DualTKB: A Dual Learning Bridge between Text and Knowledge Base

Pierre L. Dognin[*]
IBM Research
pdognin@us.ibm.com

Igor Melnyk[*]
IBM Research
igor.melnyk@ibm.com

Inkit Padhi[*]
IBM Research
inkpad@ibm.com

Cicero Nogueira dos Santos
AWS AI
cicnog@amazon.com

Payel Das
IBM Research
daspa@us.ibm.com

Abstract

In this work, we present a dual learning approach for unsupervised text to path and path to text transfers in Commonsense Knowledge Bases (KBs). We investigate the impact of weak supervision by creating a weakly supervised dataset and show that even a slight amount of supervision can significantly improve the model performance and enable better-quality transfers. We examine different model architectures, and evaluation metrics, proposing a novel Commonsense KB completion metric tailored for generative models. Extensive experimental results show that the proposed method compares very favorably to the existing baselines. This approach is a viable step towards a more advanced system for automatic KB construction/expansion and the reverse operation of KB conversion to coherent textual descriptions.

1 Introduction

The automatic construction of Knowledge Bases (KBs) from text and the reverse operation of sentence generation from KBs are dual tasks that are both active research topics.

The first task of automatic KB construction remains a significant challenge due to the difficulty of detecting parts of text representing meaningful facts and summarizing them in a systematic form. A simpler sub-task of KB completion, i.e., extending or filling-in missing nodes or edges, has also attracted the attention of the research community. For both tasks, the system needs to generate new or complete existing graph entities coherently, possibly matching to the already existing graph structure. The dual task of decoding the information from KB back to text is a valuable functionality. This enables knowledge transfer from potentially large complex

---

* Equal contribution
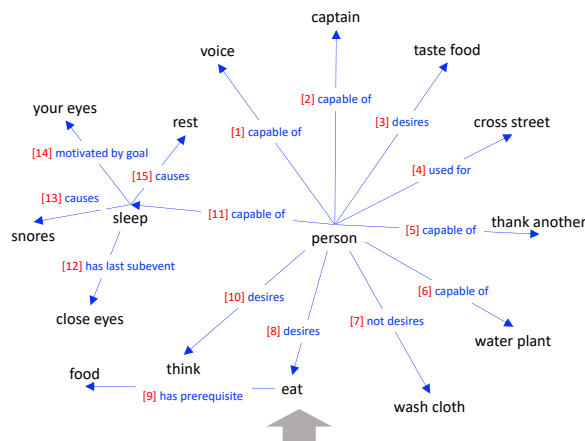
[1]   a person can voice an opinion   [2]   a person can captain a ship
[3]   a person can taste food   [4]   person can cross the street
[5]   person can thank another person   [6]   a person can water a plant
[7]   person can wash cloths   [8]   person can eat
[9]   you eat food   [10] a person can think
[11] a person can sleep
[12] if you want to sleep then you should close eyes
[13] sleep would make you want to snore
[14] if you want to sleep then you should close your eyes
[15] you sleep to rest.

Figure 1: Text to Path. Part of a larger graph generated from test sentences from our dataset. Sentences below the graph were a subset of inputs provided to the model.

graphs into a more descriptive, human-friendly output. This conditional generation is often seen as a step towards learning using KB as prior knowledge.

In this work, we address the problem of KB construction/completion and the reverse task of KB decoding, but aim at a simpler objective: transferring a single sentence to a path, and generating text from a single KB path as its dual task. In terms of data, our focus will be on Commonsense KBs, derived from sets of commonsense facts expressed in natural language sentences (Lenat, 1995; Cambria et al., 2014; Speer et al., 2017; Sap et al., 2019). They are represented as graphs where each edge is expressed as a tuple (e_h, r, e_t) with head and tail nodes e_h and e_t composed of free-form text, connected with a relationship operator r; see

ConceptNet from Speer et al. (2017) or ATOMIC in Sap et al., 2019 for recent and commonly used examples of commonsense KBs.

We observe that to train such transfer model, an additional challenge comes from the lack of datasets with parallel text and KB facts, i.e., where text sentences and KBs edges are explicitly paired/labeled from one to another. However, there exist many datasets for each individual transfer domain. Therefore, successful approaches transferring text to KB and KB to text must be able to operate in unsupervised or (at best) weakly-supervised settings. We address this challenge by proposing a model trained under dual learning of translation/transfer from text to KB and from KB to text, we name *DualTKB*. This is similar in philosophy to dual learning in Neural Machine Translation (He et al., 2016), or unsupervised style transfer (Shen et al., 2017; Tian et al., 2018; Dai et al., 2019). We design our model to be trained in completely unsupervised settings. However, we observed that even a slight supervision significantly boosts model performance and enables better-quality transfers. Therefore, we also describe a simple heuristic methodology to create weakly-supervised datasets given a text corpus and a commonsense KB.

We must emphasize that our proposed dual learning method is not limited to commonsense KBs and can generalize to other domains/types of KBs such as biomedical KBs. Commonsense KBs, and particularly ConceptNet, are good starting points due to the nature of their composition. Since ConceptNet was partly extracted from free-form text originating from the Open Mind Common Sense (OMCS) list of commonsense fact sentences, its nodes are often composed of parts of sentences from OMCS. This allowed us to first explore whether the proposed method worked at all before evaluating a semi-supervised approach by creating a weak supervision from a mapping between ConceptNet triples and the original OMCS sentences. While KBs are often dense with short named entity descriptions for nodes, many nodes for commonsense KBs are parts of sentences, making them inherently sparse which impacts their performance as empirically studied by Malaviya et al. (2020).

The evaluation of this type of transfer models is a challenge in itself. For this purpose, we selected a set of metrics to examine different facets of the system using our created weakly-supervised dataset.
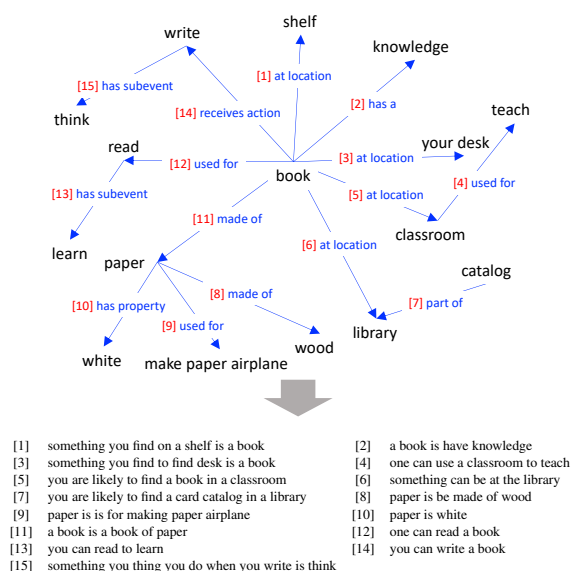

Figure 2: Path to Text. Sentences generated by our system from a subgraph of the ConceptNet dataset. The paths shown in the graph are the inputs to the model.

For path generation, we rely on a *conventional* KB completion task where the goal is to maximize the validity score of a tail entity e_t given the pair (e_h, r). For example, Malaviya et al. (2020) addresses the challenges unique to commonsense KB completion due to sparsity and large numbers of nodes resulting from encoding commonsense facts. However, KB completion does not always equate generation of edges, with the exception of COMET from Bosselut et al. (2019) that generates tail node e_t given the pair (e_h, r).

Since repurposing generative models for conventional KB completion evaluation is difficult (Malaviya et al., 2020; Bosselut et al., 2019), we propose a new commonsense KB completion evaluation task for generative models. It is close in spirit to conventional KB completion, but comes with its own set of challenges. Moreover, we employ the Graph Edit Distance (GED) to examine the quality of the generated graph as a whole. For text generation, we rely on traditional NLP metrics such as BLEU and ROUGE.

Following is a list of highlights of our paper contributions: (1) Propose a dual learning bridge between text and commonsense KB. Implement approach as unsupervised text-to-path and path-to-text transfers; (2) Construct a weakly-supervised dataset, and explore weak-supervision training. (3) Define a novel Commonsense KB completion metric tailored for generative models. (4) Investigate successfully multiple model architectures.

Finally, in Fig. 1 and Fig. 2 we present a few examples generated by our proposed model. Fig. 1 is a text to KB translation. Each sentence below the graph is independently transferred to a path, consisting of one edge tuple (e_h, r, e_t). The whole path tuple is generated at once, with e_h, r, and e_t taking a free-form not restricted to any predefined sets of entities or relations. This contrasts with many existing works operating on a limited discrete set of already-defined edges in a dense conventional KB. Once all the sentences are transferred, we observe that this set of generated edges forms a connected structure, implicitly merging some nodes to form a prototype of a Knowledge Graph (KG).

Fig. 2 shows the transfer from KB to text. Each path in the graph is converted to a sentence. There is overall diversity in the generated sentences styles. Moreover, the samples show that the generation process is more sophisticated than just a trivial path flattening (i.e., merging text from all edge parts followed by minimal edits). Therefore, the proposed approach can eventually become a part of a more sophisticated system converting graphs to a coherent textual story and vice versa. Additional examples are presented in Appendix A.4.

2 Dual Learning

In this work we propose to use a dual learning approach to build a model performing two distinct but complementary generative tasks described in Fig. 3, and using notations from Tab. 1:

Task 1 (text→path): Given a sentence x_A, generate a path x_{AB} with well-formed entities and relation, that can either belong to an already constructed KB, or extend it in a factually meaningful way. This conditional generation is framed as a translation task referred to as T_{AB} where $x_{AB} = T_{AB}(x_A)$.

Task 2 (path→text): Given a KB path x_B, generate a descriptive sentence x_A, coherently merging entities and relation from the path. This conditional generation is a translation task referred to as T_{BA}, where $x_{BA} = T_{BA}(x_B)$.

From the above two tasks follows the definition of *back-translation* tasks T_{BAB} (path→text→path) and T_{ABA} (text→path→text). Reconstruction tasks T_{AA} and T_{BB} are trivially defined as generating the same text/path from itself.

In an unsupervised setting, where sentences x_A and paths x_B are not paired, the reconstruction tasks T_{AA}, T_{BB} and back-translation tasks T_{ABA}, T_{BAB} are

x_S	sentence, where type S can be	
	A, A^m	sentence, masked sentence
	AA	given A, reconstructed sentence
	BA	given B, generated sentence
	ABA	given AB, back-translated sentence
x_P	path, where type P can be	
	B, B^m	path, masked path
	BB	given B, reconstructed path
	B^{m}B	given B^m, reconstructed path
	AB	given A, generated path
	BAB	given BA, back-translated B
w	edge $w = (e_h, r, e_t)$ with (entity, rel. op., entity)	
w^m	masked edge/tuple, e.g., $(e_h, r, .)$, $(., r, e_t)$, etc.	
T_D	Translation task with direction D	
T_{AB}	Translation from A to B. $x_{AB} = T_{AB}(x_A)$	
T_{BA}	Translation from B to A. $x_{BA} = T_{BA}(x_B)$	
T_{ABA}	Back-Translation from AB, back to A where $x_{ABA} = T_{BA}(x_{AB}) = T_{BA}(T_{AB}(x_A))$	
T_{BAB}	Back-Translation from BA to B where $x_{BAB} = T_{AB}(x_{BA}) = T_{AB}(T_{BA}(x_B))$	
T_{B^mB}	Generation from masked path $x_P = T_{B^mB}(x_P^m)$	

Table 1: Notations used throughout the paper.

the only ones available to define training losses. The back-translation tasks define the so-called cycle/consistency losses, which implicitly control the quality of the first transfers (T_{AB}, T_{BA}), by checking the reconstruction after the second transfers (T_{ABA}, T_{BAB}). By themselves, these cycle losses are effective in training good transfer model. However, as we show in Section 5, even a small weak supervision (by pairing some x_A to x_B) can significantly improve our model performance.

As shown in Fig. 3, our model uses an encoder-decoder architecture with a common encoder and two specialized decoders (A, generating sentences x_A, and B, generating paths x_B). The reason for a single encoder is to force the embeddings of path and text to lie in the same space, encoding a fact regardless of modality. It then becomes the job of each specific decoder to recover either one of the modalities. Note that the decoders share the same architecture, but not their parameters.

2.1 Un/Weakly-supervised Setups

For data source, we used the standard 600K set from Li et al. (2016) (CN-600K). It is a curated version of ConceptNet 5 (Speer et al., 2017), a well-known commonsense KB, which partly originated from the Open Mind Common Sense (OMCS) list of commonsense fact sentences. Despite no explicit pairing between sentences from OMCS and paths from CN-600K, both datasets cover a re-
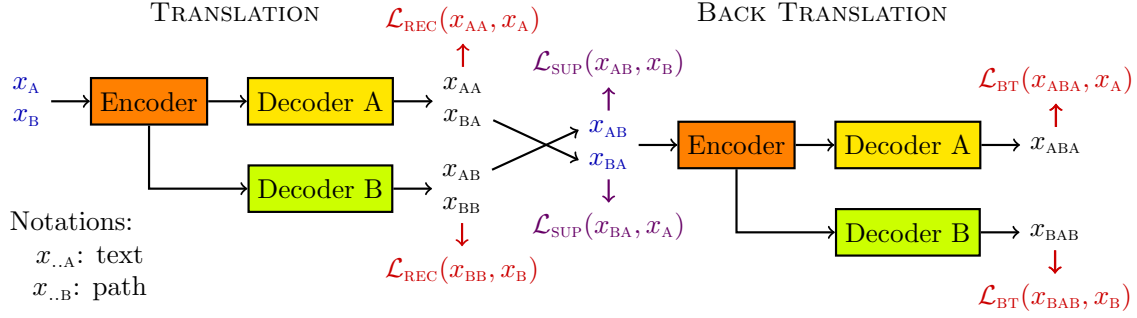
Figure 3: Our Encoder-Decoder model with translation and back-translation represented as a sequence of steps. Encoder and decoders are shared for both steps. The Encoder provides a representation of inputs x_A (text) and x_B (path) in a common embedding space. Decoders A and B are specialized to generate only sentences and paths (respectively) from these embeddings. Losses are indicated close to the generated values they require. Reconstruction loss $\mathcal{L}_{REC}$, back-translation loss $\mathcal{L}_{BT}$ are available in unsupervised learning, with $\mathcal{L}_{SUP}$ only available in supervised learning. Generation of x_{ABA} from x_A means two passes through our model: First pass, with x_A as input. Second pass, with $x_{AB} = T_{AB}(x_A)$. The same model can accommodate path or text generation.

lated set of facts making them good candidates for our unsupervised and weakly-supervised training. ATOMIC (Sap et al., 2019) was created from many data sources (books, n-grams, etc.) not easily accessible, leaving us with no relevant text corpora to pair with. Therefore, we only use CN-600K.

The weakly-supervised dataset was obtained by doing fuzzy matching of sentences x_A to paths x_B. Each sentence is mapped to a list of paths, and each path to a list of sentences. Note that KB and text set do not align exactly; noise and mislabelling are inherently present. Despite these constraints, we can investigate the effects of weak supervision on model performance, and vary the amount of it by changing the fraction of available paired data. More details about how this weakly-supervised dataset[1] was created are in Section 5.1.

3 Model

Given a dataset X of paths and sentences, let x_t^k be its k-th random sample of type t, for $k = 1, \ldots, N$ and type $t \in \{A, B\}$, where x_A is a sentence and x_B is a path in the KG. Given the input x_t^k to the model, the corresponding generated output will be denoted as $x_{tt'}^k$, where tt' is the transfer direction, i.e., $tt' \in \{AA, AB, BB, BA\}$. For example, given path x_B^k, $x_{BA}^k = T_{BA}(x_B^k)$ denotes the corresponding generated sentence. Similarly, given $x_{tt'}^k$ as input, $x_{tt't''}^k$ denotes additional possible transfer directions, out of which we will be only interested in $tt't'' \in \{ABA, BAB\}$, as they represent

the *back-translations* of a sample from a type t back to itself, since $t'' = t$. Given input sentence x_{BA}^k, x_{BAB}^k denotes its generated back-translation such that $x_{BAB}^k = T_{AB}(x_{BA}^k) = T_{AB}(T_{BA}(x_B^k))$. A model with perfect generation and back-translation would yield $x_{BAB}^k = x_B^k$ as $T_{AB}(T_{BA}(x))$ would be the identity function. Note that to reduce clutter, we drop the dataset index k from the notations.

3.1 Losses

There are three types of losses we employ in our training. The reconstruction loss is defined as

$$\mathcal{L}_{REC} = \mathop{\mathbb{E}}_{x_A \sim X} \left[-\log p_{AA}(x_A) \right] + \mathop{\mathbb{E}}_{x_B \sim X} \left[-\log p_{BB}(x_B) \right],$$

where $p_{AA}(x_A)$ is the distribution for the reconstructed sentences $x_{AA} = T_{AA}(x_A)$, and $p_{BB}(x_B)$ for paths $x_{BB} = T_{BB}(x_B)$. To enable model to perform transfers, we also employ the so-called back-translation, or cycle loss:

$$\mathcal{L}_{BT} = \mathop{\mathbb{E}}_{x_A \sim X} \left[-\log p_{ABA}(x_A | x_{AB}) \right]$$
$$+ \mathop{\mathbb{E}}_{x_B \sim X} \left[-\log p_{BAB}(x_B | x_{BA}) \right]. \quad (1)$$

Unsupervised training minimizes the combined loss $\mathcal{L}_{REC} + \mathcal{L}_{BT}$. When supervised data is available, we can additionally impose a supervision loss for the paired data:

$$\mathcal{L}_{SUP} = \mathop{\mathbb{E}}_{x_A, x_B \sim X} \left[-\log p_{AB}(x_B | x_A) \right]$$
$$+ \mathop{\mathbb{E}}_{x_A, x_B \sim X} \left[-\log p_{BA}(x_A | x_B) \right]. \quad (2)$$

Supervised training minimizes the combined loss $\mathscr{L}_{\text{REC}} + \mathscr{L}_{\text{BT}} + \mathscr{L}_{\text{SUP}}$. We explore the impact of $\mathscr{L}_{\text{REC}}$, $\mathscr{L}_{\text{BT}}$, and $\mathscr{L}_{\text{SUP}}$ on training with an ablation study detailed in Section 5.3. Note that when the supervised data is limited, i.e., the sentence-path correspondence is only known for some pairs $(x_{\text{A}}, x_{\text{B}})$, the loss in (2) is adjusted accordingly and averaged only over the available pairs. As we show in Section 5.4, this modification enables us to examine the effect of the amount of supervision on the model performance.

4 Experimental Setup

In this Section, we provide details about model architecture, training procedure, and data processing.
Model Architectures. We explored several architectures for our encoder and decoder: GRU (Chung et al., 2014), Transformer (Vaswani et al., 2017), and BERT (Devlin et al., 2019).

All possible pairings of encoder and decoder, provide a total of nine different combinations. Some of them, such as Transformer-BERT, GRU-BERT, or BERT-BERT are not valid, since BERT can only serve as an encoder model. Other configurations were rejected based on their performance on our validation set. In particular, we found that GRU-Transformer and BERT-Transformer just did not perform well, setting them aside. In the end, we selected the following three combinations: (1) GRU-GRU, since GRU is a simple, well-known RNN architecture, relatively easy and fast to train; (2) BERT-GRU, selected to leverage potentially better encoding representation from BERT, a well-known encoder; (3) Transformer-Transformer (or Trans-Trans), chosen to explore another seq2seq architecture. All models parameters are trained, while for BERT we fine-tune the pre-trained model.
Teacher Forcing. During training, in the recurrent generation steps, we employ teacher forcing by feeding the ground truth tokens only 20% of the time, using the model output otherwise. We found this strategy very effective at preventing overfitting and increasing the overall generation quality.
Back-Translation. Traditionally, when using the back-translation loss in (1), gradients are back-propagated through the encoder-decoder structure twice to reflect the two transfers being done (e.g., $x_{\text{A}} \rightarrow x_{\text{AB}} \rightarrow x_{\text{ABA}}$). However, we observed better training behavior and model performance by detaching the gradients after the first pass through the system. The model still sees both back-

propagations but the training becomes more stable.
Data processing. In our dataset, a path such as $(e_h, r, e_t) = ($"yeast", "is a", "ingredient in bread"$)$ will be encoded as "[SEP] yeast [SEP] is a [SEP] ingredient in bread [SEP]", including the special token [SEP], as done in Yao et al. (2019). Similar to Devlin et al. (2019), we mask a token by replacing it with a special token [MASK]. For a path, we replace either the head/tail by the mask token 50% of the time, e.g., $($"[MASK]"$, r, e_t)$ for head masking. For text, we replace a randomly picked token with the mask token 10% of the time. For more robust training, we mask tokens in both text and paths input sequences and reconstruct the original. This is in the same spirit as recently proposed masked language modeling techniques (Devlin et al., 2019; Liu et al., 2019). While this technique helps learning robust encoder-decoder, it also inherently prepares the model for the link-predictions of KB completion evaluation.

5 Results and Discussions

In this Section we discuss in detail our dataset construction, evaluate the proposed models, and compare performances to existing baselines.

5.1 Datasets

We designed our own dataset due to the lack of available resources of paired text corpus and commonsense KBs. For KB, we started from CN-600K (Li et al., 2016) derived from ConceptNet (Speer et al., 2017). As mentioned earlier, we define a path as composed of two nodes and a directed edge connecting them. We only keep paths from CN-600K with confidence score greater than 1.6 for a total of 100K edges. Sentences in our dataset come directly from OMCS free-text sentences, removing those with detectable profanity.

Note that as is, this dataset can only be used for unsupervised training since its path-sentence pairs are not matched. To enable weak supervision, we map paths to sentences (and sentences to paths) using fuzzy matching directly on their sequence of tokens. Fuzzy matching uses term-frequency (tf) and inverse document frequency (idf) with n-grams to compute a cosine similarity score between two sets of strings (van den Berg, 2017).

Only the top matching pair is kept if its similarity score is greater than 0.6 to ensure a minimum quality for our matches. This score of 0.6 (from a range of -1 to 1) was chosen empirically after notic-

Conventional KB Completion	MRR	HITS@1	HITS@3	HITS@10	GED↓
DISTMULT (Yang et al., 2015)	8.97	4.51	9.76	17.44	-
COMPLEX (Trouillon et al., 2016)	11.40	7.42	12.45	19.01	-
CONVE (Dettmers et al., 2018)	20.88	13.97	22.91	34.02	-
CONVTRANSE (Shang et al., 2019)	18.68	7.87	23.87	38.95	-
S+G+B+C (Malaviya et al., 2020)	51.11	39.42	59.58	73.59	-
Generative KB Completion					
DUALTKB$_{\text{GRU-GRU}}$, $\rho = 0.5$	**63.10**	**55.38**	**69.75**	**74.58**	12.5
DUALTKB$_{\text{BERT-GRU}}$, $\rho = 0.2$	61.32	53.79	67.62	72.29	12.0
DUALTKB$_{\text{Trans-Trans}}$, $\rho = 0.5$	50.54	44.54	55.12	59.67	10.0
DUALTKB$^{*}_{\text{GRU-GRU}}$, $\rho = 0.5$	50.87	44.58	55.46	60.12	9.0
DUALTKB$^{*}_{\text{BERT-GRU}}$, $\rho = 0.5$	57.79	50.25	63.75	69.54	11.0
DUALTKB$^{*}_{\text{Trans-Trans}}$, $\rho = 1.0$	40.93	35.67	44.38	48.79	**8.0**

Table 2: Conventional and generative KB completion results for ConceptNet test set for (filtered) MRR, HITS and GED metrics. Models in training w/ best MRR evaluations were selected for testing. Models with asterisk $*$ were selected based on best BLEU2 score for BA text generation task. Supervision ratio is indicated as ρ. Results from conventional KB completion methods quoted at the top of the table are as reported in their respective papers.

	B_2	B_3	R_L	B_{F1}
DUALTKB$_{\text{GRU-GRU}}$	0.32	0.24	0.46	0.89
DUALTKB$_{\text{BERT-GRU}}$	0.32	0.25	0.46	0.88
DUALTKB$_{\text{Trans-Trans}}$	0.45	0.37	0.56	0.91
DUALTKB$^{*}_{\text{GRU-GRU}}$	**0.49**	**0.42**	**0.61**	**0.92**
DUALTKB$^{*}_{\text{BERT-GRU}}$	0.37	0.30	0.51	0.89
DUALTKB$^{*}_{\text{Trans-Trans}}$	0.47	0.39	0.57	0.91

Table 3: BA text generation evaluation results for BLEU2 (B_2), BLEU3 (B_3), RougeL (R_L), and F1 BERT-score (B_{F1}). Models correspond to the ones in Tab. 2. Models with $*$ are selected by best B2 scores.

ing that the path-sentence pairs quality degraded quickly for lower scores, as some sentences did not have a good match. Indeed, *not all* OMCS sentences were used to create ConceptNet.

We create dataset splits *train* (240K) and *dev* (10K) under the strict policy that no path nor sentence in dev can exist in train, to ensure validation on text and path unseen in training.

During training, we can vary the amount of weak supervision available by applying the supervision loss $\mathscr{L}_{\text{SUP}}$ in (2) to a subset of the data. Note that for the pairs where $\mathscr{L}_{\text{SUP}}$ is not applied, the training becomes unsupervised since the remaining losses $\mathscr{L}_{\text{REC}} + \mathscr{L}_{\text{BT}}$ cannot exploit the presence of a match. Therefore, by changing the size of the subset to which $\mathscr{L}_{\text{SUP}}$ is applied we can go from full weak-supervision to unsupervised training.

Evaluations are reported on 1200 positive tuples of test set from the original split in Li et al. (2016) as done in prior works on commonsense KB (Malaviya et al., 2020). Irrespective of the choice of encoder, we tokenize the entire dataset (paths and sentences) using the BERT tokenizer.

5.2 Metrics and Results

The challenge in evaluating our models is that they accomplish different generation tasks, each with its own adequate metrics. Detecting an overall best model is therefore ambiguous and should be potentially application specific. Nevertheless, we present a set of metrics and results for all the important generation tasks for three model encoder-decoder architectures: GRU-GRU, BERT-GRU, and Trans-Trans, as well as discuss the model selection process. Results are obtained over 40 epochs of training. We present both the best results for each metric, and their averages over 4 models trained with distinct random seeds in Appendix in Tab. 6. Note that we refer to a generation task simply by its direction, i.e., use ABA for T_{ABA}.

AA, ABA: The performance on reconstruction task AA is indicative of the quality of autoencoding, while the back-translation task ABA, besides reconstruction also helps in evaluating implicitly the quality of transfer. We use the following metrics to evaluate generated sentence quality: BLEU2 (B2), BLEU3 (B3), ROUGEL (R_L), and F1 BERT-scoring metric (B_{F1}) (Zhang et al., 2019) in Tab. 5

Losses	DualTKB$_{\text{GRU-GRU}}$		DualTKB$_{\text{BERT-GRU}}$		DualTKB$_{\text{Trans-Trans}}$	
	MRR	BLEU2	MRR	BLEU2	MRR	BLEU2
$\mathscr{L}_{\text{REC}} + \mathscr{L}_{\text{BT}} + \mathscr{L}_{\text{SUP}}$	63.10	0.32	61.32	0.32	50.53	0.45
$\mathscr{L}_{\text{BT}} + \mathscr{L}_{\text{SUP}}$	17.09	0.48	0.14	0.07	45.42	0.46
$\mathscr{L}_{\text{BT}} + \mathscr{L}_{\text{REC}}$	20.08	0.25	52.52	0.03	23.56	0.26
$\mathscr{L}_{\text{REC}} + \mathscr{L}_{\text{SUP}}$	46.16	0.46	57.57	0.34	44.08	0.42

Table 4: Ablation study on different set of losses across various models

in Appendix. We observed that reconstruction AA is almost perfect in all metrics, while ABA is lagging as it must do first the translation and then recover back the sentence, a more challenging task. GRU-GRU is not as good for ABA as for AA, while Trans-Trans provides strong ABA results. BERT-GRU is ahead of the pack for ABA. Overall the models can handle AA and ABA reasonably well.

B$_m$B: KB Completion: KB completion is a common task for comparing models. Previous work established link prediction as the standard for evaluation relying on MRR (Mean Reciprocal Rank) and HITS as evaluation ranking metrics (Yang et al., 2015; Dettmers et al., 2018; Bordes et al., 2013). For tail prediction, e_t in all test tuples (e_h, r, e_t) are replaced by other nodes $\tilde{e}_t$ to produce corrupted tuples. A validity score is evaluated for all new corrupted tuples $(e_h, r, \tilde{e}_t)$, and a ranking is provided for the tuple with ground truth e_t. Head prediction is done similarly, often by reversing the relation r to r^- and ranking e_h given (e_t, r^-). As in Bordes et al. (2013), filtered versions of MRR and HITS are preferred as they filter out corrupted tuples present in training, dev, and test sets. We use filtered MRR and HITS for all results in this paper. We are aware of the recent critic of this KB completion evaluation by Akrami et al. (2020). However, since the community is yet to establish another methodology, and to be able to relate to previous work, we follow this procedure, albeit with a twist. For commonsense KB, Malaviya et al. (2020) mention the difficulty of repurposing generative models to ranking tuples for link-prediction. Our model is trained to generate a path tuple, not evaluate the quality score of its validity. Entities, relationship are all sequences of tokens that need to be generated properly, which does not fit well in the *conventional* KB completion evaluation framework. Therefore, we define a new meaningful commonsense KB completion task for generative models, and present the challenges that arise from it.

Since our models are trained from masked inputs, they can easily generate a new path $\hat{x}_{\text{B}} = T_{\text{B}^m\text{B}}(x^m_{\text{B}})$ from masked tuples $x^m_{\text{B}} = (e_h, r, .)$, or $x^m_{\text{B}} = (., r, e_t)$. This B^{m}B generation task can be done from any masked tuple, allowing for tail and head predictions without resorting to inverting the relationship r. For both predictions, the model generates a *complete* edge, not just e_h or e_t as for generative model COMET (Bosselut et al., 2019). However, the generated edge may not be a proper sequence of tokens for a (e_h, r, e_t) tuple, especially in early stages of training where the model is still learning the tuple structure. For improper generated tuples, tail and head predictions become impossible as the generated sequences of tokens cannot parse into correctly-formed tuples. For these cases, a worst case ranking score is given since no meaningful ranking is achievable, while their rankings are still required by MRR and HITS metrics. Our results are therefore *very* sensitive to the correct form of generated paths.

To summarize, for commonsense KB completion, we follow these steps: (1) Provide an input triple w/ masked entity (tail/head); (2) Generate a complete triple from our model (including a candidate for the missing/masked entity); (3) Score this predicted triple along with all corrupt triples (where the masked entity was replaced with all possible node candidates from our data) using fuzzy matching against the ground truth triple; (4) Remove corrupt triples before scoring if they already exist in train, test, or dev (filtering).

Defining a scoring function brings also another challenge. Indeed, even when producing proper tuples from masked entries, our generative model can still change some tokens in the unmasked part of the input tuple, i.e., for $x^m_{\text{B}} = (e_h, r, .)$, we could get a generated $\hat{x}_{\text{B}} = T_{\text{B}^m\text{B}}(x^m_{\text{B}})$ with $\hat{x}_{\text{B}} = (\hat{e}_h, \hat{r}, \hat{e}_t)$, where $e_h \neq \hat{e}_h$ and/or $r \neq \hat{r}$. There is no guarantee that the original pair (e_h, r) remains identical. The solution of using the origi-

nal (e_h, r) for scoring, regardless of $(\hat{e}_h, \hat{r})$, is not fair as it will provide better performance than really achieved by the generative model. Therefore, for scoring function, we decided to compute the ranking scores using fuzzy matching of corrupted tuples, using the *whole* generated tuple sequence, taking into account any potential flawed generation of $(\hat{e}_h, \hat{r})$. MRR and HITS computation is exactly the same as in conventional KB completion.

We provide results for our KB completion in Tab. 2 where we added results from previous related work. During training, if models with the best MRR evaluation results are selected and used for testing, then GRU-GRU shows overall better performance. However, if BLEU2 metric is used for the selection, then BERT-GRU achieves higher scores. We observed that the overall performance of Trans-Trans was behind in these evaluation metrics. It is to be expected that not all metrics are strongly correlated and peak at the same time during training. It also confirms that for model selection, a weighted composite of all metrics would be a better approach.

BA: Sentence evaluation: The generation of sentence from path is another important task. For qualitative results, examples of such generation are shown in Fig. 2 and in Appendix A.4. For quantitative results, we evaluated the sentences using B2, B3, R_L, and B_{F1} metrics, as reported in Tab. 3 for the same set of models as in Tab. 2. We compute these metrics against our weakly-supervised data, created using fuzzy matching. Selecting models solely on MRR has its shortcomings for other tasks. Models selected on MRR have relatively decent performances on KB completion metrics, but lag behind when compared to the models selected under text evaluation metrics. GRU-GRU and Trans-Trans are particularly good at this task, while BERT-GRU is in third place.

AB: Graph Edit Distance: In contrast to the single-instance evaluation, examining each generated path independently from others, we propose now to look at the generated graph as a whole and compare it to the ground truth one (based on our weakly-supervised dataset). In other words, given input sentences x_A, we generate the corresponding paths x_{AB} and compare them to the ground truth x_B. One of the metrics to compute graph similarity is Graph Edit Distance (GED) (Chen et al., 2019), which finds minimum cost path (consisting of node/edge substitutions, deletions, and
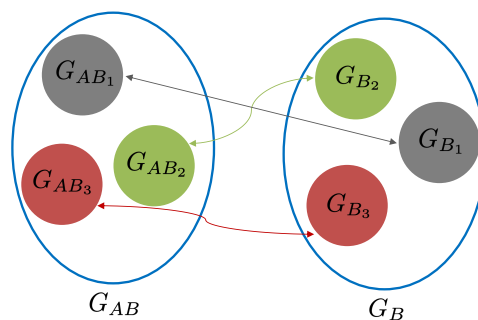

Figure 4: GED computation based on local subgraphs. GED is computed as an average of corresponding local graphs. As the size of local graph increases, the computed value better approximates the global GED. In our experiments we used subgraphs consisting of 10 paths due to computational cost. Larger numbers of paths result in prohibitive GED computation cost.

insertions) transforming one graph into another. Since in general exact GED computation is NP-hard and practically infeasible for large graphs, we propose an approximation based on local subgraphs, as illustrated in Fig. 4. To define the cost of matching nodes (corresponding to heads and tails) and arcs (corresponding to relationship operators), we encode them into feature vectors using BERT and compare the value of their euclidean distance to a predefined threshold to identify matching nodes/edges. GED values are reported in Tab. 2 for all of our models. Trans-Trans has the lowest GED for all cases.

5.3 Ablation Study of Losses

In our ablation study, we evaluate the effect of each loss $\mathscr{L}_{REC}$, $\mathscr{L}_{BT}$, and $\mathscr{L}_{SUP}$ on the overall model performance. As seen in Tab. 4, by removing any of the losses, the MRR/HITS performance drops compared to the full loss case. At the same time, for different models, each loss has its own impact: e.g., $\mathscr{L}_{BT}$ has more value for GRU-GRU model, while the availability of weak supervision $\mathscr{L}_{SUP}$ is more important for Trans-Trans architecture. Another interesting conclusion is that although $\mathscr{L}_{SUP}$ explicitly and $\mathscr{L}_{BT}$ implicitly both control the quality of transfers (T_{AB} and T_{BA}), they remain complementary to each other, i.e., there is still a benefit of using both in the same loss.

5.4 Impact of Supervision Ratio

The impact of weak-supervision on model performance is illustrated in Fig. 5. For all the models, even a slight supervision amount leads to a
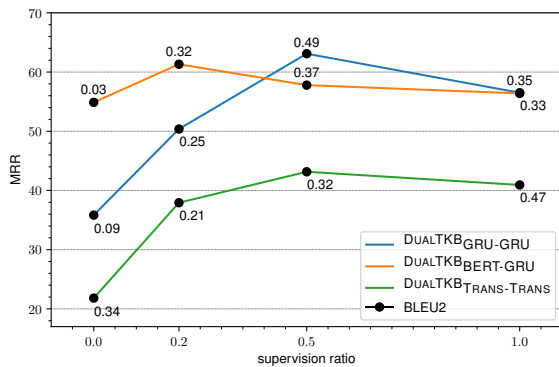
Figure 5: Models MRR and BLEU2 performance for supervision ratio 0, .2, .8 and 1.0. After gains from slight supervision, full supervision degrades performances, except for BLEU2 for Trans-Trans models.

rapid improvement for both MRR and BLEU2 metrics, which vindicates our decision to use weak-supervision. Importantly, model performance tends to peak before reaching full supervision: only our Trans-Trans model still sees an improvement for B2 scores at full supervision, while both GRU-GRU and BERT-GRU trained at 50% weak-supervision lead to better models. This can be explained by the nature of weak supervision itself. Fuzzy matching relies only on tf-idf and n-grams, which is not perfect when dealing with paths and sentences with different grammar structures. This procedure is inherently noisy and can pair sentences and paths with related but, nevertheless, not exact semantic matches. The full weak-supervision usually brings noise into the training, thus harming it more than helping. This conclusion is confirmed by the lower performance of our models at full supervision.

6 Conclusion

In this paper we proposed to use a dual learning bridge between text and commonsense KB. In this approach, a generative model is trained to transfer a sentence to a path and back. Assembling paths together results in a graph showing the presence of inherent structure, while generated sentences exhibit coherent and relevant semantics. For evaluation, we proposed a novel commonsense KB completion task tailored to generative models. Although our model is designed to work in unsupervised settings, we investigated the impact of weak-supervision by creating a weakly-supervised dataset and showed that even a slight amount of weak-supervision improves significantly model performance. The current work is one step towards the overarching goal

of KB construction/completion *and* generation of human-readable text from KBs. Future work can focus on expanding the capabilities to generating whole paragraphs of text from graphs in KB, as well as converting large parts of text into coherent graph structures.

References

Farahnaz Akrami, Mohammed Samiul Saeef, Qingheng Zhang, Wei Hu, and Chengkai Li. 2020. Realistic Re-evaluation of Knowledge Graph Completion Methods: An Experimental Study. *arXiv preprint arXiv:2003.08001*.

Chris van den Berg. 2017. Super Fast String Matching in Python.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-Relational Data. In *Advances in Neural Information Processing Systems*, 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. *arXiv preprint arXiv:1906.05317*.

Erik Cambria, Daniel Olsher, and Dheeraj Rajagopal. 2014. SenticNet 3: A Common and Common-Sense Knowledge Base for Cognition-Driven Sentiment Analysis. In *Twenty-eighth AAAI conference on artificial intelligence*.

Xiaoyang Chen, Hongwei Huo, Jun Huan, and Jeffrey Scott Vitter. 2019. An Efficient Algorithm for Graph Edit Distance Computation. *Knowledge-Based Systems*, 163:762–775.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 2019. Style Transformer: Unpaired Text Style Transfer without Disentangled Latent Representation. *arXiv preprint arXiv:1905.05621*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

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*, pages 4171–4186.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual Learning for Machine Translation. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 820–828. Curran Associates, Inc.

Douglas B Lenat. 1995. CYC: A Large-Scale Investment in Knowledge Infrastructure. *Communications of the ACM*, 38(11):33–38.

Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense Knowledge Base Completion. In *Proceedings of the 54th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 1445–1455. Association for Computational Linguistics.

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. *arXiv preprint arXiv:1907.11692*.

Chaitanya Malaviya, Chandra Bhagavatula, Antoine Bosselut, and Yejin Choi. 2020. Commonsense Knowledge Base Completion with Structural and Semantic Context. *Proceedings of the 34th AAAI Conference on Artificial Intelligence*.

Maarten Sap, Ronan LeBras, 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 *AAAI*.

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 *AAAI*.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style Transfer from Non-Parallel Text by Cross-Alignment. In *Advances in Neural Information Processing Systems*, pages 6830–6841.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Proc. of AAAI*, pages 4444–4451.

Youzhi Tian, Zhiting Hu, and Zhou Yu. 2018. Structured Content Preservation for Unsupervised Text Style Transfer. *arXiv preprint arXiv:1810.06526*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *ICML*, pages 2071–2080.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for Knowledge Graph Completion. *arXiv preprint arXiv:1909.03193*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. BERTscore: Evaluating Text Generation with BERT. *arXiv preprint arXiv:1904.09675*.

A Supplementary

A.1 Experimental Setup and Training

All our models were built using an NVIDIA V100 GPU, while validation was done with a CPU-only setup (using 2 cores). Each training epoch (240K samples) took between 1 hour for GRU-GRU (fastest), just over 1 hour for BERT-GRU, and 1.5h for Trans-Trans models (slowest). Our validation (10K samples) takes about 1.5 hours to evaluate as we need 2 passes for MRR and HITS (head and tail predictions), as well as a third pass for all NLP metrics for AA, ABA, and BA. Evaluation on our test set (1.2K samples) takes a matter of about 6 minutes for each head and tail prediction for MRR and HITS, with a third pass for NLP metrics, for a total of about 20 minutes. BLEU2 evaluation is relatively fast. GED evaluation can take up to 40 minutes. During training, we evaluate models at the end of each epoch, and use the best model over 40 epochs for testing.

All our models were built using PyTorch. They are trained with a batch size of 32 for a supervision ratio of 0.5. Masking for sentences is performed with a probability of 10% for each token while masking for paths tuples entities are set at 50% (when selected, all tokens of an entity are masked).

GRU models are trained with a learning rate of either 10^{-3} or 5×10^{-3} (10^{-4} yielded poor results) Transformer models were trained with a predefined learning rate schedule called "NoamOpt" providing a warm-up phase up to 20K training mini-batches before a slow exponential decay. All our Transformer models had three sub-layers with three heads in the multi-head attention, while GRU models had a hidden size of 100 with a single recurrent layer.

All trainings had a seeded random number generator to ensure repeatability. For instance, results from Tab. 5 were obtained for one common seed

for all models. For results in Tab. 6, 4 distinct seeds were used to show performance based on the average of the 4 individual model performances (standard deviation is also provided).

MRR and HITS metrics are defined in details in Dettmers et al. (2018) – code associated with the paper provides implementations in Python. We reimplemented them for better integration in our own codebase. BLEU metrics were used from NLTK implementations.

When running multiple supervision ratio for weak-supervision, hyper-parameters were all fixed to the values provided above, only the supervision ratio was changed.

## A.2 AA, ABA: Reconstruction and Back-Translation Tasks

Tab. 5 show results for our models GRU-GRU, GRU-BERT, and Trans-Trans for BLEU2, BLEU3, ROUGE, and F1 BERT score for both AA and ABA tasks. AA is an easier task with excellent while ABA is more difficult. GRU-GRU and Trans-Trans excel at both AA and ABA.

A.3 Results for Multiple Random Generation Seeds

In Tab. 6, we present results for our three model architectures using MRR-HITS and BLEU metrics as averages over 4 models for each architecture built with distinct random generator seeds, trained with our default hyper-parameters described in Section A.1. We report average and standard deviation for every metrics.

A.4 AB: Graph Generation

In this Section we present additional examples of the text to path transfers, see Figures 6, 7 and 8, as well as the reverse transfer from path to text, see Figure 9.
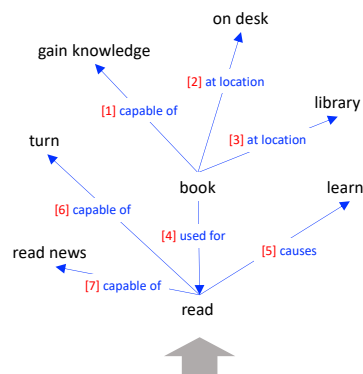
[1]	book can store knowledge
[2]	a something you find on your desk is a book
[3]	a book is part of library
[4]	book is to read
[5]	to read is to learn
[6]	a reader can turn a page
[7]	you can read the news in a newspaper

Figure 6: Text to Path. A part of a larger graph generated by our system based on the test split of ConceptNet dataset. The shown sentences are a subset of the inputs provided to the model.
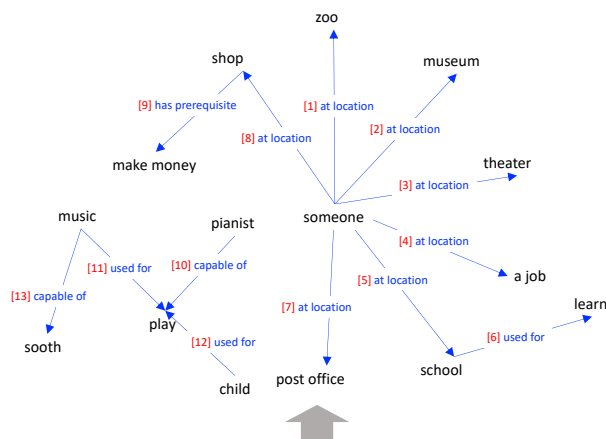
[1]	something you find at a zoo is an animal
[2]	someone can be at museum
[3]	someone can be at the theater
[4]	someone can be at a job interview
[5]	something can be at school
[6]	in school , you can learn
[7]	someone can be at a post office
[8]	someone can shop
[9]	if you want to shop then you should have money
[10]	pianist play piano
[11]	play music
[12]	a child can play
[13]	music can soothe

Figure 7: Text to Path. A part of a larger graph generated by our system based on the test split of ConceptNet dataset. The shown sentences are a subset of the inputs provided to the model.

	AA				ABA			
	BLEU2	BLEU3	R_L	$BERT_{F1}$	BLEU2	BLEU3	R_L	$BERT_{F1}$
DUALTKB$_{GRU\text{-}GRU}$	0.97	0.96	0.98	1.00	0.54	0.48	0.69	0.93
DUALTKB$_{BERT\text{-}GRU}$	0.71	0.64	0.74	0.93	0.53	0.45	0.63	0.91
DUALTKB$_{Trans\text{-}Trans}$	0.95	0.94	0.96	0.99	0.57	0.50	0.69	0.92

Table 5: Results for text generation AB and ABA evaluation for BLEU2 (B_2), BLEU3 (B_3), Rouge-L (R_L), BERT Score $BERT_{F1}$

	MRR	HITS@1	Hits@3	HITS@10	BLEU2	BLEU3
DUALTKB$_{GRU\text{-}GRU}$	51.13±5.59	44.78±5.12	55.89±6.38	60.68±6.38	0.52±0.07	0.41±0.08
DUALTKB$_{BERT\text{-}GRU}$	57.49±4.29	50.05±3.80	63.59±4.95	68.57±4.90	0.46±0.02	0.34±0.01
DUALTKB$_{Trans\text{-}Trans}$	43.87±4.65	37.57±4.93	47.80±5.02	54.02±4.11	0.48±0.08	0.38±0.05

Table 6: Results (Mean±SD) of various models across 4 different seeds for our random number generator. MRR, HITS metrics corresponds for KB completion task, whereas BLEU scores are shown for BA text generation task

[1] a human can eat   [2] if you want to sate your hunger then you should eat
[3] monkey like to eat banana   [4] a person can eat
[5] a pig can eat   [6] a mouse can eat
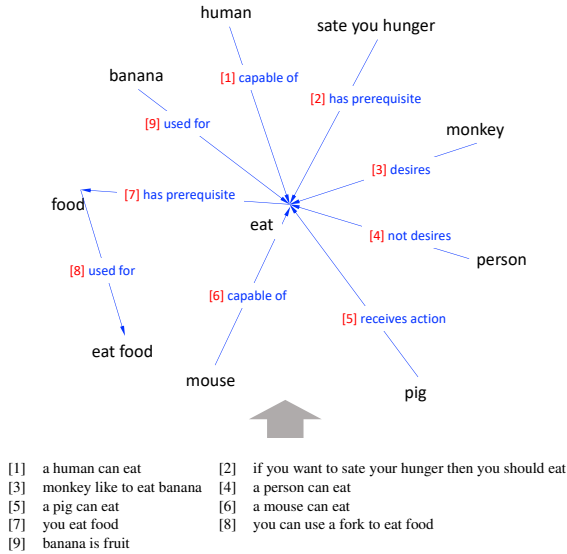[7] you eat food   [8] you can use a fork to eat food
[9] banana is fruit

Figure 8: Text to Path. A part of a larger graph generated by our system based on the test split of ConceptNet dataset. The shown sentences are a subset of the inputs provided to the model.

[1] a tomatoes is a kind of vegetable   [2] a vegetable is a kind of food
[3] chicken is a kind of food   [4] soup is a kind of food
[5] an egg is a kind of food   [6] something you find on table is food
[7] table is to put things on   [8] something you find at the supermarket is food
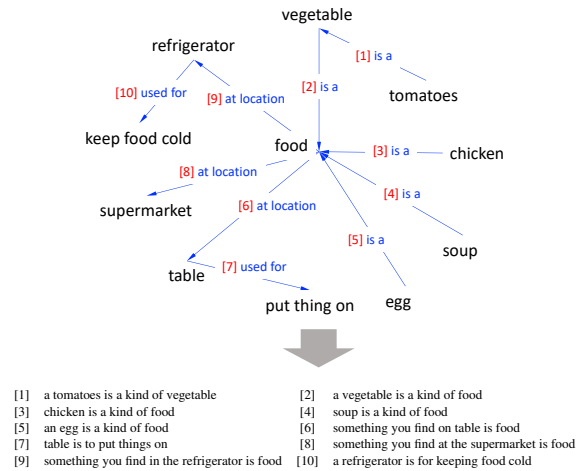[9] something you find in the refrigerator is food   [10] a refrigerator is for keeping food cold

Figure 9: Path to Text. A set of sentences generated by our system from a subgraph of the ConceptNet dataset. The paths shown in the graph are the inputs to the model.