Exploring Graph-based Representations for Taxonomy Enrichment

Irina Nikishina[‡], **Natalia Loukachevitch**[†], **Varvara Logacheva**[‡], and **Alexander Panchenko**[‡]

[‡]Skolkovo Institute of Science and Technology, Moscow, Russia
[†]Research Computing Center, Lomonosov Moscow State University, Moscow, Russia
{Irina.Nikishina,A.Panchenko,V.Logacheva}@skoltech.ru
louk_nat@mail.ru

Abstract

The vast majority of the existing approaches for *taxonomy enrichment* apply word embeddings as they have proven to accumulate contexts (in a broad sense) extracted from texts which are sufficient for attaching orphan words to the taxonomy. On the other hand, apart from being large lexical and semantic resources, taxonomies are graph structures. Combining word embeddings with graph structure of taxonomy could be of use for predicting taxonomic relations. In this paper we compare several approaches for attaching new words to the existing taxonomy which are based on the graph representations with the one that relies on fastText embeddings. We test all methods on Russian and English datasets, but they could be also applied to other wordnets and languages.

1 Introduction

Taxonomic structures are often used for the downstream tasks like lexical entailment (Herrera et al., 2005), entity linking (Moro and Navigli, 2015), named entity recognition (Negri and Magnini, 2004). Therefore, they always need to be up-to-date and to keep up with the language change. Moreover, with the rapid growth of lexical resources for specific domains it becomes more and more important to develop systems that could automatically enrich the existing knowledge bases with new words or at least facilitate the manual taxonomy extension process.

In this paper we tackle the taxonomy enrichment task which aims at associating new words (words not present in a taxonomy) with the appropriate hypernym synsets from the taxonomy. For instance, the word "foster-child" should be attached to the hypernym synset "child.n.1" (which refers to "child", "kid", "youngster") from WordNet, and the word "cactus" – to the synset "succulent.n.1". A word may have multiple hypernyms. The task of finding a single suitable synset is difficult for a machine, and a model trained to solve this task will inevitably return many false answers if asked to provide only one synset candidate. On the other hand, if we relax the requirement of uniqueness and ask instead to provide N (for example, 10 or 15) most suitable candidates, this list can contain correct synsets with higher probability. This setting is also suitable for the manual annotation: presenting an annotator with a small list of candidates will facilitate the annotation process, because the annotator will not need to look through all synsets of the taxonomy. Thus, the task is usually formulated as the soft ranking problem, where we need to rank all the synsets according to their suitability for a given word.

While word embeddings demonstrate decent results for predicting hypernyms (Arefyev et al., 2020; Dale, 2020), much less attention is paid to the approaches based on graph representations. We assume that graph-based representations are complementary to the distributional word embeddings, as they capture the hypo-hypernymy relations from graphs. We expect that models using graph representations could be beneficial for the taxonomy enrichment task in combination with distributed word vector representations or on their own. We check our hypothesis on several models which make use of graph structures: node2vec (Grover and Leskovec, 2016), Poincaré embeddings (Nickel and Kiela, 2017) and GCN autoencoder (Kipf and Welling, 2016a) and compare it with an approach of Nikishina et al. (2020b) which applies fastText (Bojanowski et al., 2017) and features from Wiktionary. All in all, our contribution is the exploration of graph-based representation for the taxonomy enrichment task and its combination with the word distributed representations.

2 Related Work

The existing studies on the taxonomies can be divided into three groups. The first one addresses the *Hypernym Discovery* problem (Camacho-Collados et al., 2018): given a word and a text corpus, the task is to identify hypernyms in the text. However, in this task the participants are not given any predefined taxonomy to rely on. The second group of works tackles *Taxonomy Induction* problem (Bordea et al., 2015; Bordea et al., 2016; Velardi et al., 2013), where the goal is to create a taxonomy automatically from scratch. The third group deals with the *Taxonomy Enrichment* task: the participants need to extend a given taxonomy with new words (Jurgens and Pilehvar, 2016; Nikishina et al., 2020a). Both word and graph representations can be applied to any of these tasks.

2.1 Approaches using word vector representations

Approaches using word vector representations are the most popular choice for all tasks related to taxonomies. When solving the *Hypernym Discovery* problem in SemEval-2018 Task 9 (Camacho-Collados et al., 2018) word embeddings are used by most of participants. Bernier-Colborne and Barrière (2018) predict the likelihood of the relationship between an input word and a candidate using word2vec (Mikolov et al., 2013) embeddings. Word2vec is used by Berend et al. (2018) to compute features to train a logistic regression classifier. Maldonado and Klubička (2018) simply consider top-10 closest associates from the Skip-gram word2vec model as hypernym candidates. Pre-trained GloVe embeddings (Pennington et al., 2014) are also used by Shwartz et al. (2016) to initialize embeddings for their LSTM-based Hypernymy Detection model.

Pocostales (2016) also solve the SemEval-2016 Task 13 on taxonomy induction with word embeddings: they compute the vector offset as the average offset of all the pairs generated and exploit it to predict hypernyms for the new data. Afterwards, Aly et al. (2019) apply word2vec embeddings similarity to improve the approaches of the SemEval-2016 Task 13 participants.

The vast majority of participants of SemEval-2016 task 14 (Jurgens and Pilehvar, 2016) and RUSSE'2020 (Nikishina et al., 2020a) also apply word embeddings to find the correct hypernyms in the existing taxonomy. For instance, Tanev and Rotondi (2016) compute a definition vector for the input word by comparing it with the definition vectors of the candidates from a wordnet using cosine similarity. Kunilovskaya et al. (2020) train word2vec embeddings from scratch and cast the task as a classification problem. Arefyev et al. (2020) compare the approach based on XLM-R model (Conneau et al., 2020) with the word2vec "hypernyms of co-hyponyms" method. It considers nearest neighbours as co-hyponyms and takes their hypernyms as candidate synsets.

Summing up, the usage of distributed word vector representations is a simple yet efficient approach to the taxonomy-related tasks and can be considered a strong baseline (Camacho-Collados et al., 2018; Nikishina et al., 2020a).

2.2 Graph-based representations for taxonomies

Graph-based representations for taxonomies have already been tested on other tasks related to the taxonomy enrichment. For instance, node2vec embeddings (Grover and Leskovec, 2016) are used by Liu et al. (2018) for taxonomy induction among other network embeddings.

Another work on Taxonomy Induction which benefits from graphs-based representations is the one by Aly et al. (2019) who achieve state-of-the-art results on all domains. The authors use hyperbolic Poincaré embeddings to enhance automatically created taxonomies. The subtask of reattaching orphan words to the taxonomy is quite similar to taxonomy enrichment. However, the datasets of the SemEval-2016 Task 13 are restricted to specific domains, which leaves an open question of the efficiency of Poincaré embeddings for the general domain and larger datasets. Moreover, Aly et al. (2019) use Hearst Patterns to discover hyponym-hypernym relationships. This technique operates on words, and cannot be transferred to word-synset relations without extra manipulation.

Graph convolutional networks (GCNs) (Kipf and Welling, 2016a) as well as graph autoencoders (Kipf and Welling, 2016b) are mostly applied to the link prediction task on large knowledge bases. Rossi et al. (2020) present an expanded review of the field and compare a wide variety of existing approaches. Graph embeddings are also often used for other taxonomy-related tasks, e.g. entity linking (Pujary et al., 2020). To the best of our knowledge, GCN embeddings have never been used for

enhancing taxonomies like wordnets.

Thus, to the best of our knowledge, our work is the first work on Taxonomy enrichment task which considers wordnets from the prospective of graph structure instead of lexico-semantic resource and makes use of graph-based representations computed from the synsets and hypo-hypernym relations for hypernym prediction.

3 Diachronic WordNet Datasets

For this task we use two diachronic datasets described by Nikishina et al. (2020b): one for English, another one for Russian based respectively on Princeton WordNet (Miller, 1995) and RuWordNet taxonomies. Each dataset consists of a taxonomy and a set of novel words to be added to this resource. The statistics are provided in Table 1.

Dataset	Nouns	Verbs
WordNet1.6 - WordNet3.0	17 043	755
WordNet1.7 - WordNet3.0	6 161	362
WordNet2.0 - WordNet3.0	2 620	193
RuWordNet1.0 - RuWordNet2.0	14 660	2 154
RUSSE'2020	2 288	525

Table 1: Datasets statistics.

3.1 English Dataset

This dataset is created by selecting words which appear in a newer WordNet version, but do not appear in an older one. The words are added to the dataset if only their hypernyms appear in both snippets. Adjectives and adverbs are excluded, as they often introduce abstract concepts and are difficult to interpret by context. Besides, the taxonomies for adjectives and adverbs are worse connected than those for nouns and verbs, thus making the task more difficult.

3.2 Russian Dataset

For the Russian language we test methods on the RUSSE'2020 (Nikishina et al., 2020a) and non-restricted dataset by Nikishina et al. (2020b) which are based on RuWordNet (Loukachevitch et al., 2016), a taxonomy analogous to English WordNet. The RUSSE dataset was filtered from short words (< 4 symbols), diminutives, named entities and other words that can distort the results of the competition. In contrast to this data, the non-restricted dataset did not undergo this preprocessing and contains all new words from RuWordNet2.0.

3.3 Evaluation Metric

The goal of diachronic taxonomy enrichment is to build a newer version of a wordnet given its older version and a list of new terms to be added to the wordnet. We cast this task as a soft ranking problem and use Mean Average Precision (MAP) score for the quality assessment:

$$MAP = \frac{1}{N}\sum_{i=1}^{N} AP_i; \quad AP_i = \frac{1}{M}\sum_{i}^{n} prec_i \times I[y_i = 1], \tag{1}$$

where N and M are the number of predicted and ground truth values, respectively, $prec_i$ is the fraction of ground truth values in the predictions from 1 to i, y_i is the label of the i-th answer in the ranked list of predictions, and I is the indicator function.

This metric is widely used in the Hypernym Discovery shared tasks, where systems are also evaluated over the top candidate hypernyms (Camacho-Collados et al., 2018). Following Nikishina et al. (2020b), we use as gold standard hypernyms not only the immediate hypernyms of each lemma, but also the second-order hypernyms(hypernyms of the hypernyms). Finding the region where a word belongs can already be considered a success. Otherwise, the task of automatically identifying the exact hypernym is too challenging.

The MAP score takes into account the whole range of possible hypernyms and their rank in the candidate list. We use the MAP computation strategy as presented by Nikishina et al. (2020b). It transforms a list of gold standard hypernyms into a list of connectivity components, as new word may have more than one candidate and they could and could not be related directly.

4 Taxonomy Enrichment Methods

We test a number of methods that make use of taxonomy structure to predict hypernyms for the unseen words and compare their performance with the existing approach that is based on fastText embeddings. We describe each method in the corresponding section.

4.1 Word Embeddings with Features Extracted from Wiktionary

We consider approach by Nikishina et al. (2020b) as our baseline. There, a vector representation for a synset in the taxonomy is created by averaging vectors of all words from this synset. Then, for each new word top 10 closest synset vectors are retrieved (we refer to them as *synset associates*). For each of these associates, we extract its immediate hypernyms and hypernyms of all hypernyms (second-order hypernyms). This list of first- and second-order hypernyms forms our candidate set. We rank the candidate set using the following features:

- $n \times sim(v_i, v_{h_j})$, where v_x is a vector representation of a word or a synset x, h_j is a hypernym, n is the number of occurrences of this hypernym in the merged list, $sim(v_i, v_{h_j})$ is the cosine similarity of the vector of the input word i and hypernym vector h_j.
- candidate presence in the Wiktionary hypernyms list for the input word (binary feature),
- candidate presence in the Wiktionary synonyms list (binary feature),
- candidate presence in the Wiktionary definition (binary feature),
- average cosine similarity between the candidate and the Wiktionary hypernyms of the input word.

Finally, feature weights are computed by training a Linear Regression model with L2-regularisation on a training dataset from the previous WordNet/RuWordNet version. Candidate hypernyms are ranked by their model output score and are limited to the $k = 10$ best candidates.

4.2 Candidate Generation Using Poincaré Embeddings

Poincaré embeddings is an approach for "learning hierarchical representations of symbolic data by embedding them into hyperbolic space — or more precisely into an n-dimensional Poincaré ball" (Nickel and Kiela, 2017). Poincare models are trained on hierarchical structures and simultaneously capture hierarchy and similarity due to the underlying hyperbolic geometry. According to the authors, hyperbolic embeddings are more efficient on the hierarchically structured data and may outperform Euclidean embeddings on several tasks, e.g, in Taxonomy Induction (Aly et al., 2019).

Therefore, we use Poincaré embeddings of our wordnets for the taxonomy enrichment task. We train Poincaré ball model for our wordnets using the default parameters and the dimensionality of 10, which yields the best results on the link prediction task (Nickel and Kiela, 2017).

However, applying these embeddings to the task is not straightforward, because Poincaré model's vocabulary is non-extensible. It means that new words that we need to attach to the existing taxonomy will not have any Poincaré embeddings at all and we cannot make use of the embeddings similarity. To overcome this limitation, we compute top-5 fastText nearest synsets (analogously to the procedure described in Section 4.1) and then aggregate embeddings in hyperbolic space using Einstein midpoint following Gülçehre et al. (2019). The resulting vector is considered as an embedding of the input word in the Poincaré space.

Then, we search for the word's top-10 Poincaré nearest neighbours and consider them as candidates. We also try to extend the candidate list with the hypernyms of each Poincaré associate and rank them according to their frequency and similarity to the input word.

4.3 Candidate Generation Using Node2vec Embeddings

The hierarchical structure of the taxonomy is a graph structure, and we may also consider taxonomies as undirected graphs and apply random walk approaches to compute embeddings for the synsets. For this purpose we apply node2vec (Grover and Leskovec, 2016) approach which represents a "random walk of fixed length l" and "two parameters p and q which guide the walk in breadth of in depth". Node2vec randomly samples sequences of nodes and then applies a Skip-gram model to train their vector representations. We train node2vec representations of all synsets in our wordnets with the following parameters: $dimensions = 300$, $walk\_length = 30$, $num\_walks = 200$. The other parameters are taken from the original implementation.

However, analogously to Poincaré vector space, node2vec model has no technique for representing

out-of-vocabulary words. Thus, it is unable to map new words to the vector space. To overcome this limitation, we apply the same technique of averaging top-5 nearest neighbours from fastText and considering their mean vector as the new word embedding and search for the most similar synsets.

We also use an alternative approach to computing out-of-vocabulary node2vec embeddings. Namely, we apply linear transformation from the source fastText to the target node2vec embeddings. For this purpose we train a matrix which is used to project fastText embeddings of the input words to the target node2vec space.

4.4 Link Prediction Using GCN Autoencoder

The models described above have a major shortcoming: the resulting vectors for the input words heavily depend on their representations in fastText model. This can lead to incorrect results if the word's nearest neighbour list is noisy and does not reflect its meaning. In this case the noise will propagate through the Poincaré model and result in inaccurate output even if the Poincaré model is of high quality.

Therefore, we test graph convolutional network architecture (Kipf and Welling, 2016a) that makes use of both fastText embeddings and the graph structure of the taxonomy. In particular, we use graph autoencoder model (Kipf and Welling, 2016b) whose encoder is a graph convolutional network architecture. This model learns vector representations in a completely unsupervised way: it encodes the nodes in the network in a low-dimensional space in such a way that the embeddings can be decoded into a reconstruction of the original network. FastText embeddings are used as input node features. Even though new words are not connected to the taxonomy, it is still possible to compute their embeddings according to their input node features.

For each new node we get its vector representation from the encoder and then predict the probability of the link between the new node and all other nodes in the graph. The top-10 synsets from the existing taxonomy with highest probabilities are considered as final candidates.

4.5 Combining Word and Graph Representations

Additionally, we extend the above model with features based on node2vec and Poincaré embeddings. Namely, we use two extra features: cosine similarity between the candidate and the input word in node2vec vector space and similarity between the candidate and the input word in Poincaré ball model. The overall formula is the following:

$$score_{h_j} = w \cdot m = \sum_{i=1}^{n} w_i m_i \qquad (2)$$

Feature weights from the Logistic Regression model are denoted as vector w, m is the feature vector.

5 Experiments

In this section, we report the performance of our models on the Taxonomy Enrichment task and discuss reasons of low performance of methods exploiting hierarchical structure of the taxonomy.

5.1 Results

We test the models suggested in Section 4 on both English (Table 2) and Russian (Table 3) datasets. It is clearly seen that distributed word vector representations outperform graph-based approaches by a large margin.

Even though Poincaré ball model is designed for the taxonomic structures, the absence of vector representations for the OOV words dramatically affects the results. The aggregated vector of top-5 nearest neighbours retrieved from fastText can often provide a noisy or an overly general representation. Such representation is likely to yield incorrect hypernyms even if the Poincaré embeddings for the taxonomy are of perfect quality.

Likewise, node2vec model also possesses a non-extensible set of embeddings for the taxonomic synsets and uses averaging of fastText associates for representing the new words which negatively affects the results. However, the approach which uses node2vec embeddings and averages top-5 fastText associates is the best-performing approach across methods with graph representations. Moreover, node2vec embeddings perform much better than the Poincaré embeddings. Einstein midpoint aggregation used in our Poincaré-based model makes generalisation of the associate synsets, which results in too abstract synset candidates. On the other hand, averaging node2vec vectors does not have such an effect. The differences between the two models are illustrated by the examples in Table 5.

However, node2vec embeddings still rely on the fastText similarities of the closest embeddings to

method	nouns			verbs		
	1.6-3.0	1.7-3.0	2.0-3.0	1.6-3.0	1.7-3.0	2.0-3.0
Poincaré embeddings	0.0593	0.0658	0.1013	0.1255	0.0656	0.1092
node2vec (top-5 fastText associates)	0.1938	0.2187	0.1554	0.1514	0.1091	0.1469
node2vec (projection)	0.0400	0.0273	0.0218	0.1041	0.0517	0.0377
GCN autoencoder	0.1570	0.1751	0.1677	0.1088	0.0937	0.1173
Nikishina et al. (2020b)	**0.3372**	**0.3800**	**0.3443**	**0.2696**	**0.2002**	0.2366
Nikishina et al. (2020b) + node2vec	0.3130	0.3797	0.3402	0.2591	0.1948	0.1999
Nikishina et al. (2020b) + node2vec + Poincaré	0.3112	0.3498	0.2995	0.2508	0.1770	**0.2482**

Table 2: MAP scores for the taxonomy enrichment methods for the non-restricted English datasets of different WordNet versions.

method	nouns		verbs	
	non-restricted	restricted	non-restricted	restricted
Poincaré embeddings	0.1431	0.2517	0.1050	0.1397
node2vec (top-5 fastText associates)	0.2660	0.3659	0.1681	0.2518
node2vec (projection)	0.1854	0.2527	0.1800	0.2531
GCN autoencoder	0.1826	0.2605	0.0948	0.1406
Nikishina et al. (2020b)	0.4132	0.5515	0.2973	0.3889
Nikishina et al. (2020b) + node2vec	0.4095	0.5575	0.2931	0.3834
Nikishina et al. (2020b) + node2vec + Poincaré	**0.4141**	**0.5587**	0.3056	0.3910
Top-1 for nouns: *Yuriy*	0.3932	0.5522	0.2925	0.4355
Top-1 for nouns: *Yuriy*, no search engine features	0.3692	0.5071	0.2665	0.3888
Top-1 for verbs: Dale (2020)	0.2878	0.4178	**0.3398**	**0.4483**

Table 3: MAP scores for the taxonomy enrichment methods for the Russian datasets non-restricted and restricted (short words, named entities, diminutives excluded) datasets from (Nikishina et al., 2020a)

the input word vector and propagate the fastText inaccuracies. Linear projection which is an alternative option for the computation of node2vec vectors for out-of-vocabulary words, does not solve the problem either. As it can be seen in Table 5, candidates generated using node2vec with the linear projection come from completely irrelevant domains.

GCN autoencoder does not outperform the majority of the approaches for neiter of languages despite being a holistic and self-sufficient approach aimed at combining word representations with the graph structure of taxonomy. The model assigns high probabilities to all synsets in the word's neighbourhood in the graph, whereas only direct and second-order hypernyms are the correct answer. Taxonomic "uncles", "siblings", "cousins", and other distant "relatives" are not welcome.

The combined approach is not very consistent: incorporating graph-based features leads to an increase in scores for the Russian nouns and verbs datasets, whereas for the English dataset the approach does not yield any improvement except for the WordNet 2.0-3.0 dataset. Nevertheless, the combined method performs on par with the best RUSSE'2020 system for nouns track. Despite the close scores, our model can be considered superior to the winner of RUSSE'2020, because it is more stable across languages and easier to replicate.The best RUSSE'2020 approach for nouns extensively uses external tools such as online Machine Translation (MT) and search engines. This approach is difficult to replicate, because its performance for different languages can vary significantly, and we have no means for quantifying this difference.

Francis_Joseph_I emperor.n.01, sovereign.n.01		
Poincaré	node2vec	node2vec projection
person.n.01	king.n.01	fish_genus.n.01
entity.n.01	edward.n.02	genus.n.02
life_form.n.01	herod.n.01	mammal_genus.n.01
causal_agent.n.01	arthur.n.02	city.n.01
worker.n.01	messiah.n.03	municipality.n.01
european.n.01	louis_xiii.n.01	arthropod_genus.n.01
leader.n.01	louis_xiv.n.01	dicot_genus.n.01
object.n.01	frederick_ii.n.01	asterid_dicot_genus.n.01
ruler.n.01	belshazzar.n.01	animal_order.n.01
animal.n.01	pyrrhus.n.01	asterid_dicot_genus.n.01
GCN	fastText	combined (best)
day.n.04	king_of_england.n.01	king_of_england.n.01
metallic_element.n.01	king.n.01	king.n.01
large_integer.n.01	pope.n.01	holy_roman_emperor.n.01
semitic_deity.n.01	islamic_calendar_month.n.01	pope.n.01
hindu_deity.n.01	holy_roman_emperor.n.01	deliberation.n.02
hindu_calendar_month.n.01	general.n.01	islamic_calendar_month.n.01
month.n.02	calendar_month.n.01	**emperor.n.01**
anomalistic_month.n.01	**emperor.n.01**	missionary.n.02
chemical_element.n.01	frank.n.01	frank.n.01
religionist.n.01	jew.n.01	gravida.n.01

Table 4: Prediction noun examples from the English v 1.6-3.0 dataset. Underlined bold text denotes predictions of the model from the ground truth.

5.2 Error Analysis

In order to better understand the difference in systems performance and their main difficulties, we performed quantitative and qualitative analysis of the results on the English nouns subset.

First of all, we wanted to know to what extent the set of correct answers of graph-based models overlaps with the one of fastText-based models. In other words, we would like to know if the graph representations are able to discover hypernymy relations which could not be identified by word embeddings.

Therefore, for each new word we computed average precision (AP) score and compared those scores across different approaches. We found that at least 90% words for which fastText failed to identify correct hypernyms (i.e. words with AP=0) also have the AP of 0 in all the graph-based models. This means that if fastText cannot provide correct hypernyms for a word, other models cannot help either. Moreover, only 8% to 55% words correctly predicted by fastText are also correctly predicted by any of the graph-based models. At the same time, the number of cases where graph-based models perform better than fastText is very low (3–5% cases). Thus, combining them cannot improve the performance significantly. This observation is corroborated by the scores of the combined models.

We list the candidate synsets predicted by different methods in Table 5. They demonstrate the main features of the tested approaches. As we can see, the Poincaré embeddings retrieved by aggregating words from fastText provide too broad concepts which are clearly too far from the correct answers (“object”, “person”, “element”). GCN is too far from the correct answers in general, whereas node2vec results depend on the fastText embeddings and are semantically close to the ground truth synsets.

The candidates provided by fastText model combined with graph-based models features are

overreact react.v.01, act.v.01		
Poincaré	node2vec	node2vec projection
change.v.01	react.v.02	play.v.01
act.v.01	**react.v.01**	compete.v.01
touch.v.01	pursue.v.04	utter.v.02
judge.v.02	**act.v.01**	change.v.01
change_magnitude.v.01	run_down.v.01	shape.v.03
interact.v.01	backfire.v.01	compete.v.01
think.v.03	buck.v.02	adjust.v.01
affect.v.01	marry.v.02	correct.v.01
tell.v.02	answer.v.02	fast.v.02
participate.v.01	wrench.v.01	travel.v.01
GCN	fastText	combined (best)
retaliate.v.02	**act.v.01**	**act.v.01**
exacerbate.v.02	react.v.02	**react.v.01**
cramp.v.01	**react.v.01**	react.v.02
respond.v.03	change.v.01	make.v.01
upset.v.01	affect.v.05	change.v.01
upset.v.06	make.v.01	fear.v.02
dictate.v.02	dramatize.v.02	terrify.v.01
irritate.v.02	misjudge.v.01	take.v.06
hurt.v.04	change_state.v.01	misjudge.v.01
sedate.v.01	right.v.01	burn.v.01

Table 5: Prediction verb examples from the English v 1.6-3.0 dataset. Underlined bold text denotes predictions of the model from the ground truth.

quite similar to those generated by the fastText model without additional features. Therefore, it is reasonable that the difference in scores is minor. However, for some cases (like "emperor.n.01" and "react.v.01" in Table 5) graph vector representations slightly improve the ranking.

6 Conclusion

In this work, we experimented with the graph-based representations for the taxonomy enrichment task and compared them to word vector representations. We tested approaches based on Poincaré and node2vec embeddings along with the approach based on graph autoencoder to predict hypernym synsets for the input word.

Our results show that the use of word vector representations is much more efficient than any of the tested graph-based approaches. Moreover, our baseline method (candidates retrieved from fastText nearest neighbour list and ranked with features extracted from Wiktionary) does not benefit from graph-based methods. Namely, combining the baseline scoring function with Poincaré and node2vec similarities results in marginal improvements for some datasets, but this does not hold for all of them.

According to our experiments, word vector representations are simple, powerful, and extremely effective instrument for taxonomy enrichment, as the contexts (in a broad sense) extracted from the pre-trained fastText embeddings are sufficient to attach new words to the taxonomy.

Error analysis also reveals that the correct synsets identified by graph-based models are usually retrieved by the fastText-based model alone. This makes graphs representations irrelevant and excessive. Nonetheless, there exist cases where graph representations were able to identify correctly some hypernyms which were not captured by fastText.

Despite the discouraging first results of the application of graph-based methods, we suggest that

the taxonomy enrichment task could still benefit from them. In order to improve their performance, we plan to switch from linear transformation to non-linear to project fastText embeddings to node2vec and to apply recently published unsupervised graph word representations GraphGlove (Ryabinin et al., 2020). Moreover, we find it promising to experiment with temporal embeddings such of those of Goel et al. (2020) for the taxonomy enrichment task.

Acknowledgments

The work of Natalia Loukachevitch in the current study (preparation of data for the experiments) is supported by the Russian Science Foundation (project 20-11-20166). We thank Yuriy Nazarov and David Dale for running their approaches from RUSSE'2020 shared task on the English datasets.

References

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 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4811–4817, Florence, Italy, July. Association for Computational Linguistics.

Nikolay Arefyev, Maksim Fedoseev, Andrey Kabanov, and Vadim Zizov. 2020. Word2vec not dead: predicting hypernyms of co-hyponyms is better than reading definitions. In *Computational Linguistics and Intellectual Technologies: papers from the Annual conference "Dialogue"*.

Gábor Berend, Márton Makrai, and Péter Földiák. 2018. 300-sparsans at SemEval-2018 task 9: Hypernymy as interaction of sparse attributes. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 928–934, New Orleans, Louisiana, June. Association for Computational Linguistics.

Gabriel Bernier-Colborne and Caroline Barrière. 2018. CRIM at SemEval-2018 task 9: A hybrid approach to hypernym discovery. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 725–731, New Orleans, Louisiana, June. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Georgeta Bordea, Paul Buitelaar, Stefano Faralli, and Roberto Navigli. 2015. SemEval-2015 task 17: Taxonomy extraction evaluation (TExEval). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 902–910, Denver, Colorado, June. Association for Computational Linguistics.

Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. SemEval-2016 task 13: Taxonomy extraction evaluation (TExEval-2). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1081–1091, San Diego, California, June. Association for Computational Linguistics.

Jose Camacho-Collados, Claudio Delli Bovi, Luis Espinosa-Anke, Sergio Oramas, Tommaso Pasini, Enrico Santus, Vered Shwartz, Roberto Navigli, and Horacio Saggion. 2018. SemEval-2018 task 9: Hypernym discovery. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 712–724, New Orleans, Louisiana, June. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July. Association for Computational Linguistics.

David Dale. 2020. A simple solution for the taxonomy enrichment task: Discovering hypernyms using nearest neighbor search. In *Computational Linguistics and Intellectual Technologies: papers from the Annual conference "Dialogue"*.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3988–3995, Apr.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Çaglar Gülçehre, Misha Denil, Mateusz Malinowski, Ali Razavi, R. Pascanu, K. Hermann, P. Battaglia, V. Bapst, D. Raposo, Adam Santoro, and N. D. Freitas. 2019. Hyperbolic attention networks. *ArXiv*, abs/1805.09786.

Jesús Herrera, Anselmo Penas, and Felisa Verdejo. 2005. Textual entailment recognition based on dependency analysis and wordnet. In *Machine Learning Challenges Workshop*, pages 231–239. Springer.

David Jurgens and Mohammad Taher Pilehvar. 2016. SemEval-2016 task 14: Semantic taxonomy enrichment. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*,

pages 1092–1102, San Diego, California, June. Association for Computational Linguistics.

Thomas N Kipf and Max Welling. 2016a. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Thomas N Kipf and Max Welling. 2016b. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*.

Maria Kunilovskaya, Andrey Kutuzov, and Alister Plum. 2020. Taxonomy enrichment: Linear hyponym-hypernym projection vs synset id classification. In *Computational Linguistics and Intellectual Technologies: papers from the Annual conference "Dialogue"*.

Ninghao Liu, Xiao Huang, Jundong Li, and Xia Hu. 2018. On interpretation of network embedding via taxonomy induction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1812–1820.

Natalia V Loukachevitch, German Lashevich, Anastasia A Gerasimova, Vladimir V Ivanov, and Boris V Dobrov. 2016. Creating russian wordnet by conversion. In *Computational Linguistics and Intellectual Technologies: papers from the Annual conference "Dialogue*, pages 405–415.

Alfredo Maldonado and Filip Klubička. 2018. 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*, pages 924–927, New Orleans, Louisiana, June. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Andrea Moro and Roberto Navigli. 2015. SemEval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297, Denver, Colorado, June. Association for Computational Linguistics.

Matteo Negri and Bernardo Magnini. 2004. Using wordnet predicates for multilingual named entity recognition. In *Proceedings of The Second Global Wordnet Conference*, pages 169–174.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6341–6350. Curran Associates, Inc.

Irina Nikishina, Varvara Logacheva, Alexander Panchenko, and Natalia Loukachevitch. 2020a. RUSSE'2020: Findings of the First Taxonomy Enrichment Task for the Russian Language. In *Computational Linguistics and Intellectual Technologies: papers from the Annual conference "Dialogue"*.

Irina Nikishina, Alexander Panchenko, Varvara Logacheva, and Natalia Loukachevitch. 2020b. Studying taxonomy enrichment on diachronic wordnet versions. In *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain, December. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October. Association for Computational Linguistics.

Joel Pocostales. 2016. NUIG-UNLP at SemEval-2016 task 13: A simple word embedding-based approach for taxonomy extraction. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1298–1302, San Diego, California, June. Association for Computational Linguistics.

Dhruba Pujary, Camilo Thorne, and Wilker Aziz. 2020. Disease normalization with graph embeddings. In *Proceedings of SAI Intelligent Systems Conference*, pages 209–217. Springer.

Andrea Rossi, Donatella Firmani, Antonio Matinata, Paolo Merialdo, and Denilson Barbosa. 2020. Knowledge graph embedding for link prediction: A comparative analysis. *arXiv preprint arXiv:2002.00819*.

Max Ryabinin, Sergei Popov, Liudmila Prokhorenkova, and Elena Voita. 2020. Embedding words in non-vector space with unsupervised graph learning. *arXiv preprint arXiv:2010.02598*.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2389–2398, Berlin, Germany, August. Association for Computational Linguistics.

Hristo Tanev and Agata Rotondi. 2016. Deftor at SemEval-2016 task 14: Taxonomy enrichment using definition vectors. In *Proceedings of the 10th International Workshop on Semantic Evaluation*

(SemEval-2016), pages 1342–1345, San Diego, California, June. Association for Computational Linguistics.

Paola Velardi, Stefano Faralli, and Roberto Navigli. 2013. OntoLearn reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 39(3):665–707.