On the Role of Conceptualization in Commonsense Knowledge Graph Construction

Mutian He[1], **Yangqiu Song**[1], **Kun Xu**[2], **Yu Dong** [2]
[1]Hong Kong University of Science and Technology
[2]Tencent
{mhear, yqsong}@cse.ust.hk
{kxkunxu,dyu}@tencent.com

Abstract

Commonsense knowledge graphs (CKG) like ATOMIC and ASER are substantially different from conventional KG as they consist of much larger number of nodes formed by loosely-structured texts, which, though, enable them to handle highly diverse queries in natural language regarding commonsense, lead to unique challenges to automatic KG construction methods. Besides identifying relations absent from the KG between nodes, the methods are also expected to explore absent nodes represented by texts, in which different real-world things or entities may appear. To deal with innumerable entities involved with commonsense in real world, we introduce to CKG construction methods *conceptualization*, i.e., to view entities mentioned in texts as instances of specific concepts or vice versa. We build synthetic triples by conceptualization, and further formulate the task as triple classification, handled by a discriminatory model with knowledge transferred from pretrained language models and fine-tuned by negative sampling. Experiments demonstrate that our methods could effectively identify plausible triples and expand the KG by triples of both new nodes and edges in high diversity and novelty.

1 Introduction

Commonsense knowledge, such as knowing that *A trophy could not fit in a suitcase because the trophy is too big*, is implicitly acknowledged among human beings through real life experience rather than systematic learning. As a result, artificial intelligence meets difficulties to capture such commonsense. To deal with the issue, Commonsense Knowledge Graphs (CKG) like ConceptNet (Speer et al., 2017), ATOMIC (Sap et al., 2019), ASER (Zhang et al., 2019), etc. are proposed, which are aimed at collecting and solidifying such implicit commonsense in the form of triples $\langle h, r, t \rangle$, with the head node h and the tail node t connected by a relation (i.e., edge) r. However, a key difference between traditional knowledge graphs (like WordNet, Freebase, etc.) and CKG is that the commonsense is often difficult to be represented as two strictly-formed nodes compared by a specific relation. Instead, recent approaches represent a node with loosely-structured texts, either annotated by human or retrieved from text corpora, and nodes are then linked by one of some predefined relations, such as the triple $\langle h$: I'm hungry, r: Result, t: I have lunch$\rangle$ in ASER.[1]

Such CKG, storing an exceptional large number of triples as shown in Table 1, are capable to represent a much broader range of knowledge, and to handle flexible queries with regard to commonsense. However, the complexity of the real world commonsense is still immense. Particularly, with innumerable eventualities involved with commonsense in real world, it is of high cost for a CKG to cover all of them in nodes; and even when covered, to acquire the corresponding relation between each pair of nodes is of quadratic difficulty. Such situation is particularly demonstrated by the sparsity of edges in current CKG as shown in Table 1: even automatic extractive methods, as used in ASER, fail to capture edges between a majority of nodes. Therefore, alternative KG construction methods are expected.

As nodes are represented by texts, utilizing semantic information becomes critical for CKG construction. Such semantic information could be leveraged by large pretrained language models like BERT (Devlin et al., 2018) and GPT-2 (Radford et al., 2019): These models capture rich semantic knowledge

[1]Words in ASER are lemmatized, but in this paper we always show the original texts for easier understanding.

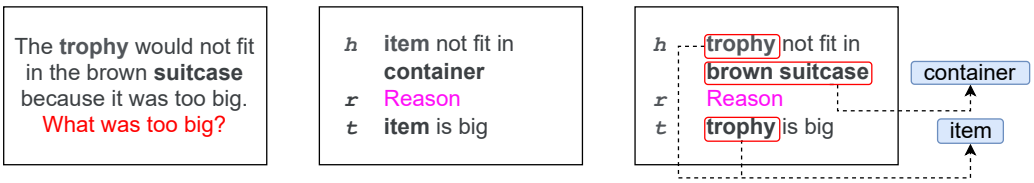
	ASER	ATOMIC
#Nodes	194.0M	309.5K
#Triples	64.4M	877.1K
#Relation Types	15	9
Average Degree	0.66	5.67
Entity Coverage	52.33%	6.98%
Average Distinct Entity	0.026	0.082

Table 1: Statistics for recently proposed Commonsense Knowledge Graphs. Entity Coverage is calculated by the proportion of top 1% frequent entities in Probase, that are mentioned by nodes in each CKG. Average Distinct Entity is given by average number of distinct Probase entities per node in each CKG. Core version of ASER is used for these two results.

from unsupervised pretraining, which could be transferred to a wide range of downstream tasks to reach impressive results. Therefore, efforts have been made to extract knowledge from such models for KG construction. COMeT (Bosselut et al., 2019) is a notable attempt which fine-tuned GPT on CKG triples to predict tails from heads and relations. However, it is often observed that such generative neural models suffer from diversity and novelty issue: They tend to overproduce high-probability samples similar to training, while fail to cover a broader range of possible triples absent in the KG with diverse entities involved. In contrary, methods like (Malaviya et al., 2019; Davison et al., 2019; Yao et al., 2019) incorporate language models to discriminatory CKG completion tasks such as link prediction and triple classification to evaluate whether a triple is valid, and could be extended to arbitrary texts for triples on various KG. However it would be computationally infeasible to identify new plausible triples on recent large CKG if we aimlessly explore new nodes and evaluate each possible triple, without leveraging existent nodes and triples as in generative methods. Therefore, all methods above have certain shortcomings.

Particularly, we observe that current methods miss the variation of real world entities, which is a critical factor for the diversity of nodes. For example, a CKG may cover the node *I eat apple* and its relevant edges, but the node *I eat banana* might be missing or the edges are incomplete. As shown in Table 1, a large portion of the most common real world entities in Probase are never mentioned in the recent CKG like ASER, not to say edges related to the entities. More, as demonstrated by the low number of average distinct entity per node, directly expansion of the CKG would not be an effective solution.

To relieve the issue, we posit the importance of a specific portion of human commonsense, **conceptualization**, which, though found useful for certain NLU tasks (Song et al., 2011; Wang et al., 2015; Hua et al., 2015), is not yet investigated in depth in this area. As observed by psychologists, "*concepts are the glue that holds our mental world together*" (Murphy, 2004). Human beings are able to make reasonable inference by utilizing the *IsA* relationship between real world concepts and instances. For



(a) A commonsense reasoning problem

(b) An existent triple in the CKG

(c) Conceptualization on the extracted triple

Figure 1: A sample for conceptualization in CKG. Given a commonsense reasoning problem like (a), even the corresponding triple (c) is not in the KG, a triple (b) which is present in the KG could be used through abstraction. This is by identifying real world entities of *trophy* and *brown suitcase* in texts of (c), and then substituting them using *IsA* relations. Following the same idea, (c) could be produced from (b) to be included in the CKG via instantiation.

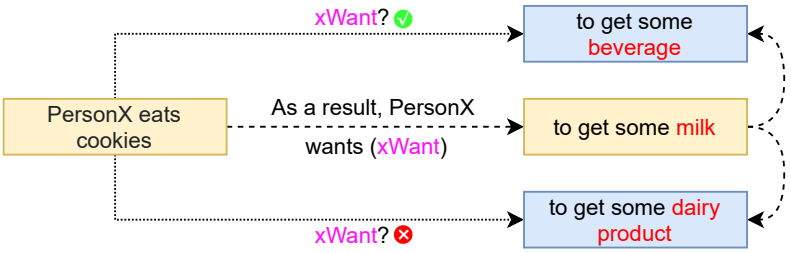


Figure 2: A sample from ATOMIC for discriminating conceptualization. Some ways of conceptualization, like replacing *milk* with *beverage* in the tail node is valid, while with *dairy* is invalid under the context, as with commonsense one would often want something to drink after eating cookies, while the general concept of *dairy* is not involved in such scenario, and dairy products are not all drinkable.

example, without knowing what a *floppy disk* is, given that it is a *memory device*, people may infer that it may store data, and it may be readable by a computer, etc. From this viewpoint, instead of directly building triples with countless entities, a CKG could be broadly expanded to handle various queries as shown in Figure 1 by such substitution of instances appeared in the head or tail by the corresponding concepts (i.e., **abstraction**), or vice versa (i.e., **instantiation**), given an extra CKG of *IsA* relations.

However, such conceptualization is never strict induction or deduction guaranteed to be true, As shown in Figure 2, it is still a challenging task to determine whether a triple built from conceptualization is reasonable, and requires both the context within the triple and a broader range of commonsense. Therefore, we propose to formulate the problem as a triple classification, one of the standard tasks for knowledge graph construction (Socher et al., 2013). Under this task, we utilize the rich semantic knowledge from large pretrained language models by fine-tuning them as discriminators. In this way, such discriminators are expected to take triples with arbitrary node (seen or unseen, produced by conceptualization or not) as inputs and evaluate whether the triple is reasonable.

To conclude, our contributions are three-fold:

1. We introduce conceptualization to CKG construction to explore a broad range of new triples.
2. We formulate the conceptualization-based CKG construction as a triple classification task to be performed on synthetic triples.
3. We propose a method for the task by fine-tuning pretrained language models with negative sampling.

Our code and pipeline is available at `github.com/mutiann/ccc`.

2 Methodologies

Our methodologies of CKG construction is based on the idea that given a set of ground truth triples as *seeds*, new triples could be built from them by abstraction or instantiation of entities mentioned in head or tail node, i.e., substituting a mentioned entity with a more general or more specific one, using the particular commonsense of *IsA* relations. Therefore, we need a CKG, **K**, viewed as seeds, as well as a conceptualization KG, **C**, both denoting a set of triples $\langle h, r, t \rangle$, while in **C** r is always *IsA*.

2.1 Conceptualization

For the diverse ways for conceptualization in commonsense, the **C** must be of sufficient coverage on various real-world entities connected by *IsA* relations. Therefore, we choose Probase (Wu et al., 2012), which is a large-scale knowledge graph consists of 17.9M nodes and 87.6M edges, extracted from various corpora, and is proved to be suitable for conceptualization (Song et al., 2011).

Besides, a single entity may have various ways of abstraction or instantiation, with different typicality. For example, either Linux or BeOS is an *operating system*, and a *pen* is either a writing tool or an enclosure for animals, though for the both examples the two choices are not equally common. Since

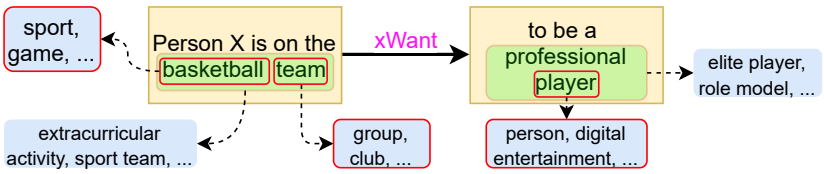


Figure 3: A sample for identifying entities. All noun phrases are identified as possible entities on which substitutions are proposed. The phrases include *basketball*, *basketball team*, *team*, *player*, *professional player*, but except *professional* which is tagged as an adjective here.

Probase is extracted from real corpora, the frequencies of texts showing the triple $\langle h, IsA, t\rangle$ in the source corpora could demonstrate how common the relation is. Such frequencies f are given by Probase along with the triple, forming 4-tuples of $\langle h, IsA, t, f\rangle$. The frequency information in Probase allows us to balance between *IsA* relations of different typicality and filter out noises of rare relations in the graph.

With **C** prepared, conceptualization could then be performed on any mentioned real-world entity in the head or tail nodes in each triple, that could be a single noun or noun phrases, serving as subjects, objects or even modifiers, as shown in Figure 3. What leads to more complexity is that although nodes in **C** are all real-world entities under some contexts, the word or phrase could be used in different manners within a triple of interests. Therefore, for raw texts as in ATOMIC, we choose to perform dependency parsing on each node by spaCy (Honnibal and Montani, 2017). Then we need to filter out all nouns or noun phrases that also present in **C** as possible candidates.

Algorithm 1 IDENTIFYCONCEPTUALIZATION

Input: W: a node, represented by words, $[w_1, w_2, ..., w_n]$
P: POS tag of each word in W, $[p_1, p_2, ..., p_n]$
D: Dependency tree for W
C: Probase of *IsA* relations, $\{\langle x, IsA, y, f\rangle\}$
Result: S: List of substituted word sequences $[W'_1, W'_2, ...]$
F: List of frequencies for each substitution $[f_1, f_2, ...]$

```
S ← []
F ← []
for k ∈ [1, n] do
    if p_k ∈ {noun, propn} then
        T ← subtree of w_k in D
        L ← min_{x∈T} (index of x in W)
        R ← max_{x∈T} (index of x in W)
        foreach (l, r), L ≤ l ≤ k ≤ r ≤ R do
            E ← W[l..r]
            A ← {(S, f)|(E, IsA, S, f) ∈ C}
            I ← {(S, f)|(S, IsA, E, f) ∈ C}
            foreach (s, f) ∈ A ∪ I do
                S.add([w_1, ..., w_{l-1}] + s + [w_{r+1}, ..., w_k])
                F.add(f)
            end
        end
    end
end
```

The method to identify entities is given in Algorithm 1: we iterate through each noun w as the root of the entity, and choose all continuous sequences of words within the range of the subtree corresponding to w in the dependency tree, to ensure that all entities rooted by w (possibly with different modifiers) are collected. We then query the possible abstraction and instantiation of each candidate with Probase, and,

for any result, add the substituted texts and the corresponding frequency into the list of results. If the texts in the CKG are given in the original form (i.e. not lemmatized, unlike ASER), we further use a set of rules to inflect the substitution s returned, and to modify the determiner (if any) in the returned text so as to avoid any false statistical clues of grammatical mistakes introduced by the substitution.

2.2 Discriminator

We model our task as a triple classification problem, a binary classification task on KG aimed at judging whether a triple $\langle h, r, t \rangle$ is true or not (Socher et al., 2013). As in the common case of Knowledge Base Construction, **K** generally contains merely ground truth, i.e., positive samples. Therefore, we turn to construct a negative sample set D_- with triples corrupted from the positive sample set, in combination with samples from **K** as positive samples D_+. Then the model could be trained by negative sampling and evaluated by the classification accuracy under held-out samples and the corresponding negative samples generated in the same way.

Two different settings are used to generate negative samples.

1. Node Substitution (**NS**): the common corruption method as in (Bordes et al., 2013) that randomly substitute the head or tail (each with 0.5 probability) by another node. For CKG like ATOMIC that head nodes and tails nodes, as well as tail nodes from triples with different relations, could often be easily distinguished between each other[2], we follow (Socher et al., 2013) to pick random heads only from other heads, and random tails only from other tails appear in triples with the same relation.

2. Entity Conceptualization (**EC**): to enable the model to identify false triples with inappropriate conceptualization, we randomly choose the head or tail (each with 0.5 probability) and corrupt the node as in Section 2.1 by substituting an entity in the node with its abstraction or instantiation. This method ensures that the substituted nodes are often plausible. Then we use the triple with head or tail substituted as negative samples. Particularly, we make use of the frequencies returned by Algorithm 1 as weights (or unnormalized probabilities), based on which we sample from possible conceptualized nodes, as shown in Algorithm 2. In this way, we make a balance between diversity and typicality of *IsA* relations used.

Algorithm 2 BUILDSAMPLEEC

Input: N: A node, represented by a sequence of words
P: POS tags of words in N
D: Dependency tree for N
C: Probase $\{\langle x, IsA, y, f \rangle\}$
Result: N': corrupted node
$S, F \leftarrow$ IDENTIFYCONCEPTUALIZATION(N, P, D, C)
$W \leftarrow F \,/\, \sum F$
$k \sim Categorical(W)$
$N' \leftarrow S_k$

Both methods are in line with the standard assumption in knowledge graph completion tasks that view potential triples not presented in the ground truth KG as negative during training, even though they could be valid ones absent in the original KG. In this way, we expect that the model would be capable of discriminate whether a triple is valid, when the triple is corrupted in either way. To reduce noise in training and evaluation, negative samples are filtered to ensure that they are different from any positive sample, which matches the *filtered* setting in (Bordes et al., 2013).

To make the best use of semantic information from the textual description of nodes in our task, we apply the widely used transformer-based pretrained language models like BERT as the discriminatory model. Particularly, the structure of our task matches the Next Sentence Prediction (NSP) task in (Devlin

[2]For instance, head node in ATOMIC always starts with *Person X*, and tail nodes for relation type *xAttr* (attribute of Person X) are often adjectives.

et al., 2018). As a result, we follow the similar setting that takes pairs of sentences separated by a special [SEP] token and marked by different token type IDs as inputs, with h the first sentence and the concatenation of r and t the second sentence. Binary classification is then performed by a fully-connected layer taking the final layer representation corresponding to the [CLS] token in the transformer model, as shown in Figure 4. All parameters in the model, except those in the final fully-connected layer for binary classification, could be loaded from the pretrained BERT model. Then the model is fine-tuned using the positive and negative samples mentioned above, with 1:1 frequency during training, using binary cross-entropy loss below, based on the output s, which is a scalar after a logistic sigmoid activation, indicating the confidence on the input being valid:

$$L = -\sum_{(x,y)\in D_+\cup D_-} (y \log s + (1-y)\log(1-s)). \tag{1}$$

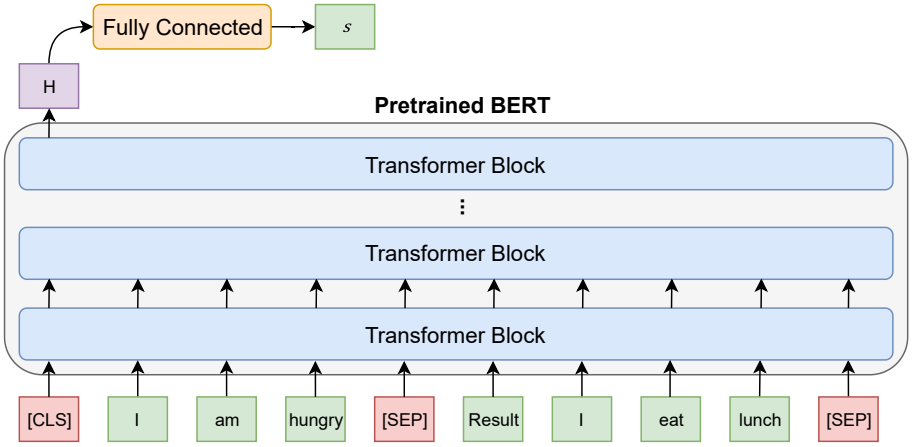


Figure 4: Architecture of the BERT-based discriminator model. Raw texts are fed into the model to predict the binary label y. All except the last fully-connected layer are pretrained but not frozen.

3 Experiments

3.1 Datasets

Two different datasets, ATOMIC and ASER, are used in our experiments, which are typical CKG using open-form texts as nodes. Earlier CKG such as ConceptNet (Speer et al., 2017), are not discussed as in ConceptNet, unlike the more recent CKG, only simple texts, mostly noun phrases, are used as nodes, and previous work could already reach close-to-human results (Saito et al., 2018).

3.1.1 ATOMIC

ATOMIC is a commonsense knowledge graph of 877K triples on cause-and-effect relations between everyday activities (Sap et al., 2019). With in ATOMIC, head nodes, or base events are extracted from text corpora, with the form of some person's action or states (e.g., *Person X is on the basketball team*). The dataset further categorizes the cause-and-effect relations into 9 types, covering the intention, prerequisites and impacts with regard to the agent *Person X* and other people. The tail entities are then written in open-form text by crowd-sourced workers under these categories. In this way, a broad range of eventualities and their relations are covered by the dataset. We follow the original data split of ATOMIC in our experiments.

3.1.2 ASER

ASER (Zhang et al., 2019) is a large scale knowledge graph of 194M nodes representing verb-centric eventualities matching certain patterns (e.g., *s-v-o* for *I love dogs* and *s-v-v-o* for *I want to eat an apple*), extracted from various corpora. Relations of 15 types between nodes are then extracted as well, identified by matching the texts with language patterns such as $\underline{E_1}$, because $\underline{E_2}$ for **Reason** and $\underline{E_1}$, $\underline{E_2}$ instead for

ChosenAlternative. Total 194.0M nodes and 64.4M triples are extracted in ASER. In our experiments, we use the core release of ASER with 27.6M nodes and 10.4M edges. Triples of **CoOccurance** type and isolated nodes are further removed to create a smaller and cleaner KG with 1.4M nodes and 1.1M edges. Triples are then randomly split to train, dev, and test set by 8:1:1.

3.2 Settings

To build our CKG Construction by Conceptualization (**CCC**) discriminator, we follow the scheme for fine-tuning BERT on downstream tasks as in (Devlin et al., 2018), and use the pretrained 12-layer BERT-base model on GTX1080 GPUs with 8GB memory. To evaluate the impact of two different ways for producing negative samples given in Section 2.2, and to trade off between model capabilities to discriminate triples under general cases and specifically identifying inappropriate conceptualization, we perform experiments with different percentage of negative samples built by conceptualization, i.e., the EC setting. Specifically, models with 50%, 75%, and 87.5% negative samples created by EC (and the rest by NS) are trained and reported. As for evaluation, negative samples are generated on dev and test samples as well by both methods, forming the EC and NS for dev and test set with 1:1 positive and negative samples. Under EC, those triples with nodes containing no entities to be conceptualized (e.g., *I am fine*, for which Algorithm 1 returns empty results) are ignored. Nevertheless, 79.65% and 83.56% of triples in dev and test sets are collected in EC set for ATOMIC and ASER respectively, showing that a majority of triples could be conceptualized. Test results of models at best EC dev accuracy are reported.

3.3 Baselines

We use COMeT[3] and KG-BERT[4] as our baselines, and have them trained on the datasets above. Particularly, our model would degenerate to KG-BERT with 0% EC samples as the NS setting is what KG-BERT trained by. Since COMeT itself is not a discriminative model, we use the perplexity per token[5] as its score given to each triple, and use the dev set to find a classification threshold with best accuracy.

	ASER		**ATOMIC**	
	EC	NS	EC	NS
COMeT	0.6388	0.5869	0.6927	0.5730
KG-BERT	0.7091	**0.8018**	0.7669	0.6575
CCC-50	0.8716	0.7775	0.9016	**0.7840**
CCC-75	0.8995	0.7250	0.9221	0.7446
CCC-87.5	**0.9156**	0.6635	**0.9355**	0.6980
CCC-75-scratch	0.8284	0.5587	0.8579	0.5003
CCC-75-RoBERTa	0.8999	0.6938	0.9305	0.7350

Table 2: Results of accuracy on EC and NS test set of baselines and our models on different datasets. CCC (CKG Construction by Conceptualization) denotes our model, with the number attached representing percentage of EC training.

3.4 Results

3.4.1 Triple Classification

Test accuracies of triple classification on both methods are given in Table 2. As shown by the results, COMeT is lack of discriminative power, which is consistent with the results in (Malaviya et al., 2019). Discriminatory models like KG-BERT that were successfully applied on traditional KG, produce satisfying results on CKG as well, while our methods perform better than both baseline methods by a margin on EC tests. Hence it is demonstrated that introducing conceptualization during training is effective to

[3]Available at `github.com/atcbosselut/comet-commonsense`

[4]Available at `github.com/yao8839836/kg-bert`

[5]Unlike the case in (Malaviya et al., 2019), conceptualization would not significantly change the length of a triple, so we only use the NORMALIZED setting.

create a model capable of identifying false conceptualization. Particularly, percentage of EC samples in training is critical for a trade-off between EC and NS tasks: Increased EC percentage would lead to better EC results, but NS results may drop. While in ATOMIC CCC models reach better results on NS than KG-BERT, which is possibly due to that ATOMIC nodes are mostly about everyday activities, contrast to ASER which covers a broader range of topics. Therefore by EC training a more diverse set of nodes could be seen by the model in training, and could be helpful for the model to generalize in the NS test.

3.4.2 Ablation Studies

We performed ablation studies on two different aspects, to examine the importance of pretraining and model selection. With the model trained from scratch on our task without using pretrained parameters, the performance significantly drops as shown in CCC-75-scratch results in Table 2. We also attempted to use RoBERTa, an alternative pretrained language model that made improvements on BERT training and has demonstrated better performance on downstream tasks (Liu et al., 2019). However, the results using pretrained RoBERTa-base model (CCC-75-RoBERTa) are generally on par with our model using BERT. This could be possibly explained by that BERT is sufficient in our current settings, that RoBERTa uses larger batch size while on our GPU the batch size is more limited, and that the NSP pretraining task is used in BERT but absent in RoBERTa, as NSP exactly matches the input scheme of our task.

	ASER		**ATOMIC**	
	COMeT	CCC-75	COMeT	CCC-75
N/Seed	10	8.28	10	5.02
Dist-N	24.68%	96.57%	6.49%	51.26%
Dist-1	1.62%	6.30%	0.63%	8.34%
Dist-2	10.56%	15.45%	2.87%	49.76%
N/T N	88.48%	98.60%	10.30%	94.65%
N/U N	93.38%	99.18%	69.17%	96.66%
Dist-N-Norm	10.74%	84.16%	4.35%	46.36%
N/T N-Norm	9.37%	86.01%	5.12%	87.95%
N/U N-Norm	65.72%	95.02%	58.71%	93.01%

Table 3: Results for diversity and novelty on generations, larger for better results. All rows except N/seed are given in percentage.

3.4.3 Generations

We generate triples using the test set from ASER and ATOMIC as seeds, by both COMeT with 10-beam search and CCC-75 model by conceptualization, and apply various metrics on results as shown in Table 3. Both methods may produce a large number of triples, as given by the number of generations per seed, **N/Seed**. For diversity, we report **Dist-1** (number of distinct words per node), **Dist-2** (number of distinct bigrams per node), and **Dist-N** (number of distinct nodes per node). Due to the different number of generated triplets, the results are all normalized by number of nodes.[6]

Novelty is measured by **N/T N**, proportion of nodes among all produced nodes that are novel, i.e. not present in the training set; and **N/U N**, proportion of novel distinct nodes, among all distinct nodes. Moreover, since generative methods may produce nodes of essentially the same meaning with slight changes in forms, we also have the produced nodes normalized, by removing structural words like determiners, auxiliary verbs, pronouns, etc. We then report the results for the metrics above but applied to generations after such normalization, denoted as **Dist-N-Norm**, **N/T N-Norm**, and **N/U N-Norm** respectively. Furthermore, samples of generations by both models are shown in Table 4.

It is clearly demonstrated in the diversity results that a majority of generations by COMeT are similar to each other given certain head node and relation, as number of distinct nodes, words and bigrams are

[6]Results from (Bosselut et al., 2019) of test perplexity are reproduced in our experiments. Differences on diversity metrics are due to that we use 10-beam search instead for fair comparison.

all relatively low. The novelty results further show that generated nodes are often similar to the seen ones in the train set as well. It could be particularly observed that though the original diversity and novelty metrics appear to be acceptable, which is consistent with (Bosselut et al., 2019), results drop sharply when the generations are normalized. This indicates that COMeT may produce slightly different nodes paraphrasing each other, as shown in Table 4 that four of five generated tails are similar to each other (saying *he does not get it*). While this is not the case for CCC as generated nodes are mostly discussing about different entities, and the results would often be diverse and novel.

	head	*tail*
Seed	another promises him a scholarship	his parents own a successful business
COMeT	–	he never gets it he does not get it he never gets one he could not pay it he does not receive it
CCC-75	another promises him a grant another promises him an award	–
	–	his parents own a successful shop his parents own a successful bank his parents own a successful hotel

Table 4: Samples for ASER generations given the seed. In this sample the head and tail are connected by the relation of **Concession**, i.e. *although*.

4 Related Work

Automatic construction of structured knowledge graph is a well-studied task, and a number of learning-based methods were proposed, including methods of KG embeddings and graph neural networks like (Bordes et al., 2013; Socher et al., 2013; Yang et al., 2014; Trouillon et al., 2016; Schlichtkrull et al., 2018; Shang et al., 2019) that leverage structural information and are aimed finding new edges between known nodes, which could be further enhanced by textual information from the node as in (Wang and Li, 2016; Xie et al., 2016; Xiao et al., 2017; An et al., 2018).

Textual information is more critical on commonsense knowledge graphs with nodes carrying complicated eventualities, often in open-form text. Therefore, (Li et al., 2016) proposed to score ConceptNet triples by neural sequence models taking text inputs so as to discover new triples, while (Saito et al., 2018) and (Sap et al., 2019) further propose to generate tail nodes by a sequence-to-sequence LSTM model with head and relation as inputs. Recently powerful large pretrained models like BERT and GPT-2 are proposed (Devlin et al., 2018; Radford et al., 2019), from which it is observed by (Trinh and Le, 2018; Radford et al., 2019) that rich knowledge including commonsense could be extracted. Therefore, different ways for KG construction are introduced on such models as downstream tasks: In KG-BERT, BERT is fine-tuned for KG completion tasks like link prediction and triple classification (Yao et al., 2019); COMeT uses GPT-based models to generate tails (Bosselut et al., 2019); LAMA directly predict masked words in triples on various KG by BERT (Petroni et al., 2019); (Davison et al., 2019) considers both generation of new tails and scoring given triples; (Malaviya et al., 2019) utilizes both structural and semantic information for CKG construction on link prediction tasks.

5 Conclusion

We introduce conceptualization to commonsense knowledge graph construction and propose a novel method for the task by generating new triples with conceptualization and examine them by a discriminator transferred from pretrained language models. Future studies would be focused on strategies of conceptualization and its role in natural languages and commonsense by deep learning approaches.

References

Bo An, Bo Chen, Xianpei Han, and Le Sun. 2018. Accurate text-enhanced knowledge graph representation learning. In *Proceedings of NAACL-HLT*, pages 745–755.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of NIPS*, pages 2787–2795.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of ACL*, pages 4762–4779.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of EMNLP-IJCNLP*, pages 1173–1178.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *Computing Research Repository*, arXiv:1810.04805.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing.

Wen Hua, Zhongyuan Wang, Haixun Wang, Kai Zheng, and Xiaofang Zhou. 2015. Short text understanding through lexical-semantic analysis. In *Proceedings of ICDE*, pages 495–506.

Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense knowledge base completion. In *Proceedings of ACL*, pages 1445–1455.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Chaitanya Malaviya, Chandra Bhagavatula, Antoine Bosselut, and Yejin Choi. 2019. Exploiting structural and semantic context for commonsense knowledge base completion. *Computing Research Repository*, arXiv:1910.02915.

Gregory Murphy. 2004. *The big book of concepts*. MIT press.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of EMNLP-IJCNLP*, pages 2463–2473.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Itsumi Saito, Kyosuke Nishida, Hisako Asano, and Junji Tomita. 2018. Commonsense knowledge base completion and generation. In *Proceedings of CoNLL*, pages 141–150.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. ATOMIC: An atlas of machine commonsense for if-then reasoning. In *Proceedings of AAAI*, pages 3027–3035.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of ESWC*, pages 593–607.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of AAAI*, pages 3060–3067.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of NIPS*, pages 926–934.

Yangqiu Song, Haixun Wang, Zhongyuan Wang, Hongsong Li, and Weizhu Chen. 2011. Short text conceptualization using a probabilistic knowledgebase. In *Proceedings of IJCAI*, pages 2330–2336.

Robert Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An open multilingual graph of general knowledge. In *Proceedings of AAAI*, pages 4444–4451.

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *Computing Research Repository*, arXiv:1806.02847.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of ICML*, pages 2071–2080.

Zhigang Wang and Juanzi Li. 2016. Text-enhanced representation learning for knowledge graph. In *Proceedings of IJCAI*, pages 1293–1299.

Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. 2015. Query understanding through knowledge-based conceptualization. In *Proceedings of IJCAI*, pages 3264–3270.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. 2012. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of SIGMOD*, pages 481–492.

Han Xiao, Minlie Huang, Lian Meng, and Xiaoyan Zhu. 2017. SSP: semantic space projection for knowledge graph embedding with text descriptions. In *Proceedings of AAAI*, pages 3104–3110.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *Proceedings of AAAI*, pages 2659–2665.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *Computing Research Repository*, arXiv:1412.6575.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for knowledge graph completion. *Computing Research Repository*, arXiv:1909.03193.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, Cane Wing-Ki, et al. 2019. ASER: A large-scale eventuality knowledge graph. *Computing Research Repository*, arXiv:1905.00270.