Cross-modal Contextualized Hidden State Projection Method for Expanding of Taxonomic Graphs

Irina Nikishina[1,2], **Alsu Vakhitova**[2], **Elena Tutubalina**[3,4], **and Alexander Panchenko**[2],
[1]Universität Hamburg, [2]Skolkovo Institute of Science and Technology,
[3]Sber AI, [4]Kazan Federal University
irina.nikishina@uni-hamburg.de, alsu.vakhitova@gmail.com,
tutubalinaev@gmail.com, a.panchenko@skoltech.ru

Abstract

Taxonomy is a graph of terms organized hierarchically using is-a (hypernymy) relations. We suggest novel candidate-free task formulation for the taxonomy enrichment task. To solve the task, we leverage lexical knowledge from the pre-trained models to predict new words missing in the taxonomic resource. We propose a method that combines graph-, and text-based contextualized representations from transformer networks to predict new entries to the taxonomy. We have evaluated the method suggested for this task against text-only baselines based on BERT and fastText representations. The results demonstrate that incorporation of graph embedding is beneficial in the task of hyponym prediction using contextualized models. We hope the new challenging task will foster further research in automatic text graph construction methods.

1 Introduction

In this paper, we focus on taxonomic structures which are quite relevant in many Natural Language Processing (NLP) tasks such as lexical entailment (Herrera et al., 2005) and entity linking (Moro and Navigli, 2015; Sevgili et al., 2022) to represent the relations between products or employees.

Taxonomies are tree-like structures where words are considered as nodes (synsets) and the edges are the relations between them. Such kinds of relationship is called a hypo-hypernym relationship. For instance, let us consider two words: "apple" and "fruit". The former word is *hyponym* ("child") to the latter and the latter is *hypernym* ("parent") to the former.

Many approaches have been proposed to automatically update existing taxonomies (Schlichtkrull and Martínez Alonso, 2016; Arefyev et al., 2020; Nikishina et al., 2020b). However, we argue about one crucial limitation of the existing setups questioning their usefulness in real-world application. In the traditional Taxonomy

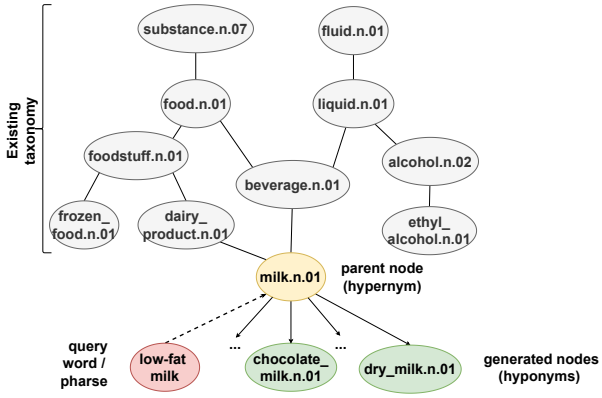

Figure 1: Two types of taxonomy enrichment task: attaching provided candidates (red, prior art) and generating nodes in place without candidates (green, our work).

Enrichment task setting the system is provided with the candidate (orphan) to add and the task is to find the correct place for it in the existing taxonomy. Compiling lists with the new words to add is extremely important but inherently challenging: it might be not clear to which of the multiple sources we would give our preference: neologisms, teenage slang from the Internet or professional jargon.

On the contrary, large pre-trained language models such as BERT (Devlin et al., 2019), ELMo (Peters et al., 2018), GPT (Brown et al., 2020) already contain information about the majority of terms in a language. For instance, many probing studies (Rogers et al., 2020; Jawahar et al., 2019; Ettinger, 2020) show that a vast amount of linguistic information is encoded inside large transformer networks, e.g. syntax or lexical semantics.

In our study, we assume that the huge amount of knowledge from pre-trained models can be leveraged to predict new words missing in taxonomic resources. We suggest a novel candidate-free task formulation for taxonomy enrichment, arguing that compiling word lists may be redundant. Information about new words is already present in the large

Proceedings of TextGraphs-16: Graph-based Methods for Natural Language Processing, pages 11–24
October 16, 2022.

pre-trained networks. There would be not need in compiling lists of "parents" to predict hyponyms either, as language models should be able to predict words only if necessary.

Furthermore, we propose a Cross-modal Contextualized Hidden State Projection Method (CHSP) for candidate-free taxonomy enrichment. The approach includes several stages: (i) learning embeddings of WordNet taxonomy, (ii) projecting them into the hidden states space of BERT, and (iii) decoding them back to text candidates.

Thus, the contribution of our work is three-fold:

- First, we formulate a novel task of candidate-free taxonomy enrichment and present a new dataset based on WordNet 3.0 taxonomy (Miller, 1995);
- Second, we implement baselines for this task based on BERT and fastText (Bojanowski et al., 2017) models, demonstrating the difficulty of the task;
- Third, we propose a method for incorporating graph information into pre-trained language models, based on hidden contextualized state projection yielding superior performance in comparison the baselines.

2 Related Work

There has been two major competitions that have introduced the task of taxonomy enrichment: SemEval 2016 (Jurgens and Pilehvar, 2016) and RUSSE-2020 (Nikishina et al., 2020a). However, their formulations both required a predefined list of candidates. A detailed overview of taxonomy-related papers is presented in Jurgens and Pilehvar (2016); Nikishina et al. (2022).

At the same time there exists a lot of research on how suitable is BERT for capturing and transferring information about hypo-hypernym relationship Ravichander et al. (2020); Hanna and Mareček (2021); Schick and Schütze (2019). For instance, Ravichander et al. (2020) examine hypernymy knowledge encoded in BERT representations. In their experiments BERT demonstrated the ability to correctly retrieve hypernyms, however, they argue that it does not necessarily follow that BERT is capable of systematic generalisation.

Another paper about BERT's knowledge of hypernymy (Hanna and Mareček, 2021) applies several patterns to predict possible hypernym candidates: "[MASK], such as x" and "My favorite [MASK] is x". Such prompts often elicit correct hypernyms from BERT. However, BERT still fails in 43% of cases, therefore, the authors claim that BERT has limited understanding of hypernymy. There exist many more Hearst patterns (Hearst, 1992) that aim to identify hypo-hypernym relationship in unlabeled texts (Snow et al., 2006; Pantel and Pennacchiotti, 2006). We compare baselines with some of them in Section 6.

Anwar et al. (2020) examine the influence of context-aware word representation models for lexical units and frame role expansion task. This task is related to our setting in a sense of generation of meaningful substitutes with preservation of content. We adopt their context-aware methods for our task. In our case the meaningful substitute will be generated for a masked hyponym with preservation of meaning represented in projected embeddings (see Section 4).

3 Taxonomy Enrichment Task

We formulate taxonomy enrichment in a new way avoiding the need of pre-supplied candidates (cf. Fig. 1) making it more challenging yet realistic. Given a taxonomy $T = \{h, r, t\} \subseteq E \times R \times E$, the task is to predict new nodes $n \in N, N \not\subseteq E$, which are not yet included in the taxonomy T, starting from the current node $h_i \in E$.

3.1 Dataset

We provide subgraphs sampled from the existing taxonomy as input to predict hyponyms at a certain place (see Fig. 1 as the example). In this research, we perform experiments on WordNet 3.0 (Miller, 1995) nouns (82,115 synsets, 117,798 lemmas). We suggest using synsets 2 hops away from the target node, as further located synsets may not be semantically related.

From this taxonomy we randomly select 1,000 nodes out of 15,646 nodes which children are leaves, i.e., the children do not have hyponyms of their own. We also take into consideration the distance length from the root to the leaf which should be more than 5 hops. This allows us to exclude the case of predicting very abstract or broad concepts. For each "parental" hypernym all its hyponyms (leaves) were replaced by a single "masked" node, e.g., *handwear.n.01* had hyponyms *glove.n.02* and *muff.n.01* that were replaced by a single *ORPHAN_100000243*. This place in the taxonomy was then considered for extension and the

candidates predicted for the masked node could be compared against true hyponyms. All in all, we masked 4,376 leaves out of 65,422 noun leaves to 1000 "[MASK]" tokens.

We limit our experiments to leaves only, replacing all children with one mask in order to be able to compare with a wide range of possible answers, as one synset might have several hyponyms. We leave node injection to future work on the topic.

3.2 Evaluation metrics

The generated candidates will be compared against the true candidates from the existing taxonomy. We utilize Precision@k (P@k), Recall (R@k), and Mean Reciprocal Rank (MRR): $Precision@k = \frac{\text{relevant items @k}}{\text{recommended items @k}}$, where k is the number of candidates at each step; $MRR = \frac{1}{|Q|}\sum_{i}^{|Q|}\frac{1}{rank_i}$, where Q is the sample of queries, $rank_i$ is the first position of the relevant candidate in the ranked list for the query i. Intuitively, MRR looks how close to the top of the list the correct answer is. Both metrics are commonly employed in the Hypernym Discovery and Taxonomy Enrichment shared tasks, which require systems to produce ranked lists of potential hypernyms (Camacho-Collados et al., 2018; Dale, 2020). Furthermore, numbers for both metrics are multiplied by 100 for clearer presentation.

4 Cross-modal Contextualized Hidden State Projection Method

The main idea of the paper is to predict new words using knowledge preserved in BERT and enhance the word generation process with graph information. Fig. 2 demonstrates the overall architecture of the CHSP approach that we use to solve the task. First, we train a graph representation model to compute graph embeddings. Furthermore, we learn a projection layer to transform target graph embeddings to the BERT vector space. Then we apply the projected embeddings as input to the masked language modelling part of BERT model. The prediction head generates new lemmas that are treated as candidate hyponyms for parent nodes. This process results in gradual joining of graph and textual modalities.

4.1 Graph Embedding Computation

In this section, we study various graph embedding representations to integrate into BERT. In Fig. 2, it is the Graph-BERT model that is depicted, however, it could be any model for representing graph structure. We evaluated several inductive and non-inductive embeddings such as Graph-BERT (Zhang et al., 2020), node2vec (Grover and Leskovec, 2016), GCN (Kipf and Welling, 2016), GAT (Velickovic et al., 2018), TADW (Yang et al., 2015), and Poincaré (Nickel and Kiela, 2017) embeddings. We also tested directed and undirected structures of Graph-BERT, node2vec and Poincaré. We performed both intrinsic and extrinsic evaluation of the computed embeddings.

As for the intrinsic evaluation, which was conducted on the unmasked WordNet, we generated the top-10 nearest neighbours and computed Precision@k and Recall@k scores (k=1, 2, 5, 10) metrics that assess the amount of hyponyms presented in the top-k list. We assume that the more "children" are presented in the list, the more suitable embeddings are for the tree-like structures and hyponym prediction. From Table 1 we can see that the best inductive embedding model is Graph-BERT on the directed graph and non-inductive node2vec on the undirected graph. We observe that node2vec and Poincaré show much higher scores than other methods. We speculate that this can be explained by the fact that these two algorithms are the only ones that do not incorporate textual features into the learned embeddings. Intuitively, similarity in textual features is not equal to similarity in graph. Additionally, degradation of node similarity in models that aggregate information from graph structure and node features is a known issue (Jin et al., 2021) and is linked to the over-smoothing problem. We believe that this could be one of the reasons why the approaches, which demonstrate promising results on traditional taxonomy enrichment task (Nikishina et al., 2022), like GAT, GCN, TADW do not perform well on predicting nearest neighbours. Moreover, we hypothesize that it also might be explained by the fact that such models better represent co-hyponymy or hypernymy, rather than hyponymy. Graph-BERT is known for avoiding over-smoothing problem, thus, performs much better than GAT, GCN and TADW.

For the extrinsic evaluation (evaluation of the downstream task) we have used two models: the best non-inductive and the best inductive embeddings. It is either a Graph-BERT (Zhang et al., 2020) that accepts a sequence of node representations and their positional embeddings describing their local and global positioning in the graph, or a node2Vec (Grover and Leskovec, 2016) that

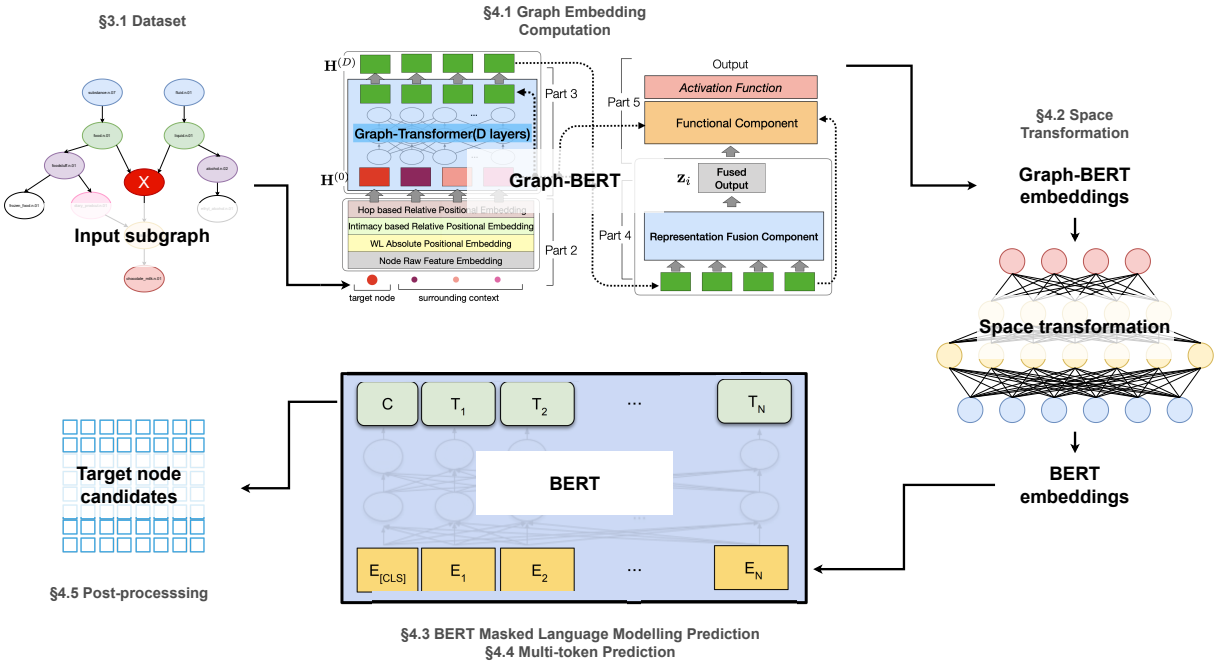

Figure 2: Cross-modal Contextualized Hidden State Projection Method (CHSP): graph-based BERT architecture that makes use of both node and text embeddings. Graph-BERT illustration source: (Zhang et al., 2020), BERT illustration source (Devlin et al., 2019). The input data is described in §3.1. §4.1 describes the choice of graph embedding algorithm. §4.2 explains the projection of embeddings from graph space to BERT space. § explains how BERT was used to predict candidates from the projected embeddings. §4.4 explains of the multi-token candidate generation algorithm. Finally §4.5 lists post-processing filters applied on the list of generated candidates.

Table 1: Graph embeddings comparison on the tree representation task.

	Embeddings	P@1	P@2	P@5	P@10	R@1	R@2	R@5	R@10
Inductive	Graph-BERT directed (node reconstruction)	0.127	0.099	0.064	0.041	0.127	0.113	0.150	0.182
	GraphBERT directed (graph recovery)	**0.190**	**0.163**	**0.115**	**0.073**	**0.190**	**0.182**	**0.260**	0.314
	Graph-BERT undirected (node reconstruction)	0.166	0.142	0.107	0.070	0.160	0.166	0.273	0.349
	Graph-BERT undirected (graph recovery)	0.164	0.140	0.100	0.062	0.164	0.153	0.227	0.268
	GCN	0.021	0.024	0.028	0.030	0.021	0.033	0.073	0.137
	GAT	0.018	0.016	0.014	0.011	0.008	0.021	0.068	0.099
Non-inductive	Node2vec directed root2leaf	0.227	0.217	0.212	0.181	0.227	0.241	0.368	0.509
	Node2vec directed leaf2root	0.451	0.359	0.244	0.173	0.451	0.470	0.563	0.674
	Node2vec undirected	**0.988**	**0.807**	**0.515**	**0.321**	**0.988**	**0.987**	**0.988**	**0.990**
	Poincare directed	0.769	0.671	0.464	0.297	0.769	0.818	0.882	0.910
	Poincare undirected	0.716	0.618	0.434	0.283	0.716	0.727	0.804	0.862
	TADW	0.006	0.005	0.005	0.004	0.006	0.006	0.008	0.010

learns low-dimensional representations for nodes in a graph through the use of random walks. However, as we will further see, good coverage of hyponyms in the nearest neighbour list does not guarantee high performance on hyponym prediction.

4.2 Space Transformation

In order to project graph embeddings into BERT embedding space, we use a simple multilayer perceptron (MLP). The architecture and training process are described in Appendix A.2.

BERT embeddings are contextualized. Therefore, for learning projection from graph space into BERT, the target words cannot be simply embedded as is because their representation will differ in various contexts. In order to generate contextualized embeddings we use a SemCor dataset (Langone et al., 2004). It consists of 352 texts from Brown Corpus (Kucera and Francis, 1967), which is an electronic collection of text samples in English language. SemCor contains manually annotated sentences where words are matched with according synsets. We adopt SemCor 3.0, which was automatically created from SemCor 1.6 by mapping senses

from WordNet 1.6 to WordNet 3.0. We extract embeddings of annotated words and use as contextualized target synset embeddings for learning projection.

4.3 BERT Masked Language Modelling Prediction

We use *bert_base_uncased* pre-trained configuration of BERT to embed a structure "[MASK] is a {parent}" where "{parent}" is a lemma of a hypernym whose hyponyms are to be predicted. In the following parts we will refer to this structure as input context. The choice of the structure was not random. To begin with, we have evaluated three different context constructions suggested in (Hanna and Mareček, 2021): 1. "[MASK] is a/an {parent}"; 2. "My favourite {parent} is a [MASK]"; 3. "{parent} such as a [MASK]" . The scores for the amount of true hyponyms in a list of predicted candidates are presented in the first three lines of Table 2 and Table 3, accordingly. The Precision@10 scores indicate that the best results were produced by the first prompt, which proved to be the most stable among the three, and it was used in all CHSP configurations. These experiments are also repurposed as three baselines.

Furthermore, we create three settings with different approaches to incorporation of graph embedding into the language model prediction:

- *pure-BERT* prediction: embedding of "[MASK]" token is left as is;
- *replaced* prediction: embedding of "[MASK]" token is replaced by projected graph embedding;
- *mixed* (or contextualized) prediction: embedding of "[MASK]" token is averaged with projected graph embedding.

The replacement can happen at three different stages: after first layer of BERT encoder, after sixth (middle) or after twelfth (last). In the first two cases space transformation learns to project graph embeddings into intermediate hidden states and after replacement the hidden states are passed through remaining encoder layers. The replacement strategies are illustrated in Fig. 3. Thus, by performing this process, we combine textual and graph modalities in order to improve candidate prediction at the certain place of the taxonomy.

4.4 Multi-token Prediction

For the experiments with single- and multi-token prediction we adopt a condBERT (Dementieva et al., 2021) multi-token generation mechanism. In addition to "[MASK] is a {parent}", "[MASK][MASK] is a {parent}" or "[MASK][MASK][MASK] is a {parent}" sentences are used. The tokens are generated progressively using beam search while each multi-token sequence is scored by the harmonic mean of the probabilities of its tokens. The beam search process is illustrated in Fig. 4. The algorithm generates 1-, 2- and 3-token predictions, which are merged into a final candidates list sorted according to their scores. The detailed description of the multi-token candidate generation algorithm is given in the Appendix A.3.

4.5 Post-processing

In order to eliminate noise from the predictions generated by the BERT language model, we apply several filters on the generated set of new words. First, we remove all predictions containing non-alphabetical symbols as well as stop-words from Stopwords Corpus (Porter, 1980) in NLTK library[1]. The multi-token generation case requires further post-processing: merging word-pieces and discarding candidates where all tokens start with "##".

Furthermore, we check merged candidates for containing permutations of same sets of words and eliminate the repeating ones with lower scores. For example, if there are two multi-token candidates "apple pie" and "pie apple", the one less-probable one is going to be discarded. Finally, the whole list of merged candidates is checked for duplicates and sorted by their scores.

5 Baselines

In our experiments we are using three baselines: 1. fastText (nearest neighbours); 2. BERT (parent embeddings on inference); 3. three patterns from (Hanna and Mareček, 2021; Schick and Schütze, 2019) .

5.1 fastText (nearest neighbours)

The first baseline uses 300-dimensional fastText (Bojanowski et al., 2017) English embeddings pre-trained on Common Crawl and Wikipedia. Hypernym embeddings are computed as an average of all lemmas embeddings. Furthermore, nearest

[1] https://www.nltk.org/

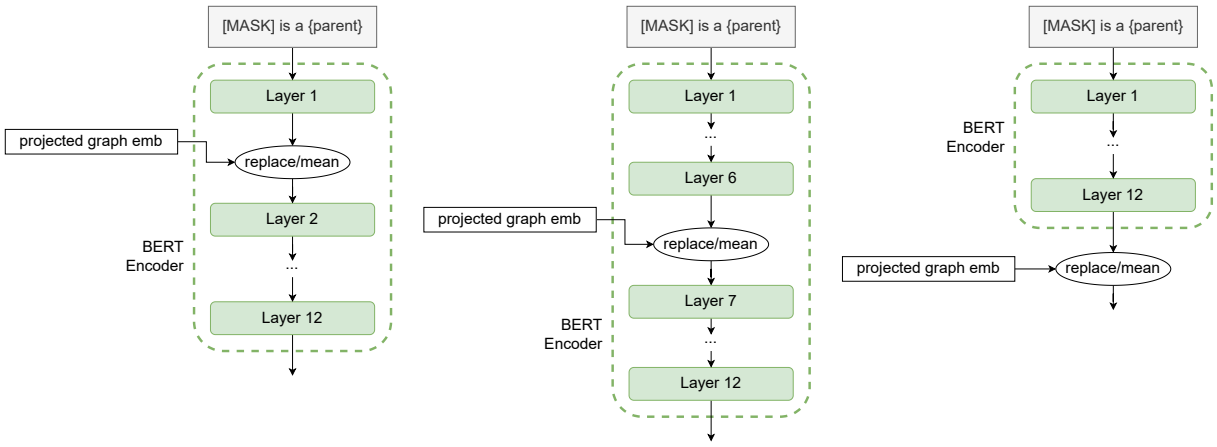

Figure 3: Illustration of replacement approaches. The projected graph embedding is inserted after (a) 1st BERT encoder layer, (b) 6th BERT encoder layer, (c) 12th BERT encoder layer. The "replace/mean" denote the replacement strategy: the projected embedding either replaces according hidden representation of "[MASK]" token, or averaged with it.

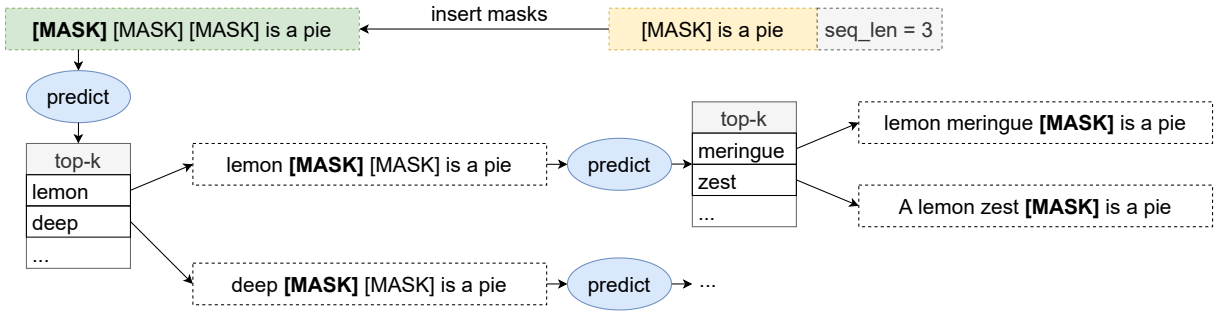

Figure 4: Beam search for multi-token generation. In this figure 3-token case is illustrated. In our research we also use 2-token case which is generated in a similar manner.

neighbours of the resulting vectors are retrieved and scored as hyponym predictions. Our approach can be seen as a reverse of the method from (Nikishina et al., 2020a). In a single-token evaluation case multi-token hyponyms are dropped from the list of gold hyponyms (see Section 3.2).

5.2 BERT (parent embeddings on inference)

The second baseline uses BERT to encode each hypernym lemma and decode it back in a single- or multi-token setting. Predictions for each parent lemma are aggregated and evaluated. This method is loosely motivated by the idea of lexical substitution (Anwar et al., 2020), which goal is to find meaning-preserving alternatives to a particular target word in its context. However, with this baseline we wanted to evaluate BERT's ability to predict hyponyms in a contextless setting.

5.3 Pattern Comparison

The last baseline is based on the approach described in these two publications: (Hanna and Mareček, 2021; Schick and Schütze, 2019). They propose a variety of constructions for prompting BERT in order to identify its linguistic capabilities and test its ability to capture semantic properties of words. Both works use the similar set of constructions, however, only (Hanna and Mareček, 2021) compare them against each other in order to identify the most efficient ones. According to their evaluations we have selected three best patterns: "[MASK] is a/an {parent}", "My favourite {parent} is a [MASK]", "{parent} such as a [MASK]". The constructions were encoded with BERT and then decoded in single- and multi-token settings with "[MASK]" predictions treated as new candidate hyponyms.

Table 2: Prediction scores for single-token hyponyms generation for different source graph embeddings and replacement strategies (x100).

Method	Context	Replaced	MRR@5	MRR@10	MRR@20	P@1	P@5	P@10
Pattern comparison (Hanna and Mareček, 2021)								
"[MASK] is a {parent}"	Yes	No	2.461	2.704	3.091	1.546	1.289	1.057
"My favourite {parent} is a [MASK]"	Yes	No	0.554	0.863	1.001	0.000	0.464	0.490
"A {parent} such as a [MASK]"	Yes	No	0.168	0.193	0.235	0.000	0.155	0.103
BERT (parent embedding on inference)	No	No	1.003	1.083	1.203	0.940	0.251	0.188
fastText (nearest neighbours)	No	No	2.400	3.500	4.000	0.130	1.839	2.100
CHSP (Graph-BERT)	Yes	Mix	**7.229**	**8.037**	**8.624**	**3.608**	**3.247**	**2.474**

Table 3: Prediction scores for multi-token hyponyms generation for different source graph embeddings and replacement strategies (x100).

Method	Context	Replaced	MRR@5	MRR@10	MRR@20	P@1	P@5	P@10
Pattern comparison (Hanna and Mareček, 2021)								
"[MASK] is a {parent}"	Yes	No	0.930	1.027	1.177	0.600	0.460	0.370
"My favourite {parent} is a [MASK]"	Yes	No	0.425	0.693	0.844	0.000	0.361	0.438
"A {parent} such as a [MASK]"	Yes	No	0.051	0.137	0.137	0.000	0.052	0.077
BERT (parent embedding on inference)	No	No	0.320	0.345	0.390	0.300	0.080	0.060
fastText (nearest neighbours)	No	-	1.860	**2.673**	**3.069**	0.100	**1.420**	**1.620**
CHSP (Graph-BERT)	Yes	Yes	**2.150**	2.281	2.378	**1.600**	0.740	0.530

6 Experiments

Our experiments can be categorised by following features: source graph embeddings, usage of context structure, replacement layer and replacement strategy. This section is divided into two parts. The first subsection compares various combinations of CHSP configurations. The second subsection analyzes performance of the best CHSP configurations against the baselines.

6.1 Graph Embeddings Comparison

Tables 5 and 6 compare single-token and multi-token hyponym predictions for methods with different source embeddings, replacement strategies and replacement layers. We observe that in single-token case for both node2vec and Graph-BERT the best replacement point is after the last (12th) BERT encoder layer with first and sixth being close seconds. We hypothesise that the reason is that, when injecting the projected graph embedding at earlier stages, remaining encoder layers dilute information incorporated in the embedding, thus deflecting from the right answers. In the case of single-token generation, Graph-BERT with the replacement point is after the last layer is a clear winning strategy among all the combinations. On the contrary, for multi-token generation significantly better scores were obtained by replacement after 6th layer. We suggest that this replacement strategy helped to diversify generated subwords and produce more meaningful results.

In general, "mixing" replacement strategy produces better results for the last-layer replacement strategy, because it allows incorporation of a context information encoded in a final hidden state of "[MASK]" token. However, there are some cases when the context actually diverts the method from the real answer (see Section 7). The complete replacement showed better scores in 1st and 6th layer replacement, because this strategy already incorporates a lot of context in the "[MASK]" embedding while passing it through remaining layers of the encoder, and "mixing" replacement reduces the influence of projected embedding too much. To sum up, both replacement strategies are important and none can be deemed winning as there is a clear pattern of where to apply each of them.

We can observe that node2vec did not perform as well as was expected judging from the graph embedding comparison. In many cases of single-token generation, words synonymous to the hypernym were predicted, instead of hyponyms. The reason for the low scores on node2vec embeddings might be explained by the fact that the Graph-BERT embeddings are easier to transform to the BERT vector space. Another hypothesis is that the performance on hyponym prediction does not guarantee high

scores on predicting hyponyms for the taxonomy enrichment.

6.2 Overall Comparison

Tables 2 and 3 contain the overall scores for different hyponym prediction methods. We can see that our approach significantly outperforms other methods on single token setup, however, it fails on predicting multi-token candidates. We observe that the patterns from (Hanna and Mareček, 2021; Schick and Schütze, 2019) show results are mostly far from the top ones. This happened because the context encapsulated in the patterns in general contains little information. We also see that our method outperforms the BERT (parent embedding on inference) baseline (which is a simple prediction of encoded parent synset) and a simple approach on fastText nearest neighbours candidates. Even though the results for multi-token predictions are better for the fastText baseline, we still consider our method to be the most effective, as fastText is also not capable to predict multi-token candidates and yields to our method in the single token setup.

For all setups, the multi-token generation did not result in improvement of the scores. This can be explained by the flawed nature of our multi-token sampler and suggests major stream of future work.

7 Error Analysis

We can categorise common errors into several groups: failing to differentiate the real meaning of the hypernym, prediction of synonymical/same domain words instead of hyponyms, weakness of multi-token generator.

The first type of errors is related to incorrect recognition of a rare meaning of a synset and mistaking of it for a more common one. For example, for hypernym "depression.n.10" (pushing down) the correct prediction would be "click". However, almost all results are medical related predictions, e.g., headache, coma, schizophrenia.

An example of the second type of errors might be predictions of multi-token pipeline with Graph-BERT embeddings for "jazz_musician.n.01" hypernym. While the correct answer is "syncopator", top produced predictions are "singer", "dj", which obviously come from the same music-related domain.

For multi-token Node2vec we observed a lot of cases where one strong word was produced and further multi-token hypothesis would retain this first word and simply permute other different words.

Table 4: Example on Graph-BERT embeddings for the node "beverage.n.01" (single-token generation).

beverage.n.01 *Gold hyponyms: alcoholic drink, oenomel, fruit crush, cooler, alcoholic beverage, hot chocolate, fizz, ade, milk, inebriant, cocoa, drinking chocolate, drinking water, tea, java, mixer, refresher, tea-like drink, alcohol, coffee, fruit drink, ginger beer, wish-wash, potion, soft drink, near beer, smoothie, chocolate, cyder, intoxicant, fruit juice, cider, mate, hydromel*			
	pure BERT	replaced	mixed
1	beer	**milk**	**coffee**
2	**coffee**	drink	**milk**
3	**alcohol**	**coffee**	drink
4	water	butter	**tea**
5	cola	pot	chocolate
6	**tea**	whisky	butter
7	wine	**tea**	beer
8	**milk**	turkey	whisky
9	chocolate	chocolate	brandy
10	rum	brandy	water

Example output for test hypernym "suburb.n.01": suburb, suburb suburbs, suburbs, suburb suburban, suburb suburbs suburban, etc.

Because of the weak multi-token decoding mechanism, many predictions failed. For example, none of the setups managed to produce adequate hyponyms for "berry.n.01", because all correct answers are multi-token in BERT vocabulary.

All in all, the results are diverse an controversial. For instance, Table 9 demonstrates that graph information from node2vec is confusing for the model. According to Tables 4 and 7, Graph-BERT improves the ranking of the results. However, none of the models handles multi-token prediction: the only case where the model manages to predict the correct answer is presented in Table 8.

For instance, the model can generate candidates that are correct but they are not yet included to the taxonomy. In this case, the evaluation system will still mark them as incorrect. Therefore, as future work we plan not only improve current methods but also perform human evaluation of the results.

Another reason for the absolute low scores is the way the test set was generated. While in (Cho et al., 2020) the data is selected from the well-known domains like "pets", "food", "sport", our test set is generated randomly and thus comprises rare terms, which may be harder to process. At the same time, simple examples like "beverage" or "meal" gain better scores. As future work we want to tackle the problem of rare terms.

8 Conclusion

In this work, we presented a novel candidate-free task formulation for taxonomy enrichment. The contribution is three-fold: task proposal, according dataset and test of multi-modal approach. We performed a computational study of various methods using knowledge from BERT. We compared different graph-based embeddings on the task and projected them to the BERT vector space. Then we identified the best position for the projected graph embedding to be injected to the BERT model. The results demonstrate that incorporation of graph embedding is beneficial in the task of hyponym prediction using BERT. Nevertheless, the BERT architecture does not allow us to easily operate with multi-token words and the pipeline accumulates errors in each component. This may be room for improvement for generative models like GPT or T5 and their prompt-tuning.

All in all, the proposed task is proven to be very challenging paving the way for future research.

Acknowledgments

The first author was partially supported by the DFG through the project "ACQuA 2: Answering Comparative Questions with Arguments" (grants BI 1544/7-2 and HA 5851/2-2) as part of the priority program "RATIO:Robust Argumentation Machines" (SPP 1999). The work of Alexander Panchenko and Alsu Vakhitova was conducted in the framework of joint MTS-Skoltech laboratory. The work of Elena Tutubalina was supported by a grant from the President of the Russian Federation for young scientists-candidates of science (MK-3193.2021.1.6).

References

Saba Anwar, Artem Shelmanov, Alexander Panchenko, and Chris Biemann. 2020. Generating lexical representations of frames using lexical substitution. In *Proceedings of the Probability and Meaning Conference (PaM 2020)*, pages 95–103, Gothenburg. Association for Computational Linguistics.

Nikolay Arefyev, Maksim Fedoseev, Andrew Kabanov, and Vadim Zizov. 2020. Word2vec not dead: Predicting hypernyms of co-hyponyms is better than reading definitions. In *Computational Linguistics and Intellectual Technologies*, pages 13–32.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2017. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics*, 5:135–146.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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. Association for Computational Linguistics.

Yejin Cho, Juan Diego Rodriguez, Yifan Gao, and Katrin Erk. 2020. Leveraging WordNet paths for neural hypernym prediction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3007–3018, Barcelona, Spain (Online). International Committee on 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*, pages 177–186.

Daryna Dementieva, Daniil Moskovskiy, Varvara Logacheva, David Dale, Olga Kozlova, Nikita Semenov, and Alexander Panchenko. 2021. Methods for detoxification of texts for the russian language. *CoRR*, abs/2105.09052.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

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,*

San Francisco, CA, USA, August 13-17, 2016, pages 855–864. ACM.

Michael Hanna and David Mareček. 2021. Analyzing BERT's knowledge of hypernymy via prompting. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 275–282, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*.

Jesús Herrera, Anselmo Peñas, and Felisa Verdejo. 2005. Textual entailment recognition based on dependency analysis and *WordNet*. In *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification and Recognizing Textual Entailment, First PASCAL Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11-13, 2005, Revised Selected Papers*, volume 3944 of *Lecture Notes in Computer Science*, pages 231–239. Springer.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Wei Jin, Tyler Derr, Yiqi Wang, Yao Ma, Zitao Liu, and Jiliang Tang. 2021. Node similarity preserving graph convolutional networks. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, pages 148–156. ACM.

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. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907.

Henry Kucera and Winthrop Nelson Francis. 1967. *Computational analysis of present-day American English*. Brown University Press, Providence, RI.

Helen Langone, Benjamin R. Haskell, and George A. Miller. 2004. Annotating WordNet. In *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*, pages 63–69, Boston, Massachusetts, USA. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. 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. Association for Computational Linguistics.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6338–6347.

Irina Nikishina, Varvara Logacheva, Alexander Panchenko, and Natalia V. Loukachevitch. 2020a. Russe'2020: Findings of the first taxonomy enrichment task for the russian language. In *Computational Linguistics and Intellectual Technologies*.

Irina Nikishina, Varvara Logacheva, Alexander Panchenko, and Natalia V. Loukachevitch. 2020b. Studying taxonomy enrichment on diachronic wordnet versions. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 3095–3106. International Committee on Computational Linguistics.

Irina Nikishina, Mikhail Tikhomirov, Varvara Logacheva, Yuriy Nazarov, Alexander Panchenko, and Natalia V. Loukachevitch. 2022. Taxonomy enrichment with text and graph vector representations. *Semantic Web*, 13(3):441–475.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 113–120, Sydney, Australia. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in BERT. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 88–102,

Barcelona, Spain (Online). Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Timo Schick and Hinrich Schütze. 2019. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. *CoRR*, abs/1904.06707.

Michael Schlichtkrull and Héctor Martínez Alonso. 2016. MSejrKu at SemEval-2016 task 14: Taxonomy enrichment by evidence ranking. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1337–1341, San Diego, California. Association for Computational Linguistics.

Özge Sevgili, Artem Shelmanov, Mikhail Y. Arkhipov, Alexander Panchenko, and Chris Biemann. 2022. Neural entity linking: A survey of models based on deep learning. *Semantic Web*, 13(3):527–570.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. 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*, pages 801–808, Sydney, Australia. Association for Computational Linguistics.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. 2015. Network representation learning with rich text information. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 2111–2117. AAAI Press.

Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. 2020. Graph-bert: Only attention is needed for learning graph representations. *CoRR*, abs/2001.05140.

A Appendix

A.1 Hyperparameters for training graph embedding models

In this subsection we are listing the hyperparameters for training of graph embedding models. Unlisted parameters were set to default values.

Graph-BERT was initialized with fastText raw textual features (each node – average of according synset's lemmas). It was trained for 200 epochs on the node attribute reconstruction task, and the process continued for 200 more epochs on on the graph structure recovery task. The learning rate was set to 1e-3 and subgraph size to 5, and the resulting vectors were 300-dimensional.

Node2vec was trained to generate embeddings of same dimensionality, with 30 nodes in each random walk and 200 walks per node.

A.2 Space transformation MLP details

The MLP consists of three hidden layers (*source_embs* $\times$ 1024, 1024 $\times$ 512, 512 $\times$ *target_embs*) with exponential linear unit (ELU) activation. During training we used AdamW (Loshchilov and Hutter, 2017) optimizerFor the objective function we used a sum of cosine embedding loss between a model output and a target and a negated cosine similarity between a model output and a random negative example (any entity from the dataset that is not a target).

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_+ - \mathcal{L}_- \\ \mathcal{L}_+ &= 1 - \cos\left(y, \hat{y}\right) \\ \mathcal{L}_- &= \max\left(0, \cos\left(y_{neg}, \hat{y}\right)\right), \end{aligned} \tag{1}$$

where y – target embedding, $\hat{y}$ – predicted embedding, y_{neg} – negative example. The projection layer was trained for 500 epochs with batch size 64 and 1e-4 learning rate.

A.3 Multi-token generation algorithm details

The pseudocode for multi-token prediction is given in Algorithm 1. It is split into two functions: multi_tok_generate() and predict_candidates(). We are going to provide line-by-line explanation for each of them.

The multi_tok_generate() function takes as input the name of a parent synset, projected graph embedding, layer of replacement for the incorporation of the embedding and the replacement strategy. Line 2 generates tokens for the context construction "[MASK] is a {parent}", and line 3 encodes them with incorporation of projected embedding according to the scheme. Furthermore, the tokens and the hidden states are passed to the predict_candidates() function. It also takes the position of "[MASK]" token, which in this context prompt is 0. Finally, predict_candidates() returns a sorted list of tuples $(candidate, score)$, where each $candidate$ – predicted hyponym, and $score$ harmonic mean of scores for each token in the multi-token sequence.

The predict_candidates() function starts with saving the embedding of the "[MASK]" token that incorporates graph information (line 2). Furthermore, in the line 3 of the Algorithm 1 the single-token candidates are predicted. Function extract_mask_preds() (line 3) separates the predictions of hyponyms from the generated sentences. For example, sentence "[MASK] is a claim" was predicted into "dibs is a claim". Then extract_mask_preds() extracts the predicted hyponym "dibs" and returns it as a candidate paired with its score. Next, multi-token candidates of lengths 2 and 3 are generated (line 6). It is done with a beam search (line 7), which is illustrated schematically in Fig. 4. The beam_search() takes as input the tokenized sentence, position of a mask, saved embedding of a mask and a maximum length of the multi-token sequence. The beam search starts with insertion of one or two (according to the maximum length) additional mask tokens in the token sequence. Furthermore, the masks are predicted iteratively while maintaining best sequences as in a classical beam search algorithm.

The beam search generation ends when the maximum sequence length of the multi-token prediction is reached. The top hypotheses sentences as well as their scores are returned. Next, in the line 8 candidate hyponyms are extracted with extract_mask_preds() and together with scores are saved. Finally, multi- and single- token predictions are merged together and sorted by scores (line 10).

Algorithm 1 Algorithm of multi-token generation with BERT.
Inputs: name of parent synset $parent$, graph embedding of according masked child node projected into BERT space $proj\_emb$, layer of replacement l_num, replacement strategy $repl\_strategy$
Outputs: sorted list $final\_res$ that consists of tuples ($candidate$, $score$).

function MULTI_TOK_GENERATE($parent$, $proj\_emb$, l_num, $repl\_strategy$)
    $tokens \leftarrow$ tokenize("[MASK] is a {parent}")
    $hidden\_states \leftarrow$ BERT.encode($tokens, proj\_emb, repl\_strategy, l\_num$)
    $final\_res \leftarrow$ predict_candidates($hidden\_states, tokens, mask\_pos = 0$)
    return $final\_res$
end function

function PREDICT_CANDIDATES($hidden\_states, tokens, mask\_pos$)
    $mask\_hidden\_state \leftarrow hidden\_states[mask\_pos]$
    $single\_tokens, single\_scores \leftarrow$ pred_single_mask($BERT, hidden\_states, mask\_pos$)
    $f\_preds, f\_scores \leftarrow$ extract_mask_preds($single\_tokens, single\_scores$)
    $multi\_preds, multi\_scores \leftarrow$ [], []
    for $seq\_len \in [2, 3]$ **do**
        $new\_tokens, new\_scores \leftarrow$
            $\leftarrow$ beam_search($tokens, mask\_pos, mask\_hidden\_state, seq\_len$)
        $m\_p, m\_s \leftarrow$ extract_mask_preds(new_tokens, new_scores)
        $multi\_preds.append(m\_p)$
        $multi\_scores.append(m\_s)$
    end for
    $final\_res \leftarrow$ merge_sort_results($f\_preds, f\_scores, multi\_preds, multi\_scores$)
    return $final\_res$
end function

Table 5: CHSP prediction scores for single-token hyponyms generation for different source graph embeddings, replacement strategies and substitution layer (x100).

Graph embeddings	Context	Replaced	Layer	MRR@5	MRR@10	MRR@20	P@1	P@5	P@10
Node2vec	Yes	Yes	1st	0.975	1.831	2.252	0.000	0.670	1.186
		Mix	1st	2.328	2.685	2.903	1.546	1.186	1.005
		Yes	6th	3.316	3.799	4.070	1.031	1.804	1.340
		Mix	6th	2.414	3.079	3.391	1.289	1.289	1.469
		Yes	12th	2.436	3.185	3.486	1.289	1.082	1.160
		Mix	12th	3.329	4.073	4.597	1.031	1.649	1.675
Graph-BERT	Yes	Yes	1st	4.502	4.995	5.371	3.093	1.598	1.340
		Mix	1st	1.448	1.813	2.033	0.773	0.876	0.979
		Yes	6th	5.503	6.216	6.453	3.093	2.371	2.010
		Mix	6th	2.981	3.500	3.836	1.546	1.649	1.495
		Yes	12th	5.215	5.674	6.027	3.093	2.113	1.598
		Mix	12th	**7.229**	**8.037**	**8.624**	**3.608**	**3.247**	**2.474**

Table 6: CHSP prediction scores for multi-token hyponyms generation for different source graph embeddings, replacement strategies and substitution layer (x100).

Graph embeddings	Context	Replaced	Layer	MRR@5	MRR@10	MRR@20	P@1	P@5	P@10
Node2vec	Yes	Yes	1st	0.945	1.231	1.395	0.515	0.515	0.515
		Mix	1st	0.287	0.374	0.492	0.000	0.206	0.180
		Yes	6th	0.587	0.674	0.732	0.200	0.300	0.210
		Mix	6th	1.924	2.073	2.193	1.200	0.740	0.550
		Yes	12th	0.520	0.534	0.586	0.500	0.120	0.070
		Mix	12th	0.453	0.534	0.610	0.400	0.120	0.110
Graph-BERT	Yes	Yes	1st	1.908	2.054	2.149	1.400	0.680	0.500
		Mix	1st	1.350	1.522	1.625	0.800	0.600	0.500
		Yes	6th	**2.150**	**2.281**	**2.378**	**1.600**	**0.740**	0.530
		Mix	6th	1.468	1.694	1.806	0.700	0.700	**0.560**
		Yes	12th	1.278	1.312	1.368	1.200	0.340	0.190
		Mix	12th	1.767	1.899	2.071	1.400	0.540	0.390

Table 7: Example on Graph-BERT embeddings for the node “meal.n.01” (multi-token generation)

meal.n.01 *Gold hyponyms: nosh-up, tea, snack, breakfast, supper, brunch, tiffin, lunch, refection, mess, ploughman's lunch, dejeuner, feast, spread, afternoon tea, picnic, dinner, square meal, luncheon, teatime, banquet, bite, buffet, potluck, collation*			
	pure BERT	replaced	mixed
1	life	**breakfast**	**breakfast**
2	food	breakfast lunch	breakfast lunch
3	**dinner**	**lunch**	**lunch**
4	**lunch**	breakfast dinner	breakfast dinner
5	**breakfast**	breakfast lunch dinner	breakfast lunch dinner
6	everything	lunch dinner	lunch dinner
7	love	breakfast dining	**dinner**
8	tomorrow	breakfast meals	breakfast meal
9	today	breakfast meal	breakfast lunch meal
10	nothing	breakfast lunch dining	breakfast meals

Table 8: Example on node2vec embeddings for the node “stock.n.01” (multi-token generation).

stock.n.01 *Gold hyponyms: capital stock, treasury stock, quarter stock, preference shares, growth stock, preferred stock, no-par-value stock, voting stock, common shares, authorized shares, hot stock, ordinary shares, authorized stock, float, reacquired stock, common stock, no-par stock, common stock equivalent, treasury shares, preferred shares, hot issue, control stock, watered stock*			
	pure BERT	replaced	mixed
1	stock	capital	capital
2	one	capital cash	capital cash
3	c	capital investment	capital investment
4	b	capital financing	capital money
5	today	capital funds	capital financial
6	x	capital financial	capital equity
7	gold	capital income	**capital stock**
8	everything	capital funding	capital financing
9	life	capital revenue	capital funds
10	r	capital crop	capital leverage

Table 9: Example on node2vec embeddings for the node “citrus.n.01” (single-token generation)

citrus.n.01 *Gold hyponyms: citrange, citron, grapefruit, kumquat, lemon, lime, mandarin, orange, pomelo*			
	pure BERT	replaced	mixed
1	fruit	date	date
2	one	year	tree
3	rose	horse	year
4	another	turkey	snow
5	**citrus**	dates	horse
6	cherry	tree	turkey
7	**orange**	snow	dates
8	tomato	calendar	winner
9	mine	winner	grass
10	wood	loser	trees