Domain-specific Taxonomy Enrichment based on Meta-Embeddings

Mikhail Tikhomirov[1][0000−0001−7209−9335] and Natalia V. Loukachevitch[1]

Lomonosov Moscow State University
Moscow, Russia
tikhomirov.mm@gmail.com
louk_nat@mail.ru

Abstract. In this paper we study the use of meta-embeddings approaches, which combine several source embeddings, for the taxonomy class prediction of new terms. We test the proposed approach in the information-security domain in the task of enriching the Ontology on Natural Sciences and technologies (OENT). We show that autoencoder-based meta-embeddings with triplet loss achieve the best results in the task. The highest results are obtained on combination of in-domain and out-of-domain embeddings.

Keywords: Taxonomy · Hypernym prediction· Meta-embeddings

1 Introduction

Ontologies, knowledge graphs in the majority of domains have a taxonomy as a backbone. Relations in taxonomies usually comprise class-subclass relations between concepts, or instance-class relations connecting a specific entity representation and a concept [3,13]: relations of both types can be called IS-A relations or hypernym relations [24]. Development of an ontology in a new domain usually begins from constructing its taxonomy, which determines the ontology scope.

It make it easier to build a taxonomy, various approaches were proposed for extracting hypernym relations for new terms from texts including specific patterns, word co-occurrences, distributional characteristics of words, and others [27]. Currently, the important component of extracting hypernym relations from texts are vector representations (embeddings) of words, which can provide an additional evidence of semantic similarity between words [12,18,28], which is important for identification of hypernym concept for a new term.

Word vectors can be calculated using various text collections and various methods, which means that different vector representations capture the context in different ways, resulting in a wide variety of vector representations for the same words. From here, we can be suppose that some combinations of vectors, so-called meta-embeddings [8], can improve vector representation of words, which allows achieving better prediction of semantic similarity between words or their hypernym concepts.

It was already shown that meta-embeddings improved performance in word analoqy and similarity tasks [41,8,7]. In recent work [38], it was shown that combinations of general word embeddings calculated on large Internet text collections have substantial impact on the performance in taxonomy enrichment of general lexical-semantic resources such as WordNet [24] and RuWordNet [20].

In this paper we show that in the task of extracting taxonomic relations in a specific domain, meta-embeddings combining general (out-of-domain) and in-domain embeddings significantly improve the hypernym concept prediction from the given taxonomy for a new term. We experiment on an information-security text collection, which is used to enrich the Ontology on Natural Sciences and Technologies [11,36] in the information-security domain.

2 Related Work

Traditional methods for hypernym detection include pattern-based methods, searching for specific hypernym patterns in sentences [15,30,31], methods based on similarity of word vector representations [12,18], and also combined approaches integrating various context and similarity features of words [35,4,34].

In 2016, the taxonomy enrichment task was organized as a shared task at SemEval workshop (task 14) [16]. At this task, the participants should to attach words with definitions to correct hypernyms in WordNet [24] using their definitions. In 2020, a new open evaluation on taxonomy enrichment of the Russian wordnet RuWordNet [20] RUSSE'2020 was organized [27]. The task was to find correct hypernyms from an older RuWordNet version for words described in a newer RuWordNet version.

In the RUSSE-2020 evaluation of predicting RuWordNet hypernym synsets for new words [27], the participants used various word embeddings (static – fastText [5], word2vec [21], and contextualized - BERT [10]), the available RuWordNet taxonomy structure, hypernym and co-hyponym patterns, definitions of words from Wiktionary, and global search engines results [2,9,37,27].

Recent methods to hypernym extraction exploit graph-based representations of taxonomy structure. Liu et al. [19] use node2vec embeddings of graph structures [14] for taxonomy induction. Aly et al. [1] use hyperbolic Poincare embeddings [26] for automatic generation of taxonomies. Graph convolutional networks (GCNs) [17] are applied to the link prediction task on large knowledge bases. In [29], the authors study graph-based representation methods on the Diachronic-wordnets dataset, which contains several English and Russian WordNet versions and correct answers of links of words from newer versions to concepts of older versions.

Most current approaches of taxonomy enrichment are based on vector representations of new words and existing concepts in a taxonomy [28]. To improve vector representations, the combined approaches of some source vector representations such as vector concatenation or averaging can be used [8]. In [41] it was shown that using singular value decomposition (SVD) over concatenation of several source vectors can improve the results in several tasks with the ability to control the final vector size.

Autoencoders [7], called Autoencoded Meta-Embeddings (AEME), became a further development of the idea of creating meta-embeddings. In [7], the authors proposed several algorithms (CAEME, AAEME and etc.) for combination various word vectors in one vector by encoding initial vectors in some meta-embedding space and then decoding backward. The CAEME approach tries to reproduce source vectors from the concatenation of encoded representation of these vectors. In the AAEME approach each vector is mapped to a fixed-size vector and all encoded representations are averaged, but not concatenated, which restricts the vector dimension.

In [25] the authors investigated the performance of the autoencoders depending on the loss function (MSE loss, KL-divergence loss, cosine distance loss and also their combinations). They found that there is no evident winner across tasks and that different loss functions should be chosen for different applications.

In [38] the best results for enriching taxonomies of general lexical-semantic resources such as WordNet [23] and RuWordNet [20] were achieved using AAEME encoders with triplet loss, combining fastText, glove and word2vec embeddings in a single meta-representation. The meta-representation was further used for training of a supervised model, which also included features from Wiktionary.

In the current paper we study the performance of meta-embedding approaches in domain-specific taxonomy enrichment: we experiment with assigning new terms from the information-security domain to Ontology in natural sciences and technologies (OENT).

3 OENT Ontology

We study the task of domain-specific taxonomy enrichment using Ontology of Natural Sciences and Technologies (OENT) [11,36]. The OENT ontology [11] is presented as a semantic net of concepts and relations between them, each concept is connected with the set of words and phrases, which can express this concept in documents (text entries). All text entries of the same concepts can be called a synset similar to WordNet synsets [23]. For example, "Mathematical analysis" concept has Russian text entries (synset) as *matematicheskii analis, matanalis, matan.* Synsets in OENT can include different parts of speech: nouns, adjective, verbs, or adverbs.

The OENT ontology is used for automatic document analysis in information-analytical systems, which includes providing of conceptual search, query expansion using the ontology, knowledge-based document categorization etc [36].

OENT comprises large volumes of concepts and terminology from several scientific disciplines and technological domains presented as a connected semantic network of concepts with corresponding text entries and relations between concepts [11]. The ontology was started from extracting terms from specialized text collections (web-sites, school and university text books) in mathematics, physics, geology, biology, and chemistry. Currently, this initial terminology is collected in the OENT subset called OENT-lite.

In further projects, the available conceptual structures were elaborated to more specific levels, also the terminologies of technological domains such as oil-and-gas industry, power energy, education policy and techniques, computer technologies, and information security were added to OENT. The full version of OENT consists of 106K concepts and 308K single and multi-word terms, while the OENT-lite consists of 37K concept and 133K terms.

In the current study we use the OENT-lite ontology and the terminology of the Information security domain to study the enrichment of the ontology with a domain-specific terminology via extracting hypernym concept relations from domain-specific text collections.

4 Taxonomy Enrichment Task

The task of taxonomy enrichment consists in finding an appropriate concept from a given taxonomy for a new word, which can be considered as a hypernym or class for this word. Similar to RUSSE-2020 evaluation, the task is to find a direct (the closest) hypernym concept from the taxonomy [28]. To make the extraction less restricted but to keep it quite precise, the second order hypernym concepts (hypernyms of hypernyms) are also considered as correct answers. In this, we try to simulate the work of knowledge engineers, which should find the most specific concept in a taxonomy to attach a new domain term.

The taxonomy enrichment is treated as a ranking task where the correct answers should be in the top of a candidate list [28]. In contrast to the classification task, the ranking is a more appropriate setting in conditions when the share of correct answers is much smaller that the overall number of candidates.

As a subject area for taxonomy enrichment, the information security domain was chosen, represented by a corpus of 500 thousand texts. For this corpus, a frequency list was built so that each word occurs at least 50 times.

The OENTCyber dataset for evaluation hypernym detection was constructed as follows:

1. All one-word text entries of concepts from the OENT ontology were selected so that they appear in the full version, but are absent in the OENT-lite version;
2. From this list, only words for which the hypernyms are present in the OENT-lite were taken.

As a result, a dataset of 4372 words was obtained, and the task was to predict hypernyms (OENT concepts) for a given set of words using OENT-lite and the available corpus of articles from information-security domain. This dataset contains specific names such as *"chrome"*, *"amazon"*, *"cisco"*, etc and specific terms such as *"css3"*, *"dbscan"*, *"dll"*, etc.

5 Method of Hypernym Concept Prediction

In our approach, we use word embeddings to generate a list of most similar taxonomy entries (words or phrases from the taxonomy) to the target word according to cosine similarity. For each target word, the top 20 taxonomy entries are considered. The number of elements for consideration was chosen experimentally [38]. For each entry in the similarity list, all corresponding concepts, their direct and second-order hypernyms are extracted from the taxonomy. They are considered as candidate concepts to be hypernyms of the target word.

For candidate hypernym concepts, several features are calculated. Logistic regression is used to predict the probability of a candidate to be a hypernym of the target word. The calculated features are as follows:

- the minimum, average, and maximum similarities of the target word to all words of the concept synset;
- the features based on hyponyms of a candidate concept synset:
  - we extract all hyponyms (lower classes) of the candidate concept;
  - for each word/phrase in each hyponym synset we compute their similarity to the target word;
  - we compute the minimum, average, and maximum similarity for each hyponym synset;
  - we form three vectors: a vector of minimums of similarities, average similarities, and maximum similarities of hyponym synsets;
  - for each of these vectors we compute minimum, average, and maximum. We use these resulting 9 numbers as features.
- the minimum, average, and maximum similarity level of the concept in the merged candidate list:
  - the level is 0 if the concept was added based on similarity to the target word;
  - the level of 1 is for the immediate hypernyms of the word in the similarity list;
  - the level of 2 is for the hypernyms of the hypernyms of words in the similarity list.
- the number of occurrences (n) of the concept in the merged candidate list and the quantity $log_2(2 + n)$ which serves for smoothing.

In total, 17 features were calculated. Training data were generated randomly and automatically from the OENT-lite (thus, the data do not contain test data).

We use two models for calculating word embeddings: fastText [6] and word2vec [22]. In order to obtain vectors for words absent in embedding models, the following procedures were carried out:

- For FastText, embeddings if not in the vocabulary, are obtained in a natural way, by calculating vectors by the model itself;
- For Word2Vec, embeddings are calculated by averaging the vectors of maximum prefixes for the constituent words of a multi-word expressions. There is a limitation on the minimum length of a prefix word, which is 4 characters;
- For meta-embeddings, if there is no vector for a word in any model, the corresponding source vector is initialized with zeros.

6 Meta-Embeddings in Taxonomy Enrichment Task

In our work we compare simple meta-embeddings such as concatenation of source embeddings and SVD over the concatenation and two variants of autoencoders generating meta-embeddings: Concatenated Autoencoded Meta-Embeddings (CAEME) and Averaged Autoencoded Meta-Embeddings (AAEME), which have shown good results in previous works [7,38].

Suppose we have two source embeddings $s_1(w)$ and $s_2(w)$, their encoders $E_1(w)$ and $E_2(w)$ and their decoders $D_1(w)$ and $D_2(w)$. Meta-embedding $m(w)$ in CAEME is constructed as the L_2-normalised concatenation of two encoded source embeddings $E_1(s_1(w))$ and $E_2(s_2(w))$:

$$m(w) = \frac{E_1(s_1(w)) \oplus E_2(s_2(w))}{||E_1(s_1(w)) \oplus E_2(s_2(w)||_2}, \tag{1}$$

where $\oplus$ is the concatenation operation.

In CAEME, the dimensionality of the meta-embedding space is the sum of the dimensions of the source embeddings. The AAEME encoder can be seen as a special case of the CAEME encoder, where the meta-embedding is computed by averaging the two encoded sources in (1) instead of their concatenation. Averaging gives the possibility to avoid increasing the dimensionality of the meta-embedding.

The AAEME encoder computes the meta-embedding of a word w from its two source embeddings $s_1(w)$ and $s_2(w)$ as the L_2-normalised sum of two encoded versions of the source embeddings $E_1(s_1(w))$ and $E_2(s_2(w))$:

$$m(w) = \frac{E_1(s_1(w)) + E_2(s_2(w))}{||E_1(s_1(w)) + E_2(s_2(w))||_2}. \tag{2}$$

The CAEME and AAEME decoders reconstruct the source embeddings from the same meta-embedding $m(w)$, thereby implicitly using both common and complementary information in the source embeddings.

The overall objective of autoencoder training is given below. Function f can be any distance or similarity measure as MSE, KL-divergence, or cosine distance.The coefficients λ_1 and λ_2 can be used to give different emphasis to the reconstruction of the two sources.

$$Loss_w(E_1, E_2, D_1, D_2) = \sum_w (\lambda_1 f_{s_1(w),\hat{s}_1(w)} + \lambda_2 f_{s_2(w),\hat{s}_2(w)}), \tag{3}$$

where $\hat{s}_i(w)$ - decoded embeddings corresponding to $s_i(w)$.

Jointly learning of E_1, E_2, D_1, D_2 minimises the total reconstruction error given by Equation 3. To obtain meta-embedding representations after training, only the encoders are applied, which convert the input source embeddings into a meta representation. Further, these meta-embedding vectors are used as vector representations of words.

The standard loss function for AEME approaches we used was cosine distance loss. We have tried variations and combinations between MSE loss, KL divergence loss and cosine distance loss, and last one works best in our case.

We can impose additional restrictions on AEME models during training. One of such restrictions is the use of triplet loss.

The triplet loss function is a loss function for machine learning algorithms where some basic example (anchor) is compared with positive and negative examples. The goal is to minimize the difference in distance between base and positive examples and base and negative examples. In this case, there is often some margin parameter that controls how much the distance to the negative example is greater than to the positive one. One of the first formulations of the triplet loss equivalent approach was introduced in [33] for the metric learning problem. The use of a similar loss function for modifying algorithms has also been used in the problems of image similarity [39], face recognition [32], text classification [40] and other tasks.

We restrict a word to be closer to the words that are semantically related to it according to the taxonomy than to a randomly chosen word with some *margin*:

$$\begin{gathered} L(w_a, w_p, w_n) = max(||m(w_a) - m(w_p))|| - \\ ||m(w_a) - m(w_n))|| + margin, 0), \end{gathered} \tag{4}$$

where $||.||$ is a distance function, w_a is the target word, w_p and w_n are positive and negative words, respectively.

The algorithm of calculating triplet loss is as follows:

1. for each word presented in the taxonomy, we compile a list of semantically related words which includes synonyms, hyponyms and hypernyms;
2. at each epoch, we randomly select K positive words from this related words set and form a set of K negative words by selecting them randomly from the vocabulary;
3. if the word is not presented in the taxonomy, then we cannot form a list of related words for it. In this case, we generate positive vectors for it by adding random noise to its vector;
4. next, we calculate the triplet margin loss by combining the triplet loss with the original loss as $\alpha * loss + (1 - \alpha) * triplet\_loss$.

We use the following parameters for the triplet loss: $K = 5$, $margin = 0.1$, $alpha = 0.005$. These parameters were selected via grid search with AAEME algorithm.

7 Experiments

The quality of the approach was evaluated using two general (external) source vector representations: fastText[1] and word2vec[2]. Also different meta-embedding

[1] Common Crawl Russian versions from https://fastText.cc/docs/en/crawl-vectors.html

[2] Araneum for Russian from http://vectors.nlpl.eu/repository/

approaches were investigated: concatenation, SVD over concatenation, CAEME, AAEME (with and without triplet loss).

In addition to the two "external" vector models word2vec and fastText, two vector models (word2vec and fastText, respectively) were trained on the information security text corpus (hereinafter "internal"). The training parameters were as follows: window = 3, vector size = 300, epochs = 10, method = skip-gram. The performance of models trained on the domain corpus and also their combination with more "powerful" models was investigated, since 500 thousand texts is significantly less compared to text collections on which external word2vec and fastText models were trained.

For evaluation of hypernym prediction, a traditional measure for ranking tasks Mean Average Precision measure is used. This measure achieves the maximal value equal 1 when all correct answers are located in the beginning of a ranking list:

$$\begin{gathered} MAP = \tfrac{1}{N}\textstyle\sum_{i=1}^{N} AP_i; \\ AP_i = \tfrac{1}{M}\textstyle\sum_{i}^{n} prec_i \times I[y_i = 1]. \end{gathered} \tag{5}$$

Another traditional metric for such tasks is Mean Reciprocal Rank (MRR), depending on positions of the first correct answers. This measure is equal to maximal value 1, when all first correct answers are located on 1st positions in ranking lists for all target words.

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i}, \tag{6}$$

where rank is the position of the first relevant item in the ranked list.

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.

In order to evaluate the quality of the approach in such a setting, the described methods of constructing meta-embeddings were used, and each vector model was evaluated separately.

In case of using the AEME approaches for all vector models, it was necessary to determine the individual contributions of each vector model when calculating the loss function. The following weights were obtained experimentally: the weight of 1.0 for the internal models trained on the corpus, the weight of 5.0 for the external word2vec model, and the weight of 2.0 for the external fasttext model. The results can be seen in Table 1 (external models), Table 2 (internal models), and Table 3 (combination of external and internal models).

From Tables 1 and 2, we can see that powerful external models calculated on large text collections still predict hypernym concepts much better than internal, domain-specific models. In both cases all variants of meta-embeddings better predict hypernyms than source vectors. The best results are achieved by encoders with triplet loss. The combination of all models (Table 3) achieves the best results in the hypernym concept prediction. The best prediction results are much higher than the prediction on any of source models. The results achieved by encoders are much better than the results of simple approaches (concatenation and SVD).

method	MAP	MRR
fastText	0.362	0.407
word2vec	0.375	0.421
concat	0.397	0.446
SVD	0.400	0.447
CAEME	0.391	0.439
CAEME triplet	0.398	0.448
AAEME	0.404	0.453
AAEME triplet	**0.412**	**0.464**

Table 1. OENT-lite enrichment: external models

method	MAP	MRR
fastText	0.277	0.317
word2vec	0.277	0.316
concat	0.287	0.327
SVD	0.283	0.324
CAEME	0.286	0.325
CAEME triplet	**0.298**	**0.339**
AAEME	0.280	0.319
AAEME triplet	0.295	0.335

Table 2. OENT-lite enrichment: internal models

7.1 Analysis of Results

We analysed hypernym predictions for new words for which correct predictions were not found in the Top-10 of correct answers and found the following cases:

– Predicted hypernyms correspond to senses missed in the taxonomy. For example, word *"halo"* is described in OENT-lite only in the sense of Russian helicopter, but also this word can mean a computer game. Predicted hypernyms include the concept *"computer program"* at the first position of the candidate list and the *"game"* concept at eighth position.
– A predicted hypernym concept is quite valid and convey another aspect of a target word. For example, for verb *"to cache"*, the correct answer in OENT is

method	MAP	MRR
concat	0.386	0.434
SVD	0.387	0.433
CAEME	0.385	0.434
CAEME triplet	0.408	0.456
AAEME	0.414	0.463
AAEME triplet	**0.427**	**0.479**

Table 3. OENT-lite enrichment: external + internal models

concept *"data storing"*, but the first predicted hypernym concept *"computer technology"* seems also correct;

- Predicted hypernyms may be too general in OENT, but more specific in predictions, for example for word *"amazon"* the correct answers are concepts: *"American company"*, *"company"*, *"foreign company"*. The predicted concepts contain such concept as *"American software company"*, *"American tech company"*. These predicted concepts seem to be more correct;
- In many cases predictions are very semantically close to correct answers but not correct. For example, for word *"CSS3"* the correct answers are concepts *"document markup language"*, *"formal language"*. The predicted hypernym concepts are *"programming language"*, *"scripting language"*, *"object-oriented language"*, *"computer technologies"*;
- there are also numerous examples when too general hypernyms are predicted; in some cases predicted conceptr are very far from reasonable answers and are difficult for explanation.

To show that the higher confidence of the model correlates with better results of hypernym concept prediction, we calculated the plots of dependence of correct answers on the weight of the first prediction. Figure 1 shows the proportion of a correct hypernym concept among first 1, 3, 5, and 10 answers depending on the prediction weight. It can be clearly seen that the higher weight of a predicted hypernym leads to the higher proportion of correct answers. This means that model predictions with high predicted weights and absence of correct answers in the top can be considered as a source of improving hypernym class descriptions in the ontology.

8 Conclusion

In this paper we considered the problem of adapting the OENT ontology to a specific domain of information security: for new words from an information-security text collection, a hypernym concept from the OENT ontology has to be predicted. We investigated methods for combining different word embeddings in a single meta-embedding. The meta-embeddings methods included concatenation of initial embeddings, SVD over the concatenation, two variants of autoencoders aimed to learn better word embeddings from initial vectors.

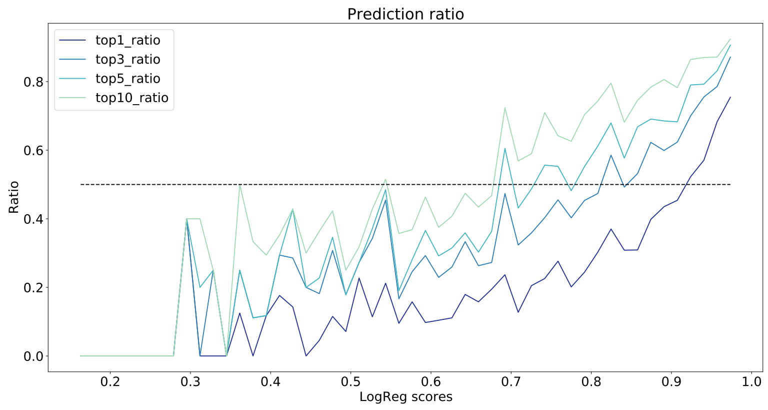

Fig. 1. Ratios between correct and incorrect predictions in: top1, top3, top5, top10 depends on first prediction weight

We showed that the use of meta-embeddings improves the performance of the system for the considered datasets. SVD always improves the results compared to concatenation. Autoencoder-based meta-embeddings achieve the best results in all cases. It can also be seen that adding the triplet loss improves the results significantly.

It has been also shown that the use of vector models trained on specific domain in combination with the meta-embedding approach can improve the quality of hypernym concept prediction. It can also be seen that the quality of the approach on the specific domain is worse then on general domain [38]. We plan to make OENT-lite and the related hypernym dataset publicly available.

Acknowledgements. The participation of M. Tikhomirov in the reported study was funded by RFBR, project number 19-37-90119. 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).

References

1. Aly, R., Acharya, S., Ossa, A., Köhn, A., Biemann, C., Panchenko, A.: 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. pp. 4811–4817. Association for Computational Linguistics, Florence, Italy (2019)
2. Arefyev, N., Fedoseev, M., Kabanov, A., Zizov, V.: 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” (2020)

3. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific american **284**(5), 34–43 (2001)
4. 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. pp. 725–731 (2018)
5. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
6. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
7. Bollegala, D., Bao, C.: Learning word meta-embeddings by autoencoding. In: Proceedings of the 27th international conference on computational linguistics. pp. 1650–1661 (2018)
8. Coates, J., Bollegala, D.: Frustratingly easy meta-embedding–computing meta-embeddings by averaging source word embeddings. arXiv preprint arXiv:1804.05262 (2018)
9. Dale, D.: 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" (2020)
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics (Jun 2019)
11. Dobrov, B.V., Loukachevitch, N.V.: Development of linguistic ontology on natural sciences and technology. In: LREC. pp. 1077–1082. Citeseer (2006)
12. Fu, R., Guo, J., Qin, B., Che, W., Wang, H., Liu, T.: Learning semantic hierarchies via word embeddings. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1199–1209 (2014)
13. Gómez-Pérez, A., Corcho, O.: Ontology languages for the semantic web. IEEE Intelligent systems **17**(1), 54–60 (2002)
14. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)
15. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Coling 1992 volume 2: The 15th international conference on computational linguistics (1992)
16. Jurgens, D., Pilehvar, M.T.: SemEval-2016 task 14: Semantic taxonomy enrichment. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). pp. 1092–1102. Association for Computational Linguistics (Jun 2016)
17. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
18. Levy, O., Remus, S., Biemann, C., Dagan, I.: Do supervised distributional methods really learn lexical inference relations? In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 970–976 (2015)

19. Liu, N., Huang, X., Li, J., Hu, X.: On interpretation of network embedding via taxonomy induction. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1812–1820 (2018)
20. Loukachevitch, N.V., Lashevich, G., Gerasimova, A.A., Ivanov, V.V., Dobrov, B.V.: Creating russian wordnet by conversion. In: Computational Linguistics and Intellectual Technologies: papers from the Annual conference "Dialogue. pp. 405–415 (2016)
21. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26, pp. 3111–3119. Curran Associates, Inc. (2013)
22. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
23. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)
24. Miller, G.A.: WordNet: An electronic lexical database. MIT press (1998)
25. Neill, J.O., Bollegala, D.: Meta-embedding as auxiliary task regularization. arXiv preprint arXiv:1809.05886 (2018)
26. Nickel, M., Kiela, D.: Poincar\'e embeddings for learning hierarchical representations. arXiv preprint arXiv:1705.08039 (2017)
27. Nikishina, I., Logacheva, V., Panchenko, A., Loukachevitch, N.: 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" (2020)
28. Nikishina, I., Panchenko, A., Logacheva, V., Loukachevitch, N.: Studying taxonomy enrichment on diachronic wordnet versions. In: Proceedings of the 28th International Conference on Computational Linguistics. Association for Computational Linguistics, Barcelona, Spain (December 2020)
29. Nikishina, I., Panchenko, A., Logacheva, V., Loukachevitch, N.: Evaluation of taxonomy enrichment on diachronic wordnet versions . In: Proceedings of the 11th Global WordNet conference GWC-2021 (2021)
30. Roller, S., Kiela, D., Nickel, M.: Hearst patterns revisited: Automatic hypernym detection from large text corpora. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 358–363 (2018)
31. Sabirova, K., Lukanin, A.: Automatic extraction of hypernyms and hyponyms from russian texts. In: AIST (Supplement). pp. 35–40 (2014)
32. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 815–823 (2015)
33. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. Advances in neural information processing systems **16**, 41–48 (2004)
34. Shwartz, V., Dagan, I.: Path-based vs. distributional information in recognizing lexical semantic relations. COLING 2016 p. 24 (2016)
35. Snow, R., Jurafsky, D., Ng, A.Y.: Semantic taxonomy induction from heterogenous evidence. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics. pp. 801–808 (2006)

36. Tikhomirov, M., Loukachevitch, N., Dobrov, B.: Methods for assessing theme adherence in student thesis. In: International Conference on Text, Speech, and Dialogue. pp. 69–81. Springer (2019)
37. Tikhomirov, M., LOukachevitch, N., Ekaterina, P.: Combined approach to hypernym detection for thesaurus enrichment. In: Computational Linguistics and Intellectual Technologies: papers from the Annual conference "Dialogue" (2020)
38. Tikhomirov, M., Loukachevitch, N.: Meta-embeddings in taxonomy enrichment task pp. 681–692 (2021)
39. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1386–1393 (2014)
40. Wei, J., Huang, C., Vosoughi, S., Cheng, Y., Xu, S.: Few-shot text classification with triplet networks, data augmentation, and curriculum learning. arXiv preprint arXiv:2103.07552 (2021)
41. Yin, W., Schütze, H.: Learning word meta-embeddings. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1351–1360 (2016)