaunay Embedding of Trees in Hyperbolic Plane. In *Graph Drawing*. Springer, 355–366.

[27] Julian Seitner, Christian Bizer, Kai Eckert, Stefano Faralli, Robert Meusel, Heiko Paulheim, and Simone Paolo Ponzetto. 2016. A Large DataBase of Hypernymy Relations Extracted from the Web.. In *LREC* (Portorož, Slovenia, 23-28). ELRA.

[28] Chao Shang, Sarthak Dash, Md. Faisal Mahbub Chowdhury, Nandana Mihindukulasooriya, and Alfio Gliozzo. 2020. Taxonomy Construction of Unseen Domains via Graph-based Cross-Domain Knowledge Transfer. In *ACL*. ACL, 2198–2208.

[29] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. 2011. Weisfeiler-Lehman Graph Kernels. *JMLR* 12, 77 (2011), 2539–2561.

[30] Liling Tan, Francis Bond, and Josef van Genabith. 2016. USAAR at SemEval-2016 Task 13: Hyponym Endocentricity. In *SemEval*. ACL, 1303–1309.

[31] Chengyu Wang, Xiaofeng He, and Aoying Zhou. 2017. A Short Survey on Taxonomy Learning from Text Corpora: Issues, Resources and Recent Advances. In *EMNLP*. ACL, 1190–1203.

[32] Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising Measures of Lexical Distributional Similarity. In *COLING*. 1015–1021.