information

Article

Fully-Unsupervised Embeddings-Based Hypernym Discovery

Maurizio Atzori [1,*,†] and Simone Balloccu [2,*,†]

1 DMI, University of Cagliari, 09124 Cagliari, Italy
2 Computing Science Department, University of Aberdeen, Aberdeen AB24 3FX, UK
* Correspondence: atzori@unica.it (M.A.); simone.balloccu@abdn.ac.uk (S.B.)
† Authors contributed equally to this work.

Received: 15 April 2020; Accepted: 11 May 2020; Published: 18 May 2020

Abstract: The hypernymy relation is the one occurring between an instance term and its general term (e.g., "lion" and "animal", "Italy" and "country"). This paper we addresses Hypernym Discovery, the NLP task that aims at finding valid hypernyms from words in a given text, proposing HyperRank, an unsupervised approach that therefore does not require manually-labeled training sets as most approaches in the literature. The proposed algorithm exploits the cosine distance of points in the vector space of word embeddings, as already proposed by previous state of the art approaches, but the ranking is then corrected by also weighting word frequencies and the absolute level of similarity, which is expected to be similar when measuring co-hyponyms and their common hypernym. This brings us two major advantages over other approaches—(1) we correct the inadequacy of semantic similarity which is known to cause a significant performance drop and (2) we take into accounts multiple words if provided, allowing to find common hypernyms for a set of co-hyponyms—a task ignored in other systems but very useful when coupled with set expansion (that finds co-hyponyms automatically). We then evaluate HyperRank against the SemEval 2018 Hypernym Discovery task and show that, regardless of the language or domain, our algorithm significantly outperforms all the existing unsupervised algorithms and some supervised ones as well. We also evaluate the algorithm on a new dataset to measure the improvements when finding hypernyms for sets of words instead of singletons.

Keywords: natural language processing; natural language understanding; unsupervised learning; hypernym discovery; word embeddings; word2vec

1. Introduction

Motivated by our OKgraph project [1,2], in this paper we address one of the most important linguistic relationships that can be found within a knowledge graph: hypernymy, that is, the is-a relation of conceptual ontologies that relates subordinate words (e.g., "lion") with its superordinate (e.g., "animal"). This relation lies at the basis of human cognition, and allows to compare words representing specific instances to generic ones; moreover it is of critical importance in any taxonomy [3]. Some examples of hypernymy are shown below:

Mango **IS A** fruit
Africa **IS A** continent
Barack Obama **IS A** politician

The most general entity, such as "fruit" in the first sentence, is called hypernym, while the most specific one, such as "mango", is called hyponym.

The main NLP tasks related to hypernymy in a given corpus are sometimes called Taxonomy Learning:

1. *taxonomy expansion*, where an initial seed taxonomy (set of pairs such as <mango,fruit> and <fruit,food>) is given and this must be made incremented with new pairs;
2. *hypernym detection*, that aims at determining whether a given pair of words has a hypernymy relation. This can be seen as a binary classification problem, where given a pair such as <mango,fruit> or <potato,fruit>, it answers positive or negative—that can be used as a building block for taxonomy expansion;
3. *hypernym discovery*, a more recent and challenging task w.r.t. hypernym detection, where a list of valid hypernyms must be retrieved starting from a given hyponym.

In this paper we focus on the last task. Furthermore, while most existing approaches in the literature are supervised, therefore relying on manually-crafted labeled training data, manually crafted patterns, assumptions on the corpus language, here we do not assume any of these.

Therefore our approach is:

- **Unsupervised**: does not require user intervention; has no supervised training steps; does not use any tool that require or is obtained through the use of supervision such as pre-trained artificial neural networks, POS taggers, knowledge bases, etc.
- **Language independent**: no specific language characteristics or structure is exploited. This includes Hearst patterns, stopwords lists etc.
- **Unstructured-data driven**: the algorithm exploits plain text as the only input, without structured information like tags, links, hierarchies etc.

The solution we propose is based on using words as candidate hyponyms and finding the corresponding hypernyms. Differently from the classical setting, we also accept sets of words as candidates, not necessarily single words. For instance, given the set APPLE, PEAR we may result in the corresponding hypernym FRUIT, while having only APPLE may lead to COMPANY. We stress that multiple co-hyponyms are not required, our algorithm also work with single words as in the classical setting.

In order to generate the correct hypernym, the algorithm exploits the distance of a candidate word with nearest words in the vector space induced by word embeddings. Although word embeddings have been already proposed in the context of taxonomy learning, only a few attempts uses them for unsupervised hypernym discovery, facing the problem that the vector space express semantic relations that may not correspond to hypernym relations, with a significant drop of performance w.r.t. supervised approaches. Instead, we weighted the ranking induced by cosine similarity in the vector space by using word frequencies computed from the corpus. In fact, given a word, hypernyms are more frequent than other similar words, and this new weights lead to better performance in term of precision in hypernym discovery. Finally, when multiple words are provided (co-hyponyms), we filter out from the list of candidate hypernyms those having a large variance in the distance with the hyponyms. The intuition is that "apple" and "pear" are expected to be similar to "fruit" to the same degree. HyperRank not only consistently outperforms all existing unsupervised approaches, but it also compares and outperforms a number of supervised ones, as shown in the experiment section.

The paper continues with the following structure—in Section 2 we describe the various approaches in the literature that are related to taxonomy learning and to hypernym discovery in particular. Section 3 describes our unsupervised approach to hypernym discovery and the 3 steps performed to compute the results. Its pseudocode is provided in Section 4. Then, we proceed to evaluate HyperRank in Section 5 on the SemEval 2018 Hypernym Discovery Task, showing performance compared to the other unsupervised algorithms which took part to the competition. Further, in Section 6 we raise some questions regarding the SemEval testset and in Section 7 we propose a custom benchmark which allows to evaluate HyperRank in different conditions and show last results and considerations. Conclusions are drawn in Section 8, with discussion and possible future work.

2. Related Works

Hypernymy-related approaches typically exploit either existing labeled data extracted from the Web (e.g., Reference [4]) or knowledge bases such as DBpedia [5], depending on the specific task. Algorithms that tries to recognise hypernymy can generally be divided into two categories—pattern-based and distributional methods [6]. Pattern-based [7] methods exploits textual patterns to extract hypernyms—it is effective but time consuming, language dependent and prone to errors due to the continuous language evolution. Distributional methods aim to extract hypernyms by inspecting word distribution with either supervised models (when a training set is available) or unsupervised similarity measures—these approaches typically struggle when training and evaluation set significantly differ. There are a good number of approaches in the literature which address hypernymy detection—Snow et al. [8] replace classical Hearst patterns with a supervised classifier which is trained on WordNet hyponym-hypernym pairs and related sentences. The sentences are parsed in order to extract a number of lexycon-syntactic patterns, which are exploited to train the classifier. HyperVec [9] is a neural system that learns "hierarchical embeddings" to perform hypernym detection and directionality and showed improvements compared to classical methodologies in both tasks. The system tries to create a model in which hyponyms and hypernyms are close points in space through the use of distributional similarity. This is achieved by using the skip-gram algorithm whose training set is a set of hypernymy relationships extracted from WordNet. DIVE [10] adopts an unsupervised embeddings-based system in which the vectors are obtained by exploiting the distributional inclusion hypothesis. The system has been tested on many different hypernym detection test sets, showing higher performances than the classic unsupervised approaches in most cases. Le et al. [11] infers hierarchies by combining Hearst patterns with hyperbolic embedding, thus gaining some advantages including consistency towards hypernymy features and efficiency in terms of model size. The system achieves SOTA performances in hypernymy detection, direction and graded entailment. Bi-GRU-CapsNet [12] exploits GRU together with an hybrid attention mechanism to better recognise hypernymy in pairs which contain compound elements (n-grams) in medical domain. The system separately encodes Chinese and English language terms to handle unknown tokens and the attention mechanism is designed to focus on medical compound entities. DIVE [10] is an unsupervised embeddings-based system—the vectors are obtained by exploiting the distributional inclusion hypothesis (DIH) according to which the context set of a hypernym tends to contain one of its hyponyms. The system has been tested on many different hypernym detection testsets, showing higher performances than the classic unsupervised approaches in most cases. HypeNet [13] adopts a path-based approach which extracts dependency paths, encodes them with RRN and combines the result with distributional models.

Given the recent formalisation of hypernym discovery there's a number of recent works, together with older ones. HypernymFinder [14] is collection of algorithms that use lexical patterns and text statistics to improve the recall of classic systems. Despite being proposed way before the formalisation and tested on detection datasets, HypernymFinder returns a ranked list of hypernyms for a given input, hence performing hypernym discovery. The first algorithm extracts Hearst patterns through another proprietary system [15] and showed an higher precision; the second one is a Support Vector Machine (SVM) that assigns a probability (obtained via various features such as frequency, total number of full or partial hearst pattern match and other one) to every hypernym-hyponym pair and showed higher performances than the one based on statistics; finally, the third model uses an Hidden Markov Model that computes the probability for each pair of words and proved to be the best performing one. Fu et al. [16] present a Chinese language framework for hypernym discovery, based on distant supervision. The approach finds all the hypernym candidates for a given hyponym through web scraping from various sources, hence emulating an online search. The result of this procedure is then filtered by co-occurrence and ranked (based on metrics such as presence of a category tag that bears the name of the candidate or the number of pages in which it appears in order to extract candidates). The framework has been tested on different domains (biology, food, movies etc.) and was manually evaluated. Yamada et al. [17] present a hypernyms discovery system based on a distributional similarity. It checks the similarity between words and a list of similar terms, that are extracted according to frequency similarity. The system was tested on relationship extracted from Wikipedia and showed a remarkably high effectiveness. Taxoembed [18] is a supervised distribution framework whose purpose is the extraction of disambiguated taxonomies. The disambiguation is achieved by using "sense-aware word embeddings" that allow to capture multiple meanings for a single term. The system has been tested in various configurations, training sets (such as WikiData, KB-Unify and BabelNet) and semantic areas (art, biology, health, music etc.) hinting at the fact that it could be possible to find a linear projection that can be used to derive a transformation matrix that is suitable for hypernyms extraction. Doval et al. [19] builds a multi-lingual word embedding model which exploits non-orthogonal transformation to make different languages vectors of the same word closer to their average, hence achieving hypernym discovery in multi-language case. Palm et al. [20] present an algorithm which exploits a domain-specific vocabulary and WordNet in order to automatically extract hypernym discovery training pairs, which can then ben used in order to train a supervised model. The results showed a significant increase in MAP when such a methodology is adopted, and its usefulness in domain adaption. ADAPT [21] adopts unsupervised word embeddings and retrieves hypernyms by means of a nearest neighbour extraction, then sort the candidates by semantic similarity. EXPR [22] adopts a dependency parser which extracts hypernyms from text, then combines it word embeddings to build a classifier. UMDUluth [23] extracts hypernyms frequencies by repeatedly parsing a given text and merging distributional embeddings to rank them. Hashimoto et al. [24] adopts a bidirectional LSTM setup that generates a language model that strengthens the DIH to perform hypernym discovery. Apollo [25] combines rule-based systems with dependency patterns and POS tagger. SJTU [26] inspects various NN architectures along with sense embeddings to model the learning of words and phrases in latent spaces. Dash et al. [27] achieves various hypernym-related tasks, including detection and discovery, by employing a novel supervised neural model named Strict Partial Order Network (SPON), which tries to capture and enforce asymmetry and transitivity in relations, properties which hypernymy possess. Experiments on both normal and augmented variant of SPON shows it is capable of outperforming SOTA in various datasets. NLP_HZ [28] applies supervised projection learning to get a linear transformation which maps hyponyms to hypernyms. 300-Sparsans [29] combines sparse word representation and formal component analysis and employs a set of linguistic feature to compare queries with hypernym candidates. Held et al. [30] adopts an hybrid approach which combines Hearst patterns and nearest neighbours obtained from a distributional model. Aldine et al. [31] achieve hypernym discovery by enriching Hearst patterns with grammatical information, which are obtained from a dependency pattern. Experiments showed a significative increase in recall and a modest increase in precision, compared to using classic regex patterns. CRIM [32] combines projection learning, trying to achieve a linear transformation which goes from hyponyms to hypernyms, with pattern-based extraction. It is also worth mentioning that recognising hypernymy represents a fundamental feature in taxonomy induction and enrichment [18,33–35]. When evaluating our approach we compare ourselves with all the systems who took part in SemEval 2018 Hypernym Discovery task. This is due to the fact that other systems which were mentioned before were manually evaluated, or used a custom test set which is not retrievable. Approaches which we compare with are CRIM [32], NLP_Hz [28], 300-sparsans [29], SJTU [26] Apollo [25], MFH [36], TaxoEmbed [18], APSyn [37], balAPInc [38] and SLQS [39]). In this regards we compare ourselves with three more systems which took part to the competition but did not publish any paper: MSGC-SANITY, Team 13 and Anu, which were listed in the original SemEval paper [36]. Finally, two more approaches are included in the evaluation but with partial results since the authors did not publish the other metrics: MEEMI [19] and the hybrid approach proposed by Held et al. [30] which is listed as "Hybrid of SVD & NN".

3. Our Proposal: Re-Ranking Semantic Similarity Based on Word Frequencies

Our proposal is made of three main steps, each accomplished by its corresponding component:

- **Embedding component**: employs a dense word embedding model to compare the corpora words at a distributional level.
- **Extraction component**: extracts potential hypernyms
- **Ranking component**: assigns a score to the candidates and returns a ranked list whose target is to get the actual hypernyms to the lowest possible position. It also provides an extra filtering function to remove false positives.

The embedding component exploits word embeddings, in order to perform a much deeper analysis than the one that classical strings allow, without any pattern matching step. The Candidates Extraction component provides a first filter to recognise false positives which could occur when comparing words' similarity via embeddings (this detail will be expanded later). The Ranking component realizes the final part of the hypernym discovery, returning a list in which each possible hypernym is assigned a score. In the following we describe in detail each component.

3.1. Embedding Component

The first step is obtaining a suitable language model that will constitute the embedding component. This could be trivially done by just taking any suitable text and feeding it to the chosen algorithm. However, before training the model we proceed to perform some pre-processing steps—please note that we use the word "training" in this context as a general term: we adopts Word2Vec which is recognised to be an unsupervised model, so no supervised "training" is involved. First the whole corpus is lowercased. This step does not introduce any kind of knowledge that could boost the algorithms; in fact, lowercased text increases the text ambiguity (i.e., it is far more simple to distinguish "Apple" as a company from "apple" as a fruit). Then all the punctuation is removed: this is a common NLP step that avoids the generation of multiple useless token like "cat","cat!","cat:"... and similar ones. After doing said steps there's still one particular case to cover: managing concepts expressed through multiple tokens. This is one of the most complex cases to deal with in the task of extracting hypernyms. In this regard, consider the following examples:

- Barack Obama
- New York Times
- New Zealand

The listed entities all feature the presence of multiple tokens. In the first example "Barack Obama" is a particularly fortuitous case, because by taking any text it is highly probable that every time we talk about "Barack" or "Obama" we are referring to the former American president. However in the case of "New York Times" the three tokens assume multiple meanings, all of which are very common. Capitalized text could help distinguish the cases but we do not recommend exploiting capitalization as it would generate an effectiveness asymmetry—an algorithm that works on lowercased text will be valid on capitalised one as well, while the opposite is generally not true. An intuitive idea, also adopted by the ADAPT system [21], is to obtain an "hybrid" vector by averaging the vectors of each individual token. However W2V skip-gram is a paradigmatic approach: it guarantees the closeness of tokens which appear in similar contexts, but does not necessarily take in account if tokens are close in sentences. In this regards, consider the first 10 neighbours of the average vector of "Steve Jobs", using the W2V model obtained from the UMBC corpus:

{jobs, steve, high-paying, low-paying, well-paying, bryant, high-wage, gwen, alex, kidsinpear}

It can be seen that the vector is far from being ideal. Although the criticisms made about the approach are valid in some cases, in others it is possible to obtain in context vectors. Not being able to verify if the method is valid for the majority of multi-token concepts, it has been implemented and tested. Finally it is important to consider that this technique cannot extract multi-token hypernyms (like "Head of State" for "Barack Obama")—this is currently a limit of HyperRank.

3.2. Extraction Component

The hypernym discovery task requires finding a set of candidates who could be hypernym for a given hyponym. Our approach exploits the neighbours of the hyponym vector (Nearest Neighbours Search or NNS) based on cosine similarity. We will call it "neighbourhood" of an hyponym from this point. The neighbourhood size is critical and must be sufficiently wide. The system manages the extraction in two main ways:

- **Singleton labeling**: if the seed-set is a singleton, the neighbourhood size is a simple parameter which can be specified by the user. During implementation, the idea of introducing a data-augmentation phase was considered, similarly to the one adopted by CRIM system [32], using the first N neighbours of the input to generate a larger seed-set but its introduction did not produce significantly better results.
- **Set Labeling**: in this case we make a strong assumption: given that the starting set must necessarily contain hyponyms that share the same hypernym, then each hyponym's neighbourhood should intersects at some point. For example it is expected that the neighbours of the continents should all contain the word "continent" at some point. Consequently in this case the algorithm executes an NNS for each seed and verifies if they overlap. When running the algorithm the user can specify how many candidates must be returned—if the neighbourhood intersection does not return enough elements, then a wider neighbourhood is explored. This goes on until the requested number of candidates is fetched or a stop criterion is met.

Moreover, in Set Labeling case, the user can specify how many neighbourhoods should be covered by a single candidate. In other words, given N hyponyms as input, the user specifies that a neighbour could be an hypernym if and only if it appears in the at least n neighbourhoods with $n \leq N$. This parameter is expressed as a percentage so, for example, specifying 100% means that a neighbour is a valid candidate if and only if it occurs in the neighbourhood of every hyponym.

3.3. Ranking Component

Final step involves taking all the candidates and returning a ranked list. Here the goal is to get every actual hypernyms the lowest possible position in list, and to limit the presence of false positives where possible. In this section we formulate the definition of the ranking criteria. Using word embeddings and given the explanation of the previous two sections, it is intuitive to suppose that the presence of is-a relationships could be investigated by means of one of the classic similarity measures, such as cosine similarity in the case of Word2Vec, which is the same one which is used to get the neighbourhood of a word. However this choice turns out to be immediately erroneous—cosine similarity returns a list of neighbours who share a semantic similarity with the starting vector. This concept is so general that it applies to practically everything. Consider the size 10 neighbourhood of the "Italy" vector, obtained by NNS on Google News' Word2Vec skip-gram model and ranked on cosine similarity:

{Italian, Sicily, Italians, ITALY, Spain, Bologna, France, Milan, Romania}

The geographical context linked to "Italy" is present but noisy—there is information about cities, languages, nationalities, states, making the starting list useless. Also note that not a single hypernym is in sight and that if we consider a non-geographical meaning for "Italy" (such as the football team), there's no trace of such information in the list, probably because it is not the most frequent meaning of the term within the corpus. Based on the above considerations, it was necessary to study a composite measure to prioritize hypernymy. HyperRank combines three metrics in order to improve hypernyms position in the final list and we expose and motivate them in the following sections.

3.3.1. Heuristics and Re-Ranking Measure

Now we define the individual metrics which are combined in order to rank hypernym candidates. The first one is semantic similarity. In our case, adopting Word2Vec, we chose descending cosine similarity. Using this measure may appear contradictory to everything that has been written so far. Please note, however, that the considerations regarding similarity set out above refer to its exclusive use. Moreover, given an hyponym, its hypernym can always be reached through a sufficiently extensive NNS. The second metric which we take in account when ranking is the candidate frequency in the given text. We now formulate a linguistic assumption in order to motivate this choice:

Heuristic 1 (Hypernym frequency assumption)**.** *Let h be a given hyponym. Let $\{S_c\}$ be a set of candidate words wich are semantically similar to h. Suppose that $\{S_c\}$ contains a candidate $\widetilde{h}$ that is a valid hypernym for h. Then $\widetilde{h}$ always features an higher frequency compared to the other elements of $\{S_c\}$ which are not valid hypernyms.*

This assumption stems from the Distributional Inclusion Hypothesis: hypernyms occurs in the majority of contexts that concern *h*: for example it is very unlikely to write about any food without using one or more hypernyms ("food" itself or "foodstuff" for example). This results in a considerably higher frequency, Note that this does not apply to all hypernyms since some could be used less frequently. It could be argued that stopwords has an high frequency as well but most word embeddings implementation cuts out such noisy elements without the need of human intervention. We further formulate the following linguistic assumption, which strengthens the contexts in which hypernyms are likely to occur, and gives us a third metric which is used only in Set Labeling:

Heuristic 2 (Hypernym similarity variance)**.** *Let $\{S_h\}$ be a set of hyponyms. Let $\{S_c\}$ be a set of words which are semantically linked to one or more elements of $\{S_h\}$. Let $\widetilde{h} \subseteq \{S_c\}$ be a valid hypernym for $\{S_h\}$. Let $var_{sim}(\widetilde{h})$ be $\widetilde{h}$ similarity variance, that is obtained by calculating the similarity of $\widetilde{h}$ with every element of $\{S_h\}$ and then computing the variance. Then $var_{sim}(\widetilde{h})$ is always lower than the similarity variance of any other $\{S_c\}$ element that is not a valid hypernym.*

This assumption derives from the Distributional Inclusion Hypothesis as well: usually there's a subset of contexts in which hypernyms occurs that stays fixed for any valid hyponym. Taking countries as an example, they will interact within a text in very different contexts—political relations, conflicts in which some of them took part and so on. However, it is logical to expect that the "country" hypernym should appear for the majority of countries in very similar and relatively "fixed" contexts, for example during the definition of the state itself. All together we define the following composed metric to rank a candidate:

$$rank(\widetilde{h}) = sim(\widetilde{h}, h_i) * var_{sim}(\widetilde{h}, h_i) * [freq(\widetilde{h})) * 100/maxFreq] \tag{1}$$

$sim(\widetilde{h}, h_i)$ stands for similarity between $\widetilde{h}$ and the hyponym h_i, $var_{sim}(\widetilde{h}, h_i)$ is the respective similarity variance, $freq(\widetilde{h}))$ is the frequency of the candidate and $maxFreq$ is the given text corpus max frequency (i.e., the most frequent word), and it is used as an upper bound.

3.3.2. Filtering

Finally we adopt two more metrics which help us in the Set Labeling case, and specifically helps with cutting out some false positives:

We define a generic candidate's average similarity as it follows:

$$avgSim(\widetilde{h}, \{S_h\}) = \frac{\sum_{h_i \in \{S_h\}} sim(\widetilde{h}, h_i)}{n} \tag{2}$$

where $\widetilde{h}$ is a candidate hypernym, $\{S_h\}$ is an input set of hyponyms of size $|n|$ and h_i is any element inside of it. This measure tells us how much the candidate is generally similar to the given hyponyms: consider the case in which we got the seed-set {apple, samsung} and want to obtain labels like "corporation". If the model learned that "apple" is a fruit, the discovery will be harder. But then some candidates like "fruit" will be really close to "apple", but way far from "samsung". This is the hint that a given candidate captures a precise meaning in one seed but that it is not linked to the one we are looking for. We can exploit this to exclude the candidate. Then we define a generic hyponyms set's average neighbourhood similarity as it follows:

$$neighAvgSim(\{S_h\}) = \frac{\sum_{i=1}^{n} neighSim(h_i)}{n} \tag{3}$$

with neighSim defined as:

$$neighSim(h_i) = \frac{\sum_{n_i \in NNS(h_i,k)} sim(h_i, n_i)}{k}, \tag{4}$$

where $NNS(h_i, k)$ is the neighbourhood of size k of the hyponym h_i, n_i is a generic neighbour of h_i neighbourhood and S_h is a given hyponyms set of size n. This measure is used to keep trace of how much the neighbours are close to their originating seed, and it is, in fact, used as an upper bound for $avgSim$.

4. Pseudo-Code

In the following, Algorithm 1 shows the high-level pseudo-code of HyperRank. Please note that it refers to Set Labeling—in case we want to retrieve hypernyms for a single hyponym it still can be used but the relevant cycles and metrics are not necessary, as stated in the previous sections.

Algorithm 1: HyperRank: Computing Ranked List of Hypernyms.

```
Data: $\{S_h\}$ (set of hyponyms); Language Model (not mentioned in code); initial NNS width $j$;
      initial neighbourhood intersection rate $k$; number of required hypernyms $n$;
      $neighAvgSim$ threshold $t$
Result: $[L_h]$ (ranked list of hypernyms)
calculate $NeighAvgSim(S_h)$;
for every element $h_i$ of $\{S_h\}$ do
    if $h_i$ is a multi-token concept then
        Calculate mean vector for $h_i$;
    end
end
while maximum NNS width is not reached do
    for every element $h_i$ of $\{S_h\}$ do
        Compute NNS of size $j$ for $h_i$;
    end
    Keep the neighbours which occurs in at least $k$ neighbourhoods $\rightarrow L_{temp}$;
    if $size(L_{temp}) < n$ then
        Increase $j$ and $k$;
    else
        save the candidates list $\rightarrow L_c$;
    end
end
for Any candidate $\widetilde{h}$ in $L_c$ do
    if $avgSim(\widetilde{h}, \{S_h\}) >= t$ then
        Rank the candidate and put it in $L_h$;
    else
        Remove the candidate;
    end
end
Sort $L_h$, cut out to the first $n$ elements and return;
```

5. Evaluation

Hypernym detection takes a test set of pairs in which one of the two elements could be an hypernym; hence it is not suitable for the evaluation of the proposed algorithm, which instead tries to derive the hypernym and works on individual tokens. Hypernym directionality tries to give a direction to the hypernymy relations, that is, to understand which pair element is the hypernym. In this case too, the task does not reflect the logic behind the proposed approach. The SemEval 2018 competition included a task called "Hypernym Discovery" [36] that required returning a list of possible hypernyms for a given hyponym. A ground truth and a scorer for the evaluation were provided. Consequently, the benchmark of SemEval 2018 proved to be the best available tool for evaluating our algorithm.

5.1. Provided Corpus

Although the task remains the same (extraction of a list of hypernym for a given a hyponym), SemEval staff chose to verify on how many different contexts a given system could be considered valid. The input (a plain text corpus) does not change but 5 in total have been provided, as shown in Table 1. The first three corpora contain general-purpose text in various languages:

- **1A**: An English text which is derived from the Stanford WebBase Project (UMBC) [40] and contains 3 billion words.
- **1B**: an Italian text which consists of 1.3 billion words and was obtained from itWac [41].
- **1C**: a Spanish text [42] consisting of 1.8 billion words.

All three corpora were obtained from scraping various websites, hence it is expected to find lots of generic and heterogeneous content. This is supposed to make the hypernym extraction much harder. Two more corpora were provided but with a specific domain:

- **2A**: a medical domain english text, which is parsed from MEDLINE (available at https://www.nlm.nih.gov/databases/download/pubmed_medline.html) and contains scientific articles and abstracts. It contains 180 million words.
- **2B**: an english text containing music-themed articles and biographies, from various sources including Last.fm and the music section of Wikipedia, for a total of 100 million words [43].

Each corpus also comes with a related hypernym vocabulary to reduce the search space. Finally, a list of hyponyms are provided as input, along with a gold standard which was evaluated by experts. SemEval organisers distinguished the hyponyms between "Concept" and "Entities": more specifically an hyponym is defined as a "Concept" when it is a subclass of its hypernym (as "dog" which is a sub-class of "mammal"); an hyponym is an "Entity" if it is a named entity of its hypernym (for example "Rome" is a named entity of "city", but not a subclass). Each system in the competition was able to participate separately for the recognition of Concepts or Entities or both at the same time; we chose to take part to the last variant and with every given corpus. In addition, each task received a 50% split so that the supervised systems could have a training set—given that the proposed system is completely unsupervised these data were ignored. Table 1 is a brief recap of the different testsets, while Table 2 reports the corresponding sizes of Word2Vec models computed from each corpus.

Table 1. Corpora information: here TS stands for Training set, while GS stands for gold standard.

Corpus	N. of Words	Language	Topic	TS Elements	GS Elements
1A	3B	English	General	1500	8548
1B	1.3B	Italian	General	1000	5770
1C	1.8B	Spanish	General	1000	7070
2A	180M	English	Medical	500	4616
2B	100M	English	Music	500	5733

Table 2. W2V models size.

Corpus	Model Size (GB)
1A	7.6
1B	5.5
1C	7.0
2A	1.8
2B	1.2

Table 3 regards the multi-token concepts, a.k.a entities (from the test set and gold standard) that are composed by more than one word. This information is crucial since HyperRank applies a workaround to handle multi-token concepts, and ultimately because it is not able to obtain multi-token hypernyms. These two factors represents a penalization for the approach.

Table 3. Multi-token concepts and hypernyms across the corpora: MT stands for "multi-token" and ST stands for "single-token".

Corpus	ST Concepts	MT Concepts	ST Hypernyms	MT Hypernyms
1A	70%	30%	73%	27%
1B	79%	21%	84%	26%
1C	71%	29%	77%	23%
2A	47%	53%	64%	36%
2B	56%	44%	52%	48%

5.2. Results

We now proceed to show the experimental results. The algorithm parameters value were kept fixed in order to verify that the algorithm can be effective regardless of language or context. We consider Precision at k (1,5,15) to evaluate our HyperRank. The results were obtained directly through the Python script provided by the competition CodaLab (see https://competitions.codalab.org/competitions/17119) page. Given that our approach is unsupervised we point out that we are competing only against those algorithms which did not use any kind of training or supervision. However, for the sake of completeness, and to give a more general idea of our approach performances, all the systems that took part in the task are listed (including the supervised ones). To make the results clearer, an additional improvement column has been added to the table: this is calculated by measuring how much (%) each approach was better or worse (in terms of P@15) than the best unsupervised one. Where HyperRank results as the best unsupervised one we ignore this and still calculate the improvement. In Tables 4–6 shown below the proposed approach is called "HyperRank":

Table 4. Results of evaluation on 1A corpus.

Method	Superv.	P@1	P@5	P@15	Improvement (Over "Team 13")
CRIM r1	✓	29.67	19.03	18.27	+636%
Hybrid of SVD & NN	✓	-	15.00	-	-
MSCG-SANITY r1	✓	19.6	11.60	10.28	+314%
vanillaTaxoEmbed	✓	19.73	9.91	9.26	+273%
HyperRank		6.8	5.99	**9.11**	**+267%**
NLP HZ	✓	12	9.19	8.78	+254%
300-sparsians r1	✓	14.93	8.63	8.13	+227%
MFH	✓	19.8	7.81	7.53	+203%
SJTU BCMI	✓	4.07	5.96	5.78	+133%
Team 13		4.33	2.72	2.48	-
Apollo r2		4.07	2.69	2.42	−3%
apsyn r1		1.4	1.39	1.34	−46%
balapinc r1		1.8	1.30	1.30	−48%
slqs		0.67	0.61	0.61	−75%

It can be seen in Table 4 that HyperRank outperforms all of the unsupervised approaches in terms of P@15. Moreover it beats 50% of the supervised ones. Considering P@5 it still manages to significantly surpass all the unsupervised methods, and to beat the first supervised one. Finally, even considering P@1 the algorithms outperforms the unsupervised ones.

Table 5 shows how HyperRank still significantly outperforms all of the unsupervised algorithms when using the Italian general corpus as well. Overall it is the third most effective approach considering P@15, hence beating the majority of the supervised ones as well. Considering P@5 and P@1, the system is still better than any other unsupervised approach in the competition.

In Table 6 it can be seen that even on the last general purpose corpus (the Spanish one) HyperRank is able to outperform all of the other unsupervised algorithm considering P@15, and beats the last supervised one. Considering both P@5 and P@1, it still reaches the highest score among the unsupervised approaches. During 2A and 2B evaluation there was a change in the nature of the test set; besides the fact that the supplied corpora are much more specific, their size decreases considerably. In particular, by comparison, the W2V model obtained in the corpus 2A is 76% smaller than that obtained on the corpus 1A. Predictably, this could lead to a certain performance drop.

Table 5. Results of evaluation on 1B corpus.

Method	Superv.	P@1	P@5	P@15	Improvement (over "Balapinc")
300-sparsians r1	✓	17.6	11.73	11.20	+212%
NLP_HZ	✓	13.1	11.23	10.90	+204%
HyperRank		7.8	6.29	**10.10**	**+182%**
MFH	✓	17.1	6.82	6.51	+81%
vanillaTaxoEmbed	✓	12.2	6.47	6.00	+67%
Meemi (VecMap)	✓	-	5.95	-	-
balapinc		5.2	3.9	3.58	-
apsyn		3.9	3.56	3.42	−4%
slqs		1.1	1.63	1.67	−54%
Team 13		1	0.48	0.43	−88%

Table 6. Results of evaluation on 1C corpus.

Method	Superv.	P@1	P@5	P@15	Improvement (over "apsyn")
NLP_HZ	✓	21.4	20.39	19.38	+746%
300-sparsians r1	✓	27.8	17.06	16.28	+610%
MFH	✓	24.6	11.26	11.04	+382%
Meemi (VecMap)	✓	-	7.48	-	-
HyperRank		6.7	4.48	**6.39**	**+179%**
vanillaTaxoEmbed	✓	12.2	7.16	5.99	+161%
apsyn		2.3	2.35	2.29	-
balapinc		3.1	2.48	2.23	−3%
Team 13		2.7	1.65	1.30	−44%
slqs		0.3	0.83	1.05	−55%

Even in the case of specific corpus, the algorithm shows good results. In Table 7 it can be seen how HyperRank performed on the 2A corpus. It got a significantly higher score, in terms of P@15, than any other unsupervised approaches. Considering P@5 and P@1 it is the second best unsupervised approach. It must be kept in mind the specific corpora contains the biggest amount of multi-token concepts, for which the algorithm applies a workaround, and most importantly they contains a significantly higher amount of multi-token hypernyms which cannot be obtained by the approach at all. Considering the above the fact that the algorithm still manages to give good results in the unsupervised domain is promising.

The 2B evaluation, shown in Table 8 showed the lowest performances compared to all of the other corpora. Still, HyperRank manages to outperform all the unsupervised approaches (on P@15). We point out that Anu is declared as unsupervised in the challenge paper [36] but we treat it as a supervised one since exploits WordNet [36] as an external knowledge source. One supervised algorithms has been beaten as well. Under P@5 and P@1 the situation remains pretty much the same.

Table 7. Results of evaluation on 2A corpus.

Method	Superv.	P@1	P@5	P@15	Improvement (over ADAPT)
Hybrid of SVD & NN	✓	-	40.19	-	-
CRIM r1	✓	49.2	36.77	27.10	+328%
MFH	✓	32.6	34.2	21.39	+237%
300-sparsians r1	✓	31.6	21.43	17.05	+169%
vanillaTaxoEmbed	✓	35.4	20.71	12.4	+96%
SJTU BCMI	✓	15.2	11.69	10.24	+61%
HyperRank		10.2	7.23	**7.37**	+16%
ADAPT		13.2	8.32	6.33	-
Anu	✓	12.6	7.29	5.57	−13%
Team 13		5.6	2.52	1.83	−71%
balapinc		0.6	1.08	0.9	−76%
apsyn		0.2	0.72	0.68	−79%
slqs		0	0.33	0.4	−94%

Table 8. Results of evaluation on 2B corpus.

Method	Superv.	P@1	P@5	P@15	Improvement (over Team 13)
Hybrid of SVD & NN	✓	-	55.08	-	-
CRIM r2	✓	47.6	41.58	38.54	+1040%
MFH	✓	36.2	35.76	28.44	+741%
300-sparsians r1	✓	33	28.86	27.69	+719%
vanillaTaxoEmbed	✓	33.2	12.41	8.70	+157%
Anu	✓	14.4	8.87	6.8	+101%
HyperRank		10.6	5.40	**5.89**	+74
SJTU BCMI	✓	2.6	5.41	5.09	+50%
Team 13		12.2	4.51	3.38	-
ADAPT		4	1.89	1.35	−60%
balapinc		1.6	1.58	1.24	−63%
apsyn		0.8	1.3	1.13	−67%
slqs		0	0.65	0.84	−76%

6. Considerations Regarding SemEval Evaluation

The results of SemEval look interesting—the algorithm was able to outperform the other unsupervised systems in every tasks, with a considerable gap in terms of both P@5 and P@15. HyperRank showed to be the most effective unsupervised algorithm across all the corpora and domains. In all cases, considering P@15, the algorithm has been able to compete with some supervised approaches and to actually beat them. Considering the absence of structures (tagged text, Hearst pattern), external knowledge (WordNet, WikiData, DBpedia, etc.) or training sets it is promising. The specific-purpose evaluations 2B showed a drop in performance, while still being able to outperform every unsupervised systems. It must be considered that the 2A and 2B corpus are the ones that contains the highest amount of multi-token concepts and multi-token hypernyms, which makes the limits of the algorithm emerge. Moreover, it is possible to notice a certain stability in the performances in both corpora compared to other algorithms like ADAPT, another unsupervised system that participated. While the proposed algorithm loses 33% on P@5 and 25% on P@15 between one corpus and another, ADAPT has a much more pronounced gap, equal to 340% on P@5 and 368% on P@15. The same considerations are valid even for more sophisticated supervised algorithms that shows a significant change when changing corpus or topic, and definitely works better in english language and specific domains. More considerations can be done regarding the generality of our approach—the Set Labeling algorithm has been tested on all of the proposed tasks showing to be the best unsupervised approach in general corpora, and among the best in the specific ones: the other competing approaches have participated only in some tasks. For example CRIM took part in the tasks concerning english language only and ADAPT ran on the specific-purpose ones only, which suggests a certain lack of generality:

this is particularly true for the supervised methods, most of which make use of Hearst patterns making clear a certain language barrier.

Limits of the SemEval Benchmark

Although the evaluation system provided during SemEval 2018 is valid, it has some limits: the first problem that emerges is the lack of completeness in the given gold standard. In this regard, consider the following cases:

- **Buckler** (1A corpus): defined as "a small shield [...] gripped in the fist" has only one label in ground truth, that is "body armor". Not only it is wrong to define a buckler as an "armor" (a shield is normally defined as a defense weapon) but it is the American connotation of the word: if an approach finds "armour" (which is to be expected in an English corpus and found by our algorithm) it is considered wrong. Most importantly "shield" (HyperRank finds as well) is considered wrong too.
- **Pragmatism** (1A): commonly defined as "a philosophical tradition" or "a field of linguistics" and so forth, but the gold standard does not contain any related label at all. Instead we find labels like "social function", "usefulness" or "ceremonial". The proposed algorithm labels it as "philosophy" and it is considered wrong.
- **Liberty** (1A): accepted labels looks more like synonyms than hypernyms (freedom, independence). Even in this case HyperRank labels it as "right" and it is considered as an error.
- **Zinc** (1A): does not contain any label related to the chemical element while HyperRank correctly label it as "mineral" and "metal".

Many other similar cases can be found, highlighting a lack of precision and generality within the ground truth provided for evaluation. It should be kept in mind that, since corpora are provided with a wide variety of subjects and languages, obtaining a very precise gold standard is complex. Still in many cases the most common meanings are omitted. Moreover the task considered hypernym discovery for singletons only; consequently we fall into what was previously defined as "singleton labeling". In this context, variance cannot be used as a ranking metric and it is not possible to evaluate whether it is useful or not; therefore it was necessary to use a different approach to evaluate its effectiveness. Finally, the proposed metrics in SemEval test set do not take into account the algorithm's coverage of any semantic fields: the entities are mixed and have no distinction. Although corpora 2A and 2B focuses on medical and music content, there is no distinction about 1A, 1B and 1C (which are the biggest and most heterogeneous among the available cases). Therefore it is impossible to understand whether an approach or adopted embeddings lacks of effectiveness in some particular fields. Finally it must be kept in mind that any Singleton Labeling (Hypernym Discovery) task can be transformed into a Set Labeling one by augmenting the given hyponym via a Singleton (Set) Expansion [2] algorithm.

7. Custom Testset Evaluation

Motivated by the issues described in the previous section, we created a custom test set to overcome them. This consists of 27 semantic categories, 25 of which contain 10 hyponyms; one of the categories (binary sexes) contains only 2 elements ("male" and "female") and another (continents) contains 8 elements. Some dataset entries can be seen in Table 9. For each category, except for sexual genders and continents, we took 10 random samples (without repetitions), obtaining 259 different sets in total. This procedure was also repeated by progressively expanding the size of the seed, starting with the singleton and up to sets of size 4. The entire procedure was repeated 10 times, and the precision results were then averaged, in order to avoid imprecise evaluation that were inducted by fortuitous cases. All the elements have been lowercased; where possible, a certain overlap between different semantic categories has been included to test the algorithm's ability to capture multiple meanings for the same word (for example Obama is present both as a candidate in the elections and as a president). Given the limitations of the approach, multi-token hypernyms were avoided. Both singular and plural as hypernyms are considered correct since the purpose of the algorithm is to label sets. We trained a Word2Vec model on Wikipedia dump (english) (parsed to plain text with https://github.com/attardi/wikiextractor). The algorithm parameters have been left unchanged. In Figures 1–4 charts show the results of the evaluation on the custom test-sets with different conceptual domains. The improvement related to the seed-set size augmentation are highlighted.

Table 9. Details of the custom testset with some examples.

Field	Hyponyms Example	Hypernyms Example
Colours	Red, Purple, White...	Colour, Coloration, Hue...
Continents	Asia, Africa, Antarctica...	Continent, Mainlands, landmass...
Foods/Ingred.	Bread, Onion, Salt...	Food, Ingredient, foods...
Vehicles	Car, Bycicle, Tank...	Vehicle, Transpor, Vehicles...
Deities	Thor, Allah, Horus...	God, Divinities, Deity...
Ideologies	Marxism, Nazism, Fascism	Ideology, Movement, Policies...
Parties	Radical, Pirate, Civic...	Party, Parties, Faction...
Animals	Lion, Dog, Chameleon...	Animal, Creatures, Exemplary...
Electr. companies	Google, Amazon, DELL...	Companies, Company, Firm...
Fashion Brands	Gucci, Armani, Guess...	Fashion, Brand, Label...
Languages/origins	Greek, Italian, English	Languages, Citizenship, Origin...
Countries	Italy, Germany, Belgium	Country, State, Nation...
CEOs	Zuckerbeg, Bezos, Musk...	CEO, principal, businessman...
Progr. languages	PERL, Ocaml, LISP...	Programming, Language, Languages...
Bin. Sex Genres	Male, Female	Sex, Human, Gender...
USA Candidates	Kaine, Biden, Moore...	Candidate, Politician, Nominee...
USA Presidents	Obama, Nixon, Bush...	Politician, Candidate, Presidents...
Chemicals	Calcium, Chlorine, Silver...	Chemical, Chemicals, Element...
Study Fields	History, Maths, Art...	Subjects, Studies, Field...
Jobs	Plumber, Actor, Teacher...	Job, Occupations, Labor...
Cities	Turin, London, Berlin...	City, Places, Localities...
Capitals	Tokyo, Paris, London...	City, Capital, Places...
Islands	Sardinia, Java, Rhodes...	Island, Islands, Places...
Clothes	Shirt, Trouser, Sock...	Clothes, Clothing, Dresses...
Tech lineups	Surface, Macbook, Galaxy...	Electronics, Lineup, Model...
Fruits	Apple, Pear, Kiwi...	Food, Fruit, Fruits...
Currencies	Dollar, Euro, Franc...	Money, Cash, Currency...

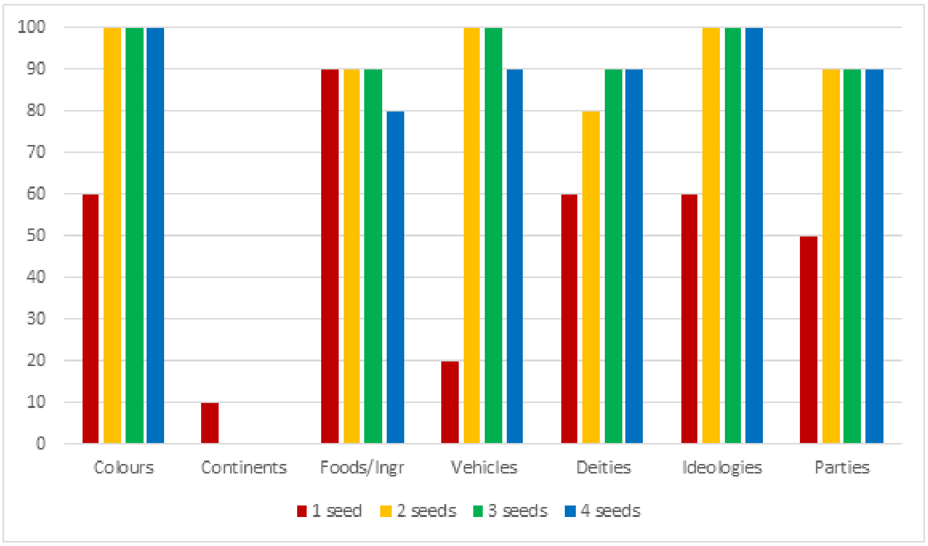

Figure 1. Results on our testset (1/4).

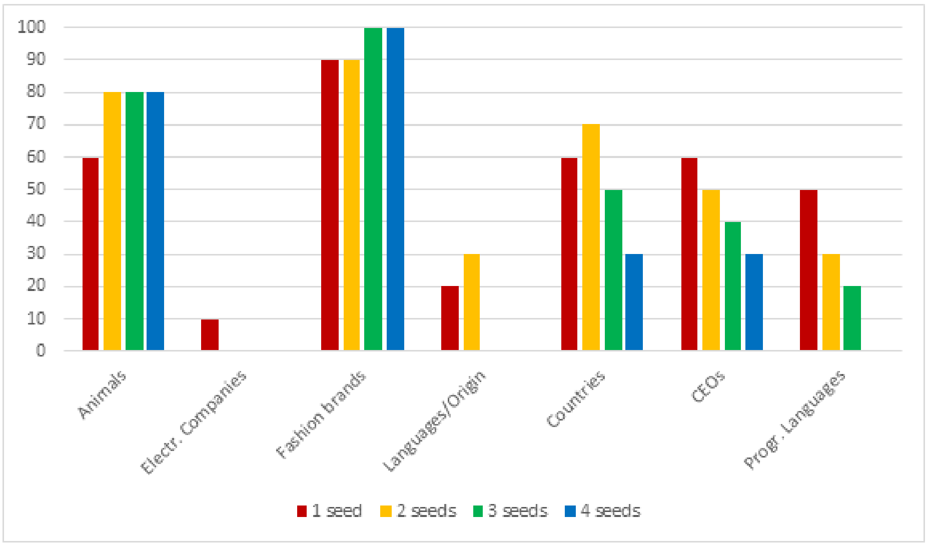

Figure 2. Results on our testset (2/4).

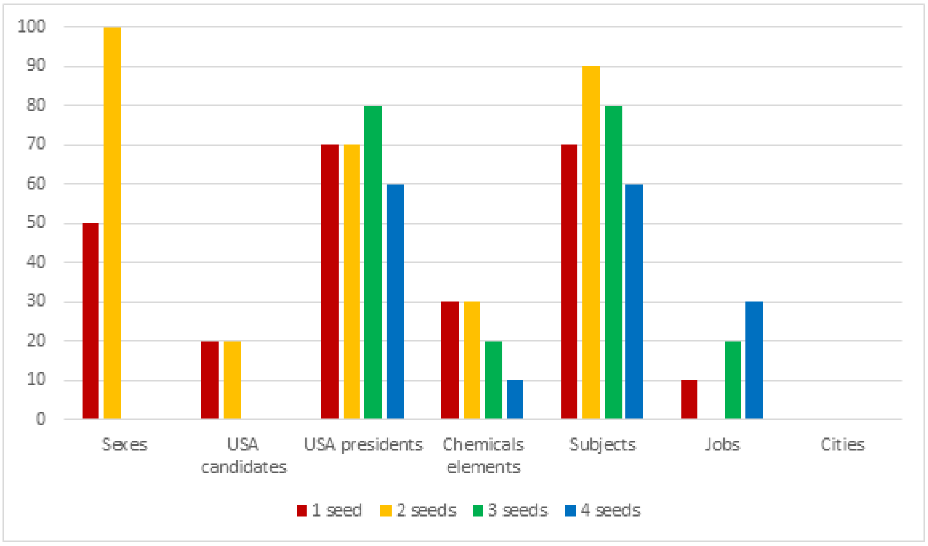

Figure 3. Results on our testset (3/4).

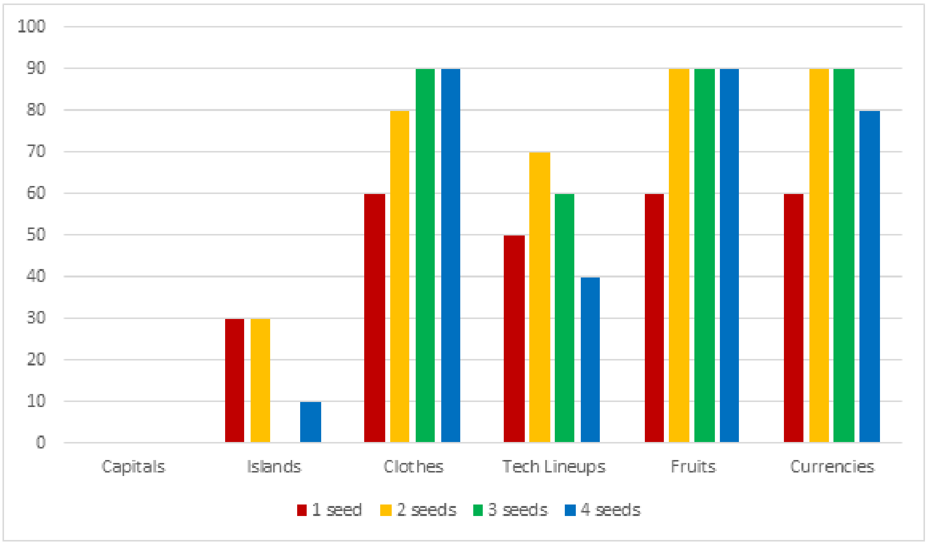

Figure 4. Results on our testset (4/4).

The algorithm's performance on singletons looks interesting, with 48% of the semantic fields being able of reach 60% of successful matches. Some fields looks particularly hard to process, given the massive presence of co-hyponyms or other false positives that cannot be filtered out without degrading the general results: for example trying to label a languages set results in a list of languages as well. A similar problem occurs for the electronic companies like Google, Amazon and so forth, resulting in lists of related words (like "software", "app", "web"...); a particular case is the continents one—the algorithm is not able to correctly label them but keeps returning "countries" in the final list. This tells us that the general geographical context is correctly acquired but cannot be exploited by the algorithm as it is. Moreover it is possible to see how by increasing the size of the starting set much more cases become able to reach a good amount of matched cases. In particular, it is possible to notice a big gap between the semantic fields that can reach ≥80% of successful matches passing from the singleton to the set. This can be seen better by taking a look at Table 10, where for every seed-set size it is showed how many semantic fields, on the total of 27, reach sufficient (60%), good (80%) and full (100%) coverage rate:

Table 10. % of semantic fields which reach sufficient, good or full coverage by seed size.

Seed-Set Size	Sufficient Coverage (≥60%)	Good Coverage (≥80%)	Full Coverage (≥100%)
1	48%	7%	0%
2	**60%**	48%	7%
3	55%	**51%**	7%
4	51%	44%	**11%**

In conclusion, with the introduction of more words into the set (and consequently by exploiting the common neighbours and the informations given by the variance) 60% of the cases (from 48%) are able to reach a sufficient level of matches; up to the 51% of the cases (initially 7%) are able to reach a good level of matches; finally there's a small amount (11%, from the initial 0%) of semantics fields which that achieve full coverage that is the case in which by randomly selecting a certain number of elements, it is always possible to reach a valid hypernym, when using the max allowed set size (4 elements).

8. Conclusions and Future Work

In this work, the theme of hypernymy has been introduced and placed within the context of Natural Language Processing. We proposed HyperRank, an approach for Hypernym Discovery. The approach has innovative features compared to most of those available in the literature such as the fact of being completely unsupervised and language independent, the use of variance as a sub-metric and the fact that it does not exploit any external knowledge. The algorithm has been evaluated on SemEval 2018 hypernym discovery task, proving to be the best unsupervised approach in almost any case. Moreover we created a custom benchmark, designed to verify the effectiveness of our algorithm with sets. Our results show that going from singletons to sets improves effectiveness. We do believe that the contribution provided by our approach can represent a further step towards the efficient management and structuring of plain text and overall shows that unsupervised approaches can show interesting performance and different advantages, namely as working on any given language and exposing an highly maintainable code.

Although the results obtained during the evaluation are promising, there are a number of ways to improve it further. The first and most important issue is the need to properly manage multi-token concepts: this has an important place within linguistics and a lack of efficiency in their management heavily limit the potential of the algorithm. A direct consequence of this point is also the possibility of obtaining multi-token hypernyms. Several tokenizers are available to handle n-grams and they can be used as a base for implementing it. A special mention should be made about the time performance of the algorithm, which is able to label a word in approximately one second. These times could be significantly reduced if the word2vec model used was lighter, for example by filtering out the most frequent words from the provided corpus.

Author Contributions: M.A. and S.B. conceived the approach and wrote the manuscript. All authors have read and approved the final manuscript.

Funding: Supported in part by Sardegna Ricerche project *OKgraph* (CRP 120) and MIUR MIUR PRIN 2017 (2019-2022) project *HOPE—High quality Open data Publishing and Enrichment*.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Atzori, M. The Need of Structured Data: Introducing the OKgraph Project. In Proceedings of the 10th Italian Information Retrieval Workshop, Padova, Italy, 16–18 September 2019.
2. Atzori, M.; Balloccu, B.; Bellanti, A. Unsupervised Singleton Expansion from Free Text. In Proceedings of the 12th IEEE International Conference on Semantic Computing, ICSC 2018, Laguna Hills, CA, USA, 31 January–2 February 2018.
3. Yu, Z.; Wang, H.; Lin, X.; Wang, M. Learning term embeddings for hypernymy identification. In Proceedings of the 24th International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015.
4. Seitner, J.; Bizer, C.; Eckert, K.; Faralli, S.; Meusel, R.; Paulheim, H.; Ponzetto, S.P. A Large DataBase of Hypernymy Relations Extracted from the Web. In Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC'16), Portorož, Slovenia, 23–28 May 2016; pp. 360–367.
5. Kliegr, T. Linked hypernyms: Enriching dbpedia with targeted hypernym discovery. *J. Web Sem.* **2015**, *31*, 59–69. [CrossRef]
6. Wang, C.; He, X.; Zhou, A. A short survey on taxonomy learning from text corpora: Issues, resources and recent advances. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 1190–1203.
7. Hearst, M.A. Automatic Acquisition of Hyponyms from Large Text Corpora. In Proceedings of the 14th Conference on Computational Linguistics (COLING '92), Gothenburg, Sweden, 23–28 August 1992; Association for Computational Linguistics: Stroudsburg, PA, USA, 1992; pp. 539–545.
8. Snow, R.; Jurafsky, D.; Ng, A.Y. Learning syntactic patterns for automatic hypernym discovery. In Proceedings of the Neural Information Processing Systems Conference (NIPS 2004), Vancouver, BC, Canada, 13–18 December 2004; pp. 1297–1304.
9. Nguyen, K.A.; Köper, M.; Walde, S.S.i.; Vu, N.T. Hierarchical embeddings for hypernymy detection and directionality. *arXiv* **2017**, arXiv:1707.07273.
10. Chang, H.S.; Wang, Z.; Vilnis, L.; McCallum, A. Distributional inclusion vector embedding for unsupervised hypernymy detection. *arXiv* **2017**, arXiv:1710.00880.
11. Le, M.; Roller, S.; Papaxanthos, L.; Kiela, D.; Nickel, M. Inferring concept hierarchies from text corpora via hyperbolic embeddings. *arXiv* **2019**, arXiv:1902.00913.
12. Xu, C.; Zhou, Y.; Wang, Q.; Ma, Z.; Zhu, Y. Detecting Hypernymy Relations Between Medical Compound Entities Using a Hybrid-Attention Based Bi-GRU-CapsNet Model. *IEEE Access* **2019**, *7*, 175693–175702. [CrossRef]
13. Shwartz, V.; Goldberg, Y.; Dagan, I. Improving hypernymy detection with an integrated path-based and distributional method. *arXiv* **2016**, arXiv:1603.06076.
14. Ritter, A.; Soderland, S.; Etzioni, O. What Is This, Anyway: Automatic Hypernym Discovery. In Proceedings of the 2009 AAAI Spring Symposium: Learning by Reading and Learning to Read, Stanford, CA, USA, 23–25 March 2009; pp. 88–93.
15. Yates, A.; Cafarella, M.; Banko, M.; Etzioni, O.; Broadhead, M.; Soderland, S. Textrunner: open information extraction on the web. In Proceedings of the Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, Rochester, NY, USA; Association for Computational Linguistics: Strudsburg, PA, USA, 2007; pp. 25–26.
16. Fu, R.; Qin, B.; Liu, T. Exploiting multiple sources for open-domain hypernym discovery. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1224–1234.

17. Yamada, I.; Torisawa, K.; Kazama, J.; Kuroda, K.; Murata, M.; De Saeger, S.; Bond, F.; Sumida, A. Hypernym discovery based on distributional similarity and hierarchical structures. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore, 6–7 August 2009; Association for Computational Linguistics: Strudsburg, PA, USA, 2009; pp. 929–937.
18. Espinosa-Anke, L.; Camacho-Collados, J.; Delli Bovi, C.; Saggion, H. Supervised distributional hypernym discovery via domain adaptation. In Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 424–435.
19. Doval, Y.; Camacho-Collados, J.; Espinosa-Anke, L.; Schockaert, S. Meemi: A Simple Method for Post-processing Cross-lingual Word Embeddings. *arXiv* **2019**, arXiv:1910.07221.
20. Palm Myllylä, J. Domain Adaptation for Hypernym Discovery via Automatic Collection of Domain-Specific Training Data. 2019. Available online: http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1327273&dswid=1297 (accessed on 14 May 2020).
21. Maldonado, A.; Klubička, F. Adapt at semeval-2018 task 9: Skip-gram word embeddings for unsupervised hypernym discovery in specialised corpora. In Proceedings of the 12th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2018; pp. 924–927.
22. Aldine, A.I.A.; Harzallah, M.; Berio, G.; Béchet, N.; Faour, A. EXPR at SemEval-2018 Task 9: A Combined Approach for Hypernym Discovery. In Proceedings of the 12th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2018; pp. 919–923.
23. Hassan, A.Z.; Vallabhajosyula, M.S.; Pedersen, T. UMDuluth-CS8761 at SemEval-2018 Task 9: Hypernym Discovery using Hearst Patterns, Co-occurrence frequencies and Word Embeddings. *arXiv* **2018**, arXiv:1805.10271.
24. Hashimoto, H.; Mori, S. LSTM Language Model for Hypernym Discovery. In Proceedings of the 20th International Conference on Computational Linguistics and Intelligent Text Processing 2019, La Rochelle, France, 7–13 April 2019.
25. Plamada-Onofrei, M.; Hulub, I.; Trandabat, D.; Gîfu, D. Apollo at SemEval-2018 Task 9: Detecting Hypernymy Relations Using Syntactic Dependencies. In Proceedings of the 12th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2018; pp. 898–902.
26. Zhang, Z.; Li, J.; Zhao, H.; Tang, B. Sjtu-nlp at semeval-2018 task 9: Neural hypernym discovery with term embeddings. *arXiv* **2018**, arXiv:1805.10465.
27. Dash, S.; Chowdhury, M.F.M.; Gliozzo, A.; Mihindukulasooriya, N.; Fauceglia, N.R. Hypernym Detection Using Strict Partial Order Networks. *arXiv* **2020**, arXiv:1909.10572.
28. Qiu, W.; Chen, M.; Li, L.; Si, L. NLP_HZ at SemEval-2018 Task 9: A Nearest Neighbor Approach. In Proceedings of the 12th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2018; pp. 909–913.
29. Berend, G.; Makrai, M.; Földiák, P. 300-sparsans at SemEval-2018 Task 9: Hypernymy as interaction of sparse attributes. In Proceedings of the 12th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2018; pp. 928–934.
30. Held, W.; Habash, N. The Effectiveness of Simple Hybrid Systems for Hypernym Discovery. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 3362–3367.
31. Aldine, A.I.A.; Harzallah, M.; Berio, G.; Bechet, N.; Faour, A. Mining Sequential Patterns for Hypernym Relation Extraction. In Proceedings of the TextMine'19, Metz, France, 22 January 2019.
32. Bernier-Colborne, G.; Barriere, C. CRIM at SemEval-2018 task 9: A hybrid approach to hypernym discovery. In Proceedings of the 12th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2018; pp. 725–731.
33. Shi, Y.; Shen, J.; Li, Y.; Zhang, N.; He, X.; Lou, Z.; Zhu, Q.; Walker, M.; Kim, M.; Han, J. Discovering Hypernymy in Text-Rich Heterogeneous Information Network by Exploiting Context Granularity. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing China, 3–7 November 2019; pp. 599–608.
34. Shen, J.; Shen, Z.; Xiong, C.; Wang, C.; Wang, K.; Han, J. TaxoExpan: Self-supervised Taxonomy Expansion with Position-Enhanced Graph Neural Network. In Proceedings of the Web Conference 2020, Taipei, Taiwan, 20–24 April 2020; pp. 486–497.
35. Luo, X.; Liu, L.; Yang, Y.; Bo, L.; Cao, Y.; Wu, J.; Li, Q.; Yang, K.; Zhu, K.Q. AliCoCo: Alibaba E-commerce Cognitive Concept Net. *arXiv* **2020**, arXiv:2003.13230.

36. Camacho-Collados, J.; Delli Bovi, C.; Espinosa-Anke, L.; Oramas, S.; Pasini, T.; Santus, E.; Shwartz, V.; Navigli, R.; Saggion, H. SemEval-2018 task 9: Hypernym discovery. In Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), New Orleans, LA, USA, 5–6 June 2018.
37. Santus, E.; Chiu, T.S.; Lu, Q.; Lenci, A.; Huang, C.R. Unsupervised measure of word similarity: How to outperform co-occurrence and vector cosine in vsms. *arXiv* **2016**, arXiv:1603.09054.
38. Kotlerman, L.; Dagan, I.; Szpektor, I.; Zhitomirsky-Geffet, M. Directional distributional similarity for lexical inference. *Nat. Lang. Eng.* **2010**, *16*, 359–389. [CrossRef]
39. Santus, E.; Lenci, A.; Lu, Q.; Im Walde, S.S. Chasing hypernyms in vector spaces with entropy. In Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, Gothenburg, Sweden, 26–30 April 2014; pp. 38–42.
40. Han, L.; Finin, T. UMBC Webbase Corpus. 2013. Available online: http://ebiq.org/r/351 (accessed on 14 May 2020).
41. Baroni, M.; Bernardini, S.; Ferraresi, A.; Zanchetta, E. The WaCky wide web: A collection of very large linguistically processed web-crawled corpora. *Lang. Resour. Evaluat.* **2009**, *43*, 209–226. [CrossRef]
42. Cardellino, C. Spanish Billion Words Corpus And Embeddings. 2016. Available online: https://crscardellino.github.io/SBWCE/ (accessed on 14 May 2020).
43. Oramas, S.; Espinosa-Anke, L.; Sordo, M.; Saggion, H.; Serra, X. ELMD: An automatically generated entity linking gold standard dataset in the music domain. In Proceedings of the 10th International Conference on Language Resources and Evaluation LREC 2016, Portoroz, Slovenia, 23–28 May 2016; Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., et al., Eds.; European Language Resources Association: Paris, France, 2016; pp. 3312–3317.