Generating Commonsense Explanation by Extracting Bridge Concepts from Reasoning Paths

Haozhe Ji[1], **Pei Ke**[1], **Shaohan Huang**[2], **Furu Wei**[2], **Minlie Huang**[1]*
[1]Department of Computer Science and Technology, Institute for Artificial Intelligence,
State Key Lab of Intelligent Technology and Systems,
Beijing National Research Center for Information Science and Technology,
Tsinghua University, Beijing 100084, China
[2]Microsoft Research
{jhz20, kp17}@mails.tsinghua.edu.cn, {shaohanh, fuwei}@microsoft.com,
aihuang@tsinghua.edu.cn

Abstract

Commonsense explanation generation aims to empower the machine's sense-making capability by generating plausible explanations to statements against commonsense. While this task is easy to human, the machine still struggles to generate reasonable and informative explanations. In this work, we propose a method that first extracts the underlying concepts which are served as *bridges* in the reasoning chain and then integrates these concepts to generate the final explanation. To facilitate the reasoning process, we utilize external commonsense knowledge to build the connection between a statement and the bridge concepts by extracting and pruning multi-hop paths to build a subgraph. We design a bridge concept extraction model that first scores the triples, routes the paths in the subgraph, and further selects bridge concepts with weak supervision at both the triple level and the concept level. We conduct experiments on the commonsense explanation generation task and our model outperforms the state-of-the-art baselines in both automatic and human evaluation.[1]

1 Introduction

Machine commonsense reasoning has been widely acknowledged as a crucial component of artificial intelligence and a considerable amount of work has been dedicated to evaluate this ability from various aspects in natural language processing (Levesque et al., 2011; Talmor et al., 2018; Sap et al., 2019). A large proportion of existing tasks frame commonsense reasoning as multi-choice reading comprehension problems, which lack direct assessment to machine commonsense (Wang et al., 2019) and impede its practicability to realistic scenarios (Lin et al., 2019b). Recently, Wang et al. (2019) proposed a commonsense explanation generation challenge that directly tests machine's sense-making capability via commonsense reasoning. In this paper, we focus on the challenging explanation generation task where the goal is to generate a sentence to explain the reasons why the input statement is against commonsense, as shown in Figure 1.

Statement: The *school* was open for *summer*.
Explanation: *Summertime* is typically *vacation* time for *school*.

Figure 1: Generating a reasonable and informative explanation involves generating *bridge concepts* like *vacation* by identifying the relation to the *source concepts*, i.e. *school* and *summer* in the statement.

Generating a reasonable explanation for a statement faces two main challenges: 1) **Trivial and uninformative explanations**. As this task can be formulated as a sequence-to-sequence generation task, existing neural language generation models tend to generate trivial and uninformative explanations. For example, one of the existing neural models generates an explanation "*The school wasn't open for summer*" to the statement in Figure 1. Although it is sometimes reasonable, simple modification of the statement to the negation form with no additional information cannot explain the reasons why the statement conflicts with commonsense. 2) **Noisy commonsense knowledge grounding**. It's still challenging for most existing language generation models to generate explanations that are faithful to commonsense (Lin et al., 2019b). Thus, explicitly incorporating external knowledge sources is necessary for this task. Since the nature of the explanation generation task involves using underlying commonsense knowledge to explain, locating useful commonsense knowledge from large-scale knowledge graph is not trivial and generally requires multi-hop reasoning.

* Corresponding author
[1]The source code is available at https://github.com/cdjhz/CommExpGen.

Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 248–257
December 4 - 7, 2020.

To address the above challenges, we propose a two-stage generation framework that first extracts the critical concepts served as *bridges* between the statement and the explanation from an external commonsense knowledge graph, and then generates plausible explanations with these concepts. We first retrieve multi-hop reasoning paths from ConceptNet (Speer et al., 2017) and heuristically prune the paths to maintain the coverage to plausible concepts while keeping the scale of the subgraph tractable. Before the extraction stage, we initialize the representation of each node on the subgraph by fusing both the contextual and graph information. Then, we design a bridge concept extraction model that scores triples, propagates the probabilities along multi-hop paths to the connected concepts and further extracts plausible concepts. In the second stage, we use a pre-trained language model (Radford et al., 2019) to generate the explanation by integrating both the statement and the extracted concept representations. Experimental results show that our framework outperforms knowledge-aware text generation baselines and GPT-2 (Radford et al., 2019) in both automatic and human evaluation. Particularly, our model generates explanations with more informative content and provides reasoning paths on the knowledge graph for concept extraction.

To summarize, our contributions are two-fold:

- We analyze the under-explored commonsense explanation generation task and investigate the challenges in incorporating external knowledge graph to aid the generation problem. To the best of our knowledge, this is the first work on generating explanations for counter-commonsense statements.
- We propose a two-stage generation method that first extracts the bridge concepts from reasoning paths and then generates the explanation based on these concepts. Our model outperforms state-of-the-art baselines on the commonsense explanation generation task in both automatic and human evaluation.

2 Related Work

2.1 Machine Commonsense Reasoning

Previous work on machine commonsense reasoning mainly focuses on the tasks of inference (Levesque et al., 2011), question answering (Talmor et al., 2018; Sap et al., 2019) and knowledge base completion (Bosselut et al., 2019). While the ultimate goals of these tasks are different from ours, we argue that performing explicit commonsense reasoning is also critical to generation. A line of work (Bauer et al., 2018; Lin et al., 2019a) resorts to structured commonsense knowledge and builds graph-aware representations along with the contextualized word embeddings to tackle the commonsense question answering problem. In our work, we focus on reasoning over structured knowledge to explicitly infer discrete bridge concepts that are further used for text generation. Another line of work (Rajani et al., 2019; Khot et al., 2019) identifies the knowledge gap critical for the complete reasoning chain and fills the gap by writing general explanation or acquiring fine-grained annotations with human effort. While sharing a similar motivation, our method differs from theirs in the sense that we acquire distant supervisions for the bridge concepts to extract reasoning paths and generate plausible explanations without the need of additional human annotation.

2.2 Knowledge-Grounded Text Generation

Existing work that utilizes structured knowledge graphs to generate texts mainly lies in conversation generation (Zhou et al., 2018; Tuan et al., 2019; Moon et al., 2019), story generation (Guan et al., 2019) and language modeling (Ahn et al., 2016; Logan et al., 2019; Hayashi et al., 2019). Zhou et al. (2018) and Guan et al. (2019) propose to use graph attention that incorporates the information of neighbouring concepts into context representations to help generate the target sentence. Yang et al. (2019) resort to a dynamic concept memory that updates during essay generation. Guan et al. (2020) conduct post-training on knowledge triples to enhance the GPT-2 with commonsense knowledge. Since one-hop graphs of concepts in the statement have low coverage to the concepts in the explanation, merely leveraging information of individual concepts or triples is not suitable for this task. Another direction that utilizes more complex graph is to model multi-hop reasoning by performing random walk (Moon et al., 2019) on the knowledge graph or simulating a Markov process on the pre-extracted knowledge paths (Tuan et al., 2019). While in our task, we don't have access to a parallel grounded knowledge source nor the bridge concepts, which makes the problem even more challenging.

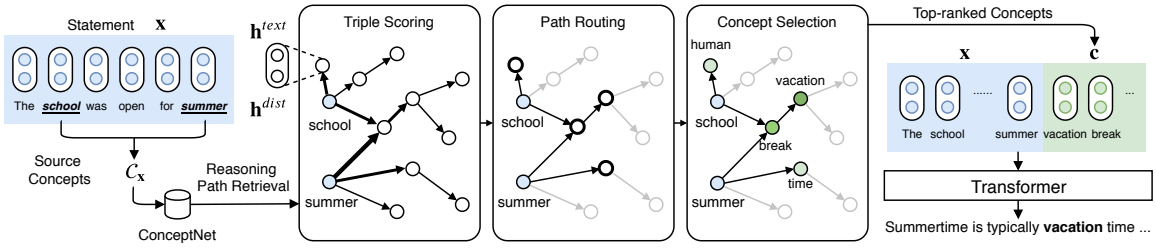

Figure 2: The inference process of our model. In the reasoning path retrieval stage (§3.3), a subgraph is firstly retrieved from the ConceptNet given the source concepts ($\mathcal{C}_\mathbf{x}$), where each node representation is fused with both textual and graph-aware representations (§3.4). Then the model scores each triple on the subgraph, routes the path by propagating the probabilities along paths to the connected nodes, and selects concepts from activated nodes (§3.5). Finally, the model generates the explanation by integrating the token embeddings of both the statement and the top-ranked concepts (§3.6).

3 Methodology

3.1 Task Definition

The commonsense explanation generation task is defined as generating an explanation given a statement against commonsense. Let $\mathbf{x} = x_1 \cdots x_N$ be the input statement with N words and $\mathbf{y} = y_1 \cdots y_M$ be the explanation with M words. A simple sequence-to-sequence formulation which learns a mapping from $\mathbf{x}$ to $\mathbf{y}$ can be adopted in this task:

$$P(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{M} P(\mathbf{y}_t|\mathbf{y}_{<t}, \mathbf{x}). \quad (1)$$

3.2 Model Overview

Formally, our model generates the explanation by firstly extracting the critical bridge concepts $\mathbf{c}$ on a retrieved knowledge graph $G_\mathbf{x}$ given the statement $\mathbf{x}$ and then integrating the bridge concepts and the statement to generate a proper explanation $\mathbf{y}$, which can be formulated as follows:

$$P(\mathbf{y}, \mathbf{c}|\mathbf{x}) = P(\mathbf{c}|\mathbf{x})P(\mathbf{y}|\mathbf{x}, \mathbf{c}) \quad (2)$$

where the bridge concepts $\mathbf{c}$ are defined as the unique concepts delivered in the explanation but not mentioned in the statement. Figure 2 presents the overview of our model framework. Firstly, we retrieve multi-hop reasoning paths from the ConceptNet based on the statement, and heuristically prune the noisy connections to obtain a subgraph for further concept extraction (§3.3). To score the paths and concepts, we obtain the fused concept representation for each node on the subgraph by considering both the contextual and graph information (§3.4). Secondly, we design a path routing algorithm to propagate the triple probabilities along multi-hop paths to the connected concepts and further extract plausible concepts (§3.5). Finally, our model generates the explanation by integrating the statement representation and the selected concept representation as inputs (§3.6).

3.3 Reasoning Path Retrieval

In this section, we demonstrate how we retrieve and prune the reasoning paths to form a subgraph. We also acquire distant supervision for uncovering the bridge concepts in the subgraph to supervise the concept extraction in the next stage.

Given an external commonsense knowledge graph $G = (V, E)$, for each statement $\mathbf{x}$, we extract source concepts $\mathcal{C}_\mathbf{x} = \{c^i_\mathbf{x}\}$ from $\mathbf{x}$ by aligning the surface texts in $\mathbf{x}$ to the concepts in V. We also use the stem form of the surface texts to enable soft alignment and filter out stop words. At the training phase, we extract the target concepts $\mathcal{C}_\mathbf{y} = \{c^j_\mathbf{y}\}$ from the explanation $\mathbf{y}$ with a similar procedure.

Starting with the source concepts, we then retrieve reasoning paths from the knowledge graph to form a subgraph that has relatively **high coverage** to the bridge concepts with a **tractable scale**.

We first examine the minimum length of paths that connect source concepts $\mathcal{C}_\mathbf{x}$ with each concept in the explanation set $\mathcal{C}_\mathbf{y} - \mathcal{C}_\mathbf{x}$. As shown in Figure 3, over 80% of the examples require two or three hops of connection from the source concepts to the concepts that are merely mentioned in the explanation, which indicates the necessity for multi-hop reasoning.

We then count the number of concepts covered by subgraphs with different numbers of hops starting from the source concepts (We only consider concepts in the training data). As Figure 3 shows, the average number of nodes covered by 3-hop sub-

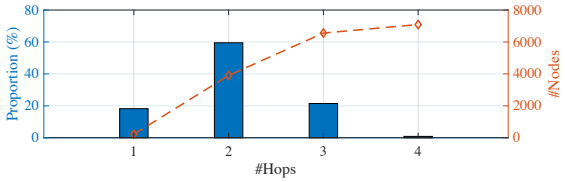

Figure 3: The left axis presents the distribution of the minimum required number of hops to reach the concepts in the explanation set $\mathcal{C}_\mathbf{y} - \mathcal{C}_\mathbf{x}$ from the source concepts in $\mathcal{C}_\mathbf{x}$. The right axis shows the number of nodes in the subgraph with different number of hops.

graph exceeds 6,000, indicating the need of path pruning to keep the scale tractable.

Therefore, we design a heuristic algorithm to retrieve a subgraph $G_\mathbf{x} = \{V_\mathbf{x}, E_\mathbf{x}\}$ from the ConceptNet by expanding the source concepts with 3 hops to cover most bridge concepts. To keep the scale of the subgraph tractable, at each iterating step, we enlarge $V_\mathbf{x}$ with B neighbour concepts most commonly visited by concepts in $V_\mathbf{x}$. Intuitively, the salient bridge concepts should be in a reasonable distance from the source concepts on the graph to maintain the semantic relation and should be commonly visited nodes that support the information flow on the graph.

We distantly label the bridge concepts as the unique concepts in the explanation that could be covered by the subgraph:

$$\mathcal{B}_{\mathbf{x}\to\mathbf{y}} = \{c | c \in \mathcal{C}_\mathbf{y} - \mathcal{C}_\mathbf{x}, c \in V_\mathbf{x}\} \quad (3)$$

3.4 Fused Concept Representation

We initialize each node on the subgraph with a fused concept representation $\mathbf{h}_c$ by considering both the contextual feature of the concept and the graph-aware information. We first obtain the contextualized statement representation $\mathbf{H}_\mathbf{x} \in \mathbb{R}^{N\times d_1}$ using a multi-layer bi-directional Transformer encoder (Vaswani et al., 2017).

$$\mathbf{H}_\mathbf{x}^0 = \texttt{one\_hot}(\mathbf{x}) \cdot \mathbf{W}_e + \mathbf{W}_p \quad (4)$$
$$\mathbf{H}_\mathbf{x}^l = \texttt{trm\_block}(\mathbf{H}_\mathbf{x}^{l-1}),\ l = 1, \ldots, L \quad (5)$$

where $\mathbf{W}_e$ is the token embedding matrix, $\mathbf{W}_p$ is the position embedding matrix, $\texttt{trm\_block}(\cdot)$ is the transformer block with bi-directional attention and L is the number of Transformer blocks. We typically choose the output of the last layer $\mathbf{H}_\mathbf{x}^L$ as the statement representation $\mathbf{H}_\mathbf{x}$.

Then we consider the following embeddings:

- **Context-aware token embedding**. In order to enhance the contextual dependency of the concept c to the statement $\mathbf{x}$, we utilize a bi-attention network (Seo et al., 2016) that models the cross interaction between the concept and the statement.

$$\mathbf{H}_c^{tok} = \texttt{one\_hot}(c) \cdot \mathbf{W}_e \quad (6)$$
$$\mathbf{H}_c^{con} = \texttt{bi-attention}(\mathbf{H}_c^{tok}, \mathbf{H}_\mathbf{x}) \quad (7)$$

  Then we integrate $\mathbf{H}_c^{tok}$ and $\mathbf{H}_c^{con}$ by max pooling and linear transformation to obtain a fixed-length representation that encodes the textual information of the concept c:

$$\mathbf{h}_c^{text} = \texttt{mlp}\Big(\texttt{max}\big([\mathbf{H}_c^{tok}; \mathbf{H}_c^{con}]\big)\Big) \quad (8)$$

- **Concept distance embedding**. To encode the graph-aware structure information into the node representation, we design a concept distance embedding $\mathbf{h}_c^{dist} \in \mathbb{R}^{d_1}$ that encodes the relative distance from concept c to the source concepts $\mathcal{C}_\mathbf{x}$ on the subgraph. Specifically, the concept distance for concept c is defined as the minimum length of the path that can be reached from one source concept in $\mathcal{C}_\mathbf{x}$:

$$d_c = \min_{c_\mathbf{x} \in \mathcal{C}_\mathbf{x}} \text{Dist}(c_\mathbf{x}, c) \quad (9)$$

  The concept distance is then used as an index to look up a trainable matrix $\mathbf{W}_d$ and obtain the $\mathbf{h}_c^{dist} \in \mathbb{R}^{d_1}$.

Finally, the fused concept representation $\mathbf{h}_c$ is obtained by concatenating the context-aware token embedding and the concept distance embedding.

$$\mathbf{h}_c = [\mathbf{h}_c^{text}; \mathbf{h}_c^{dist}] \quad (10)$$

3.5 Bridge Concept Extraction

We describe the core component of our method in this section, which extracts the bridge concepts for further explanation generation. It first scores triples on the subgraph to downweight the noisy paths. Then it aggregates the path scores to each connected concepts by a path routing process and deactivates the nodes with low routing scores. Finally it selects top-ranked bridge concepts from the activated nodes.

3.5.1 Triple Scoring

Firstly, we calculate the triple scores according to the representation of triples and the input statement. For each triple $e = (c_{e,head}, r_e, c_{e,tail})$ where $c_{e,head}$/$c_{e,tail}$ indicates the head / tail concept and r_e denotes the relation, we can obtain its representation by concatenating the representations of the head concept, the relation and the tail concept:

$$\mathbf{h}_e = [\mathbf{h}_{c_{e,head}}; \mathbf{h}_{r_e}; \mathbf{h}_{c_{e,tail}}] \tag{11}$$

Both the head and the tail representations are calculated by Equation (10) and the relation representation is acquired by indexing a trainable relation embedding matrix $\mathbf{W}_r$. Then we use the statement representation to query each triple representation by taking the bilinear dot-product attention and calculate the selection probability for each triple:

$$\mathbf{h}_\mathbf{x} = \texttt{max-pooling}(\mathbf{H}_\mathbf{x}) \in \mathbb{R}^{d_1} \tag{12}$$

$$P(\mathbf{e}|\mathbf{x}) = \sigma(\mathbf{h}_\mathbf{e}\mathbf{W}_2\mathbf{h}_\mathbf{x}^T) \tag{13}$$

We adopt weak supervision to supervise the triple scoring process. For each concept $c \in \mathcal{B}_{\mathbf{x}\to\mathbf{y}}$, we obtain the set of the shortest paths $\mathbf{P}_{\mathbf{x}\to c}$ using the breadth-first search from each concept of $\mathcal{C}_\mathbf{x}$ to c. We consider all these shortest paths $\mathbf{P}_{\mathbf{x}\to\mathbf{y}} = \bigcup_{c\in\mathcal{B}_{\mathbf{x}\to\mathbf{y}}} \mathbf{P}_{\mathbf{x}\to c}$ as the supervision of our triple scoring process as they connect the reasoning chain from the statement to the explanation with minimum distractive information. Accordingly, other triples in $G_\mathbf{x}$ which don't belong to $\mathbf{P}_{\mathbf{x}\to\mathbf{y}}$ are regarded as negative samples. The loss function of triple scoring is devised as follows:

$$\begin{aligned}\mathcal{L}_{triple} = &- \sum_{\mathbf{e}\in G_\mathbf{x}} \mathbb{I}(\mathbf{e} \in \mathbf{P}_{\mathbf{x}\to\mathbf{y}}) \log P(\mathbf{e}|\mathbf{x}) \\ &+ [1 - \mathbb{I}(\mathbf{e} \in \mathbf{P}_{\mathbf{x}\to\mathbf{y}})] \log[1 - P(\mathbf{e}|\mathbf{x})]\end{aligned} \tag{14}$$

where $\mathbb{I}(\mathbf{e} \in \mathbf{P}_{\mathbf{x}\to\mathbf{y}})$ is an indicator function that takes the value 1 iff $\mathbf{e} \in \mathbf{P}_{\mathbf{x}\to\mathbf{y}}$, and 0 otherwise.

3.5.2 Path Routing

Next, we describe the path routing process which involves propagating the scores along the paths to each concept on the subgraph from the source concepts. For each path $\mathbf{p}$ retrieved from the subgraph $G_\mathbf{x}$, we calculate a path score $s(\mathbf{p})$ by aggregating the triple score $P(\mathbf{e}|\mathbf{x})$ along the path:

$$s(\mathbf{p}) = \frac{1}{|\mathbf{p}|} \sum_{\mathbf{e}\in\mathbf{p}} P(\mathbf{e}|\mathbf{x}) \tag{15}$$

For each concept c, we consider all the shortest paths $\mathbf{P}_{\mathbf{x}\to c}$ that starts with the source concepts and ends with c monotonically, i.e., the concept distance of each node on the path increases monotonically along the path. Then we calculate the routing score for the concept c by averaging the path scores of $\mathbf{P}_{\mathbf{x}\to c}$.

$$s(c) = \frac{1}{|\mathbf{P}_{\mathbf{x}\to c}|} \sum_{\mathbf{p}\in\mathbf{P}_{\mathbf{x}\to c}} s(\mathbf{p}) \tag{16}$$

Intuitively, this process disseminates the triple scores and aggregates them to the connected concepts. Then we deactivate some paths based on the path routing results and obtain $V_{\mathbf{x}\to\mathbf{y}}$ by preserving concepts with the top-K_1 routing scores.

3.5.3 Concept Selection

Finally, we conduct concept selection based on the concept representation and the statement representation. For each concept in $V_{\mathbf{x}\to\mathbf{y}}$, we calculate the selection probability for it by taking the dot-product attention and adopt a similar cross-entropy loss with supervision from bridge concepts $\mathcal{B}_{\mathbf{x}\to\mathbf{y}}$:

$$P(c|\mathbf{x}) = \sigma(\mathbf{h}_c\mathbf{W}_3\mathbf{h}_\mathbf{x}^T) \tag{17}$$

$$\begin{aligned}\mathcal{L}_{concept} = &- \sum_{\mathbf{c}\in V_{\mathbf{x}\to\mathbf{y}}} \mathbb{I}(\mathbf{c} \in \mathcal{B}_{\mathbf{x}\to\mathbf{y}}) \log P(c|\mathbf{x}) \\ &+ [1 - \mathbb{I}(\mathbf{c} \in \mathcal{B}_{\mathbf{x}\to\mathbf{y}})] \log[1 - P(c|\mathbf{x})]\end{aligned} \tag{18}$$

where the indicator function is similar to that of Equation (14).

Finally, the bridge concepts with top-K_2 probability $P(c|\mathbf{x})$ are selected as the additional input to the generation model.

3.6 Explanation Generation

We utilize a pre-trained Transformer decoder (Radford et al., 2019) as our generation model which shares the parameter with the Transformer encoder. Essentially, it takes the statement $\mathbf{x}$ and the concepts $\mathbf{c}$ as input and auto-regressively generates the explanation $\mathbf{y}$:

$$\begin{aligned}P(\mathbf{y}|\mathbf{x},\mathbf{c}) &= P(\mathbf{y}|\mathbf{x}, c_1, \cdots, c_{K_2}) \\ &= \prod_{t=1}^{M} P(\mathbf{y}_t|\mathbf{x}, c_1, \cdots, c_{K_2}, \mathbf{y}_{<t})\end{aligned} \tag{19}$$

$$\mathcal{L}_{generation} = -\log P(\mathbf{y}|\mathbf{x}, c_1, \cdots, c_{K_2}) \tag{20}$$

As shown in Figure 2, the input to the Transformer decoder is the token embeddings of both the statement and the selected concepts concatenated along the sequence length dimension.

To model bi-directional attention on the input side while preserving the causal dependency of the generated sequence, we adopt a hybrid attention mask where each token on the input side could attend to all the tokens in the input sequence while the generated token at each time step only attends to the input sequence and the previously generated tokens.

3.7 Training and Inference

To train the model, we optimize the final loss function which is the sum of the three loss functions:

$$\mathcal{L}_{final} = \mathcal{L}_{generation} + \lambda_1 \mathcal{L}_{triple} + \lambda_2 \mathcal{L}_{concept} \tag{21}$$

As for the inference process, Figure 2 demonstrates how our model retrieves reasoning paths given the statement, extracts bridge concepts and finally generates the explanation.

4 Experiment

4.1 Dataset and Experimental Setup

4.1.1 Commonsense Explanation Dataset

We adopt the dataset from the Commonsense Validation and Explanation Challenge[2] which consists of three subtasks, i.e., commonsense validation, commonsense explanation selection and commonsense explanation generation. We focus on the explanation generation subtask in this paper. The commonsense explanation generation subtask contains $10,000$ statements that are against commonsense. For each statement, three human-written explanations are provided. To evaluate our proposed model and other baselines, we randomly split 10% data as the test set, 5% as the development set and the latter as the training set. Note that we further split each example in the training set into three statement-explanation pairs, while for the development set and the test set we use the three corresponding explanations as references for each statement. This results in our final data split (25,596 / 476 / 992) denoted as (train / dev / test).

[2] https://competitions.codalab.org/competitions/21080

4.1.2 Commonsense Knowledge Graph

We use the English version ConceptNet as our external commonsense knowledge graph. It contains triples in the form of (h, r, t) where h and t represent head and tail concepts and r is the relation type. We follow Lin et al. (2019a) to merge the original 42 relation types into 17 types. We additionally define 17 reverse types corresponding to the original 17 relation types to distinguish the direction of the triples on the graph.

4.2 Automatic Evaluation Metrics

To automatically evaluate the performance of the generation models, we use the **BLEU-3/4** (Papineni et al., 2001), **ROUGE-2/L** (Lin, 2004), **METEOR** (Banerjee and Lavie, 2005) as our main metrics. We also propose **Concept F1** to evaluate the accuracy of the unique concepts in the generated explanation that do not occur in the statement.

Specifically, given the generated explanation $\hat{\mathbf{y}}$ and the reference explanation $\mathbf{y}$, we extract a set of concepts $\mathcal{C}_{\hat{\mathbf{y}}}$ and $\mathcal{C}_{\mathbf{y}}$ from the generated explanation and the reference explanation respectively using the method in §3.3. We denote the sets of unique concepts in the explanation as $\mathcal{U}_{\mathbf{y}} = \mathcal{C}_{\mathbf{y}} - \mathcal{C}_{\mathbf{x}}$ and $\mathcal{U}_{\hat{\mathbf{y}}} = \mathcal{C}_{\hat{\mathbf{y}}} - \mathcal{C}_{\mathbf{x}}$. Then we can compute the Concept F1 as the harmonic mean of *recall* and *precision*.

$$recall = \frac{|\mathcal{U}_{\hat{\mathbf{y}}} \cap \mathcal{U}_{\mathbf{y}}|}{|\mathcal{U}_{\mathbf{y}}|}, \; precision = \frac{|\mathcal{U}_{\hat{\mathbf{y}}} \cap \mathcal{U}_{\mathbf{y}}|}{|\mathcal{U}_{\hat{\mathbf{y}}}|} \tag{22}$$

4.3 Implementation Details

For the reasoning path retrieval process, we set the maximum number of neighbours $B = 300$ at each hop. For each example, we restrict the concepts of the subgraph to those only appeared in the training and development set.

We use a pre-trained Transformer language model GPT-2 (Radford et al., 2019) as the initialization of the Transformer model. We set the hidden dimension $d_1 = 768$ identical to the hidden size of the Transformer. We empirically set the following hyperparameters by tuning the model on the development set: selection threshold $K_1 = 30, K_2 = 3$, loss coefficients $\lambda_1 = 1, \lambda_2 = 1$, number of epochs $= 3$, batch size $= 4$, learning rate $= 4 \times 10^{-5}$ and use the Adam optimizer (Kingma and Ba, 2015) with 10% warmup steps. We select the model with the highest BLEU-4 score on the development set and evaluate it on the test set. At the decoding

Model	B-3/4	R-2/L	M	Concept F1
Seq2Seq	10.7/6.1	9.9/25.8	11.4	11.1
MemNet	10.2/5.7	8.8/25.7	11.0	11.5
Transformer	10.0/5.8	9.6/26.0	12.0	11.7
GPT-2-FT	23.4/15.7	18.9/36.5	17.7	17.4
Ours	**24.7/17.1**	**20.2/37.9**	**18.3**	**20.1**

Table 1: Automatic evaluation of explanation generation in terms of BLEU (B), ROUGE (R), METEOR (M) and Concept F1.

Setting	BLEU-4	Concept F1
Ours	17.1	20.1
w/o Context Emb.	16.0	18.6
w/o Distance Emb.	16.4	18.5
w/o Path Routing	16.5	19.2
#Hop = 2	16.2	18.3
#Hop = 1	15.9	17.3

Table 2: Ablation study of our framework on the test set. We present the model ablation results in the upper block and the data ablation results in the lower block.

phase, we use beam search with a beam size of 3 for all models.

4.4 Baseline Models

We compare with the following baseline models:

- **Seq2Seq**: a sequence-to-sequence model based on gated recurrent unit (GRU) (Cho et al., 2014) and attention mechanism, which is widely used in text generation tasks (Bahdanau et al., 2015).
- **MemNet**: a knowledge-grounded sequence-to-sequence model (Ghazvininejad et al., 2018). In our experimental setting, we regard all the concepts which are connected with those in the statements as knowledge facts.
- **Transformer**: an encoder-decoder framework commonly used in machine translation tasks (Vaswani et al., 2017).
- **GPT-2**: a multi-layer Transformer decoder pre-trained on WebText (Radford et al., 2019) which is then directly fine-tuned on our dataset.

4.5 Experimental Results

As shown in Table 1, our model achieves the best performance in terms of all the automatic evaluation metrics, which demonstrates that our model

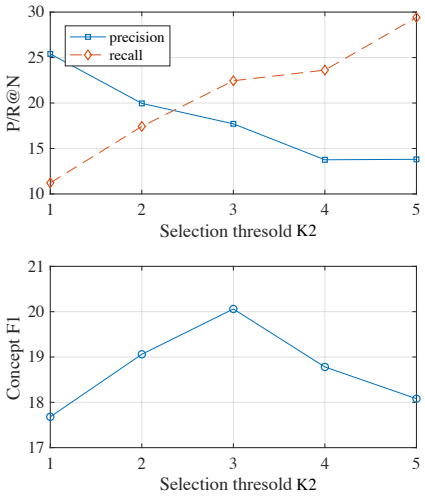

Figure 4: P/R@N measures the precision / recall of the top-N selected bridge concepts. Concept F1 measures the F1-score of concepts in the generated explanations.

can generate high quality explanations. Specifically, our model achieves a 2.7% gain on Concept F1 compared with GPT-2 which indicates that explicitly extracting bridge concepts enhances the informativeness of the generated explanation.

To evaluate the effects of different modules in our method, we conduct ablation studies on both the model components and the external knowledge base. For the model components, we test the following variants: (1) without the context-aware token embeddings (**w/o Context Emb.**); (2) without the concept distance embeddings (**w/o Distance Emb.**); (3) without the path routing process (**w/o Path Routing**). As for the data ablation, we sample subgraphs by restricting the maximum number of hops to 2 (**#Hop=2**) and 1 (**#Hop=1**).

As shown in Table 2, each module contributes to the final results. Particularly, discarding the context-aware embeddings leads to the most remarkable performance drop, which indicates the significance for context modeling in multi-hop reasoning. Besides, the data ablation results demonstrate that as the subgraph has less coverage, the generation model will suffer from the noisy concepts and thus deteriorate the generation results.

We additionally present the results of the selected and generated concepts with different concepts selection threshold K_2. As shown in the upper part of the Figure 4, as the number of selected concepts increases, more true positives are selected, resulting in the increase of the recall (Recall@N) while the inclusion of more false positives leads to

Error Type	Ratio (%)	Input	Output
Repetition	7.7	She begins working for relaxation.	People work to relax, not relax.
Overstatement	19.2	Less people seek knowledge.	People don't seek knowledge.
Unrelated	26.9	The simplest carbohydrates are amino acid.	Alkaloids are not found in bread.
Chaotic	11.5	Giving assistance is for revenge.	If you help someone, you are grateful.

Table 3: Distribution and typical cases of different error types of the explanations generated by our model. Underlined texts denote the error types including repetition, overstatement, unrelated words and chaotic expression.

Model	Fluency		Reasonability		Informativeness	
	Win	Lose	Win	Lose	Win	Lose
vs. Seq2seq	0.41	0.02	0.86	0.04	0.84	0.05
vs. MemNet	0.48	0.00	0.84	0.03	0.87	0.03
vs. Transformer	0.33	0.01	0.71	0.03	0.72	0.03
vs. GPT-2	0.20	0.10	0.40	0.27	0.34	0.15

Table 4: Human evaluation results. The scores are the percentages of win and $lose$ of our model in pair-wise comparison (tie can be calculated by $1 - win - lose$). Our model is significantly better (sign test, p-value $<$ 0.005) than all the baseline models on all three criteria.

the decrease of the precision (Precision@N) . The Concept F1 reaches maximum when $K_2 = 3$ (see the lower part), which demonstrates that the model learns to extract critical concepts for explanation generation while keeping out most noisy candidates with an appropriate selection threshold.

4.6 Human Evaluation

To further evaluate the quality of the generated explanations, we conduct the human evaluation and recruit five annotators to perform pair-wise comparisons. Each annotator is given 100 paired explanations (one generated by our model and the other by a baseline model, along with the statement) and is required to give a preference among "win", "tie", and "lose" according to three criteria: (1) *Fluency* which measures the grammatical correctness and the readability of the explanation. (2) *Reasonability* which measures whether the explanation is reasonable and accords with the commonsense. (3) *Informativeness* which measures the amount of new information delivered in the explanation that helps explain the statement.

The results are shown in Table 4, our model outperforms all the baseline models significantly on all three criteria (sign test, p-value $<$ 0.005). Specifically, our model wins GPT-2 substantially in terms of reasonability and informativeness.

To evaluate the inter-rater agreement for each criterion, we calculate the Fleiss' kappa (Fleiss, 1971). For *Reasonability* / *Informativeness*, the kappa is 0.429 / 0.433 respectively indicating a moderate agreement among annotators. In terms of *Fluency*, annotators show diverse preferences ($\kappa =$ 0.245) since GPT-2 has strong ability in generating fluent texts.

Statement 1: I buy popcorn and knife when I go to the cinema.
Seq2Seq: A person cannot buy a person to watch a movie. **MemNet:** A toothbrush is not a place to play a movie. **Transformer:** A fridge is not a place to store groceries. **GPT-2:** You don't buy popcorn and knife at the cinema.
Ours: Knives are not **sold** at the cinema. **Top-3 reasoning paths:** (buy→*antonym*→sell), (popcorn → *related to*→food), (cinema→ *related to*→ movie) **Selected concepts:** **sell**, place, movie
Statement 2: He eats his chips with toothpaste.
Seq2Seq: Chopsticks are not edible. **MemNet:** A potato is too soft to eat juice with your teeth. **Transformer:** You do not eat sand with a cup. **GPT-2:** Toothpaste is not edible.
Ours: Toothpaste is used to clean teeth. **Top-3 reasoning paths:** (eat→*related to*→tooth), (toothpaste→*related to*→paste→*related to*→use), (eat →*has subevent* →work→*related to*→use) **Selected concepts:** **use**, **tooth**, food

Table 5: Examples of generated explanations. Irrelevant contents are in red and critical concepts for explanation are in green.

4.7 Case Study

Table 5 presents the generated explanations. Our model is capable to generate reasonable and informative explanations by utilizing the extracted bridge concepts. Specifically, in the first case our model extracts bridge concepts "sell" and identifies the incompatibility between "knives" and "cinema". In the second case, our model clarifies the function of the "toothpaste" by extracting "use" from two reasoning paths and provides more information rather than simply negative phrasing.

4.8 Error Analysis

To analyze the error types of the explanations generated by our model, we manually check all the failed cases[3] in the pair-wise comparison between our model and the strong baseline GPT-2. The number of these cases is 26 in all 100 explanations. We manually annotated four types of errors from the failed explanations: **repetition** (words repeating), **overstatement** (overstate the points), **unrelated** concepts towards the statement (the explanation itself may be reasonable), **chaotic** sentences (difficult to understand). As shown in Table 3, it is still challenging for the model to generate explanations highly related to the statement with accurate wording.

5 Conclusion

In this paper, we analyze the challenges in incorporating external knowledge graph to aid the commonsense generation problem and propose a two-stage method that first extracts bridge concepts from a retrieved subgraph and then generates the explanation by integrating the extracted concepts. Experimental results show that our model outperforms baselines including the strong pre-trained language model GPT-2 in both automatic and manual evaluation.

Acknowledgments

This work was jointly supported by the NSFC projects (key project with No. 61936010 and regular project with No. 61876096), and the Guoqiang Institute of Tsinghua University with Grant No. 2019GQG1. We thank THUNUS NExT Joint-Lab for the support.

References

Sungjin Ahn, Heeyoul Choi, Tanel Pärnamaa, and Yoshua Bengio. 2016. A neural knowledge language model. *ArXiv*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *EMNLP*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *ACL*.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, pages 1724–1734.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *AAAI*.

Jian Guan, Fei Huang, Minlie Huang, Zhihao Zhao, and Xiaoyan Zhu. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Trans. Assoc. Comput. Linguistics*, 8:93–108.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *AAAI*.

Hiroaki Hayashi, Zecong Hu, Chenyan Xiong, and Graham Neubig. 2019. Latent relation language models. *ArXiv*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2019. What's missing: A knowledge gap guided approach for multi-hop question answering. In *EMNLP*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2011. The winograd schema challenge. In *KR*.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019a. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *EMNLP/IJCNLP*.

Bill Yuchen Lin, Ming Shen, Yu Xing, Pei Zhou, and Xiang Ren. 2019b. Commongen: A constrained text generation dataset towards generative commonsense reasoning. *ArXiv*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *ACL*.

Robert L. Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August*

[3]The decision is based on majority voting by the five annotators.

2, 2019, Volume 1: Long Papers, pages 5962–5971. Association for Computational Linguistics.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *ACL*, pages 845–854.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

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

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *ACL*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social iqa: Commonsense reasoning about social interactions. In *EMNLP*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *ArXiv*.

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

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL-HLT*.

Yi-Lin Tuan, Yun-Nung Chen, and Hung-yi Lee. 2019. DyKgChat: Benchmarking dialogue generation grounding on dynamic knowledge graphs. In *EMNLP-IJCNLP*, pages 1855–1865.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. 2019. Does it make sense? and why? a pilot study for sense making and explanation. In *ACL*.

Pengcheng Yang, Lei Li, Fuli Luo, Tianyu Liu, and Xu Sun. 2019. Enhancing topic-to-essay generation with external commonsense knowledge. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2002–2012, Florence, Italy. Association for Computational Linguistics.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629.