Paraphrase Generation by Learning How to Edit from Samples

Amirhossein Kazemnejad[†], **Mohammadreza Salehi**[‡], **Mahdieh Soleymani Baghshah**[‡]
[†]Iran University of Science and Technology, [‡]Sharif University of Technology
a_kazemnejad@comp.iust.ac.ir, mrezasalehi@ce.sharif.edu,
soleymani@sharif.edu

Motivation

- paraphrases generated by seq2seq model suffer from lack of quality and diversity.
- the proposed Seq2Seq methods for paraphrase generation have shown promising results, but available training data for paraphrasing is scarce and has been evaluated, no previous study, the usage of this domain-specific.
- Although retrieval-based text generation approach in paraphrase generation

Task Definition and our method

- Let $D = \{x_{\mathrm{k}}, y_k\}_{k=1}^{n}$ denotes a dataset where x_n is a sequence of words of words, and y_n is the target paraphrase.
- Our goal is to find the model that maximizes $\prod_{k=1}^{n} p_{model}(y_k|x_k)$
- Given an input sequence x, the retriever first finds a paraphrase pair (p, q) from the training corpus based on the similarity of x and p. Then, the editor utilizes the retrieved pair (p,q) to paraphrase x.

Retriever

- The goal of the retriever module is to select the paraphrase pairs (from the training corpus) that are similar to input sequence x
- We first find a neighbor hood set $\mathcal{N}(x)$ consisting of the K most similar source sentences $\{p_k\}_{k=1}^{K}$ to x and their associated paraphrases $\{q_k\}_{k=1}^{K}$.
- We embed them employing the pre-trained transformer based encoder, The similarity is then calculated using cosine similarity measure in the resulted embedding space.

Retriever

- With the help of FAISS software package, we pre-compute the neighborhood set of each source sentence in the training set, so at the training time, we only need to sample one of the pairs in the neighbor hood set.

$$p((p,q)|x) = \frac{1}{K}\mathbf{1}[(p,q) \in \mathcal{N}(x)].$$

Edit Provider

- This part of the editor extracts the edits from the retrieved pair as a set of vectors which we call **M**icro **E**dit **V**ectors (MEVs).
- . MEVs are responsible for encoding the information about fine-grained edits that transform p into q.

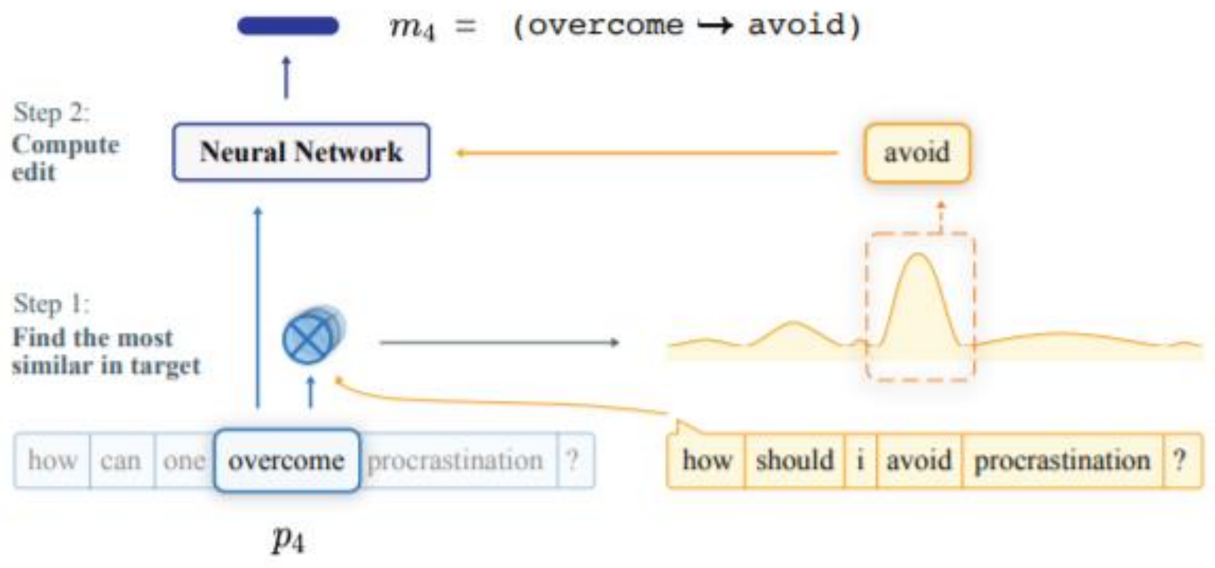


Figure 2: The general scheme of computing a MEV corresponding to a token of p.

Edit Provider

- For each arbitrary token of p, such as p_i, we intend to compute a MEV that encodes the edit corresponding to p_i using the attention over q.

- This module outputs a vector that encodes the most semantically relevant parts of q to p_i After that, the MEVs, i.e., the m_i s, are Computed by feeding these vectors one by One into a single dense layer.

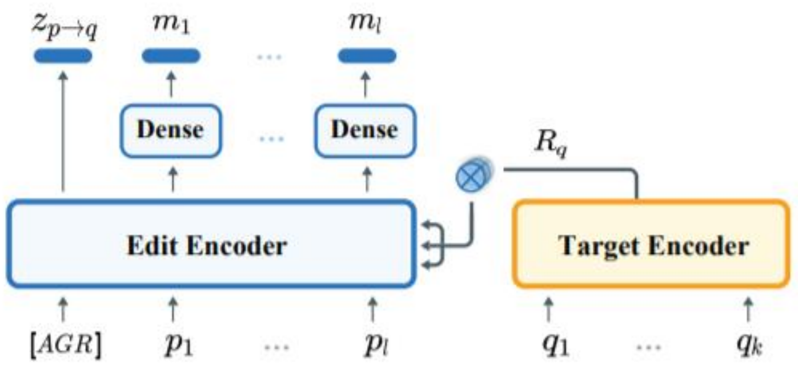


Figure 3: Architecture of Edit Provider. The Edit En coder uses multi-head attention on R_q to select the ta get of edit for each token of p. Note that by prependin $[AGR]$ to p, we can encode all of the MEVS into single edit vector $z_{p \to q}$.

Edit Provider

- Finally, all of the MEVs are aggregated into a single vector z; we prepend a special [ARG] to p in order to encode all edits into a single vector $z_{p\rightarrow q}$.
- We run the edit Provider in the reverse direction (from q to p), and get $z_{q\rightarrow p}$. The final edit vector is then computed as :

$$z = \texttt{Linear}(z_{p\rightarrow q} \oplus z_{q\rightarrow p}),$$

Edit Performer

- The Edit Performer transforms the input sequence $x = [x_1, x_2, \ldots, x_s]$ to the final output $\hat{y}$. We employ a fully-attentional Seq2seq architecture.
- The encoder encodes the input sequence and outputs a context-aware representation $R_x = \{r_x^i\}_{i=1}^{s}$ of the input sequence.
- For the decoder, as the relation between p and q is encoded in MEVs M and the vector z, therefore ,we need M,z to specify the edits. The sentence p to identify the locations in x to which the edits should be applied. Thus we aim to model $p(y|x, p, M, z)$ for the decoder.

Edit Performer

- We append z to each token of the decoder's input
- the model also attends to MEVs M using an extra multi-head attention sub-layer which computes the representation

$$h' = \texttt{MultiHeadAtt}(\mathbf{Q}\text{: } h, \mathbf{K}\text{: } R_p, \mathbf{V}\text{: } M),$$

- h comes from the previous sub-layer and R_p is the context-aware representation of the retrieved sequence p, calculated by the Edit Provider

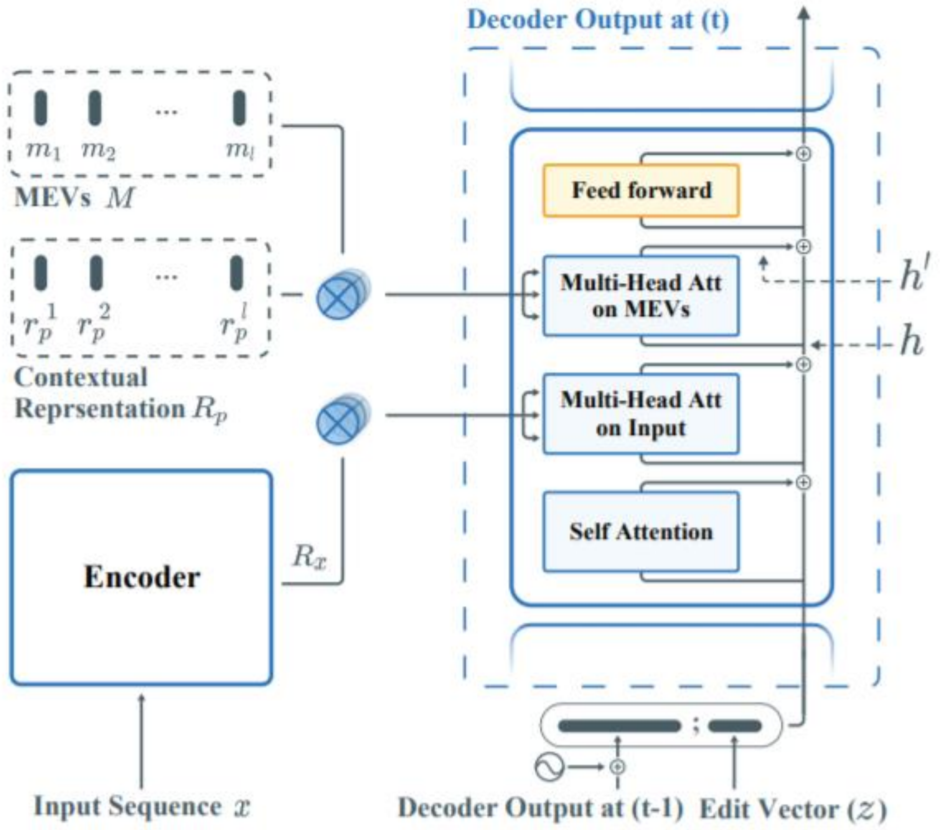


Figure 4: Illustration of the Edit Performer generating the output token at t-th time step. Note that only one layer of the decoder is depicted and the layernorms are not shown for simplicity.

Training

- Our aim is to maximize the following log likelihood objective:

$$\mathcal{L} = \sum_{(x,y)\in\mathcal{D}} \log p(y|x).$$

- As we decompose the training procedure to two stages of retrieving and editing:

$$p(y|x) = \sum_{(p,q)\in\mathcal{D}} p(y|x,(p,q))p((p,q)|x).$$

- Which is equal to:

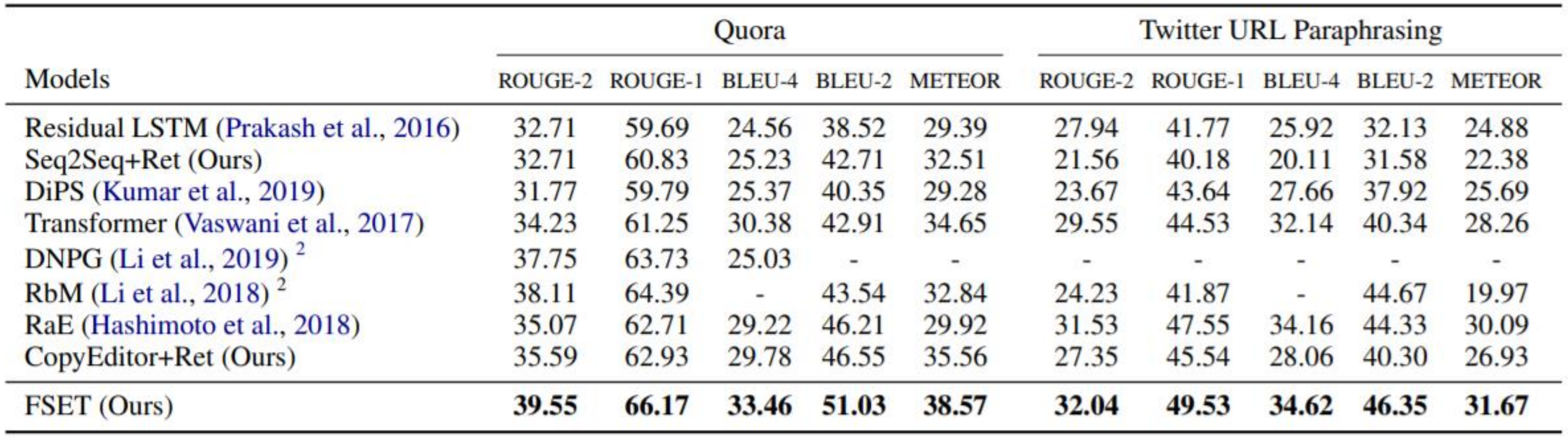
$$\mathcal{L} = \sum_{(x,y)\in\mathcal{D}} \log(\frac{1}{K} \sum_{(p,q)\in\mathcal{N}(x)} p(y|x,(p,q))).$$

- The lower bound:

$$\mathcal{L} \geq \mathcal{L}' = \frac{1}{K} \sum_{(x,y)\in\mathcal{D}} \sum_{(p,q)\in\mathcal{N}(x)} \log p(y|x,(p,q)).$$

Experiments

- We conduct experiments on two of the most frequently used datasets for paraphrase generation: the Quora question pair dataset and the Twitter URL paraphrasing corpus.

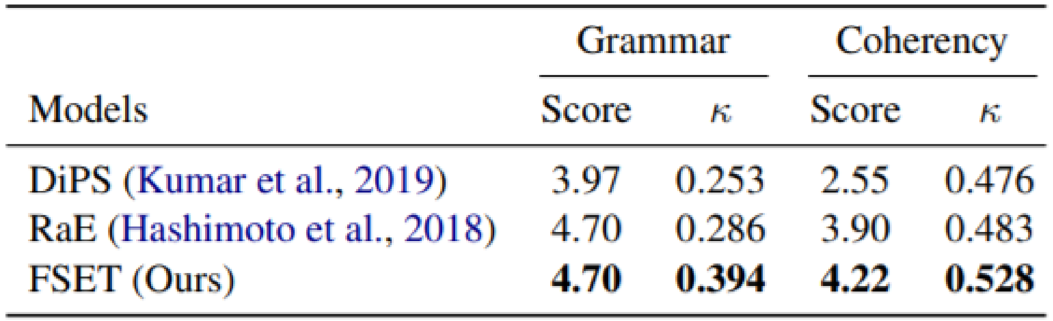


	Quora					Twitter URL Paraphrasing				
Models	ROUGE-2	ROUGE-1	BLEU-4	BLEU-2	METEOR	ROUGE-2	ROUGE-1	BLEU-4	BLEU-2	METEOR
Residual LSTM (Prakash et al., 2016)	32.71	59.69	24.56	38.52	29.39	27.94	41.77	25.92	32.13	24.88
Seq2Seq+Ret (Ours)	32.71	60.83	25.23	42.71	32.51	21.56	40.18	20.11	31.58	22.38
DiPS (Kumar et al., 2019)	31.77	59.79	25.37	40.35	29.28	23.67	43.64	27.66	37.92	25.69
Transformer (Vaswani et al., 2017)	34.23	61.25	30.38	42.91	34.65	29.55	44.53	32.14	40.34	28.26
DNPG (Li et al., 2019) [2]	37.75	63.73	25.03	-	-	-	-	-	-	-
RbM (Li et al., 2018) [2]	38.11	64.39	-	43.54	32.84	24.23	41.87	-	44.67	19.97
RaE (Hashimoto et al., 2018)	35.07	62.71	29.22	46.21	29.92	31.53	47.55	34.16	44.33	30.09
CopyEditor+Ret (Ours)	35.59	62.93	29.78	46.55	35.56	27.35	45.54	28.06	40.30	26.93
FSET (Ours)	**39.55**	**66.17**	**33.46**	**51.03**	**38.57**	**32.04**	**49.53**	**34.62**	**46.35**	**31.67**

Table 2: Results of the different models on two paraphrasing datasets.

Experiments

the inter-annotator agreement measured by Cohen's kappa κ shows fair or intermediate agreement between raters assessing the models.



	Grammar		Coherency	
Models	Score	κ	Score	κ
DiPS (Kumar et al., 2019)	3.97	0.253	2.55	0.476
RaE (Hashimoto et al., 2018)	4.70	0.286	3.90	0.483
FSET (Ours)	**4.70**	**0.394**	**4.22**	**0.528**

Table 3: Human evaluation on Quora dataset.

Language Generation with Multi-Hop Reasoning on Commonsense Knowledge Graph

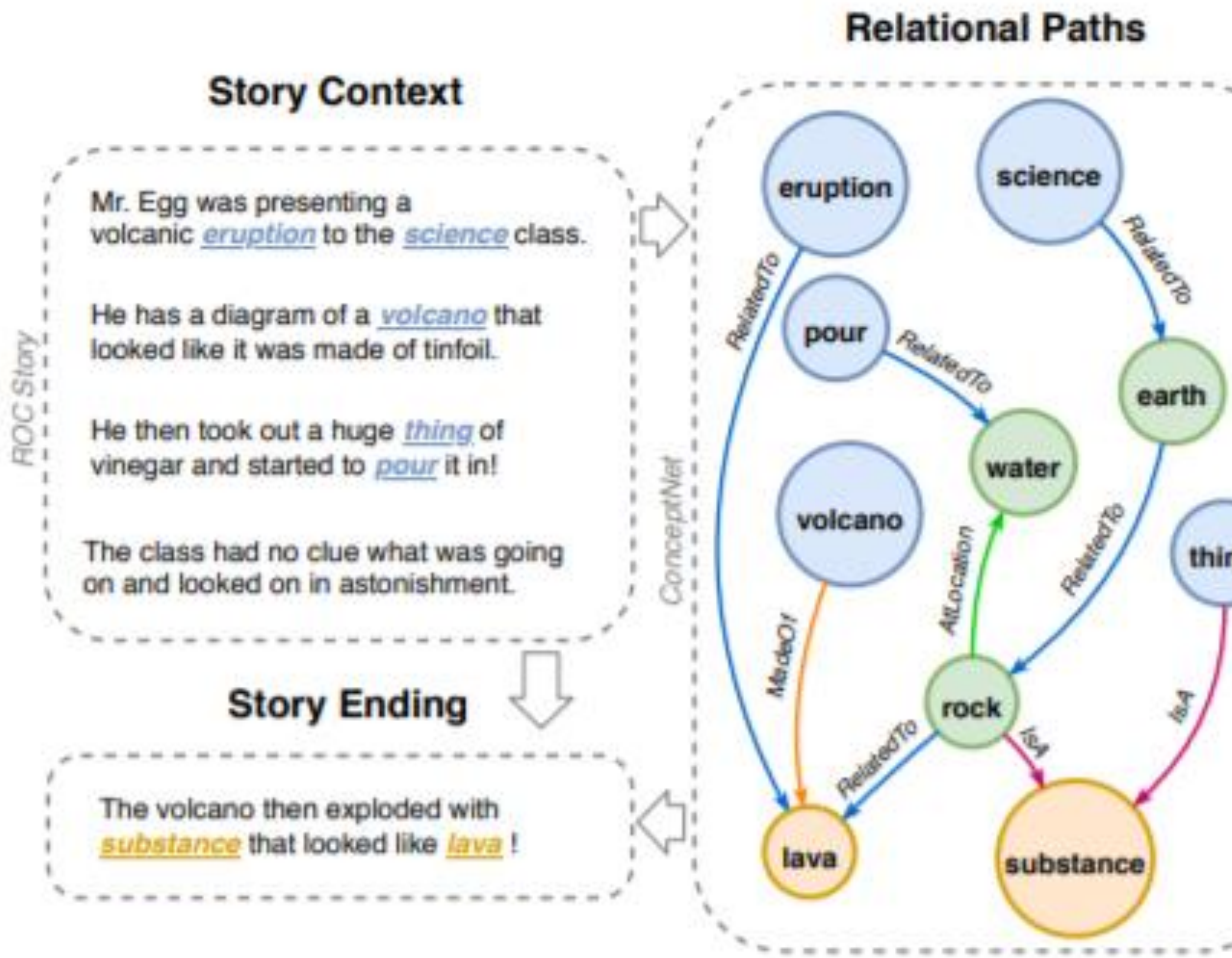
Haozhe Ji[1], Pei Ke[1], Shaohan Huang[2], Furu Wei[2], Xiaoyan Zhu[1], Minlie Huang[1*]
[1]Department of Computer Science and Technology, Institute for Artificial Intelligence,
State Key Lab of Intelligent Technology and Systems,
Beijing National Research Center for Information Science and Technology,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

Motivation

- Despite the success of generative pre-trained language models on a series of text generation tasks, they still suffer in cases where reasoning over underlying commonsense knowledge is required during generation
- Models require reasoning over commonsense knowledge that is not explicitly stated in the context.

Motivation

- For example, in Figure1 , the external commonsense knowledge in the form of relational paths can guide the generation of the key concepts “substance” and “lava” , by providing background knowledge (volcano, MadeOf, lava)
- Existing methods possess commonsense reasoning ability by implicitly learning some relational patterns from large-scale corpora, they don’t fully utilize KB


Relational Paths
Story Context
ROC Story
Mr. Egg was presenting a volcanic eruption to the science class.
He has a diagram of a volcano that looked like it was made of tinfoil.
He then took out a huge thing of vinegar and started to pour it in!
The class had no clue what was going on and looked on in astonishment.
ConceptNet
Story Ending
The volcano then exploded with substance that looked like lava !
eruption
science
pour
earth
water
volcano
rock
lava
substance
RelatedTo
MadeOf
AtLocation
IsA

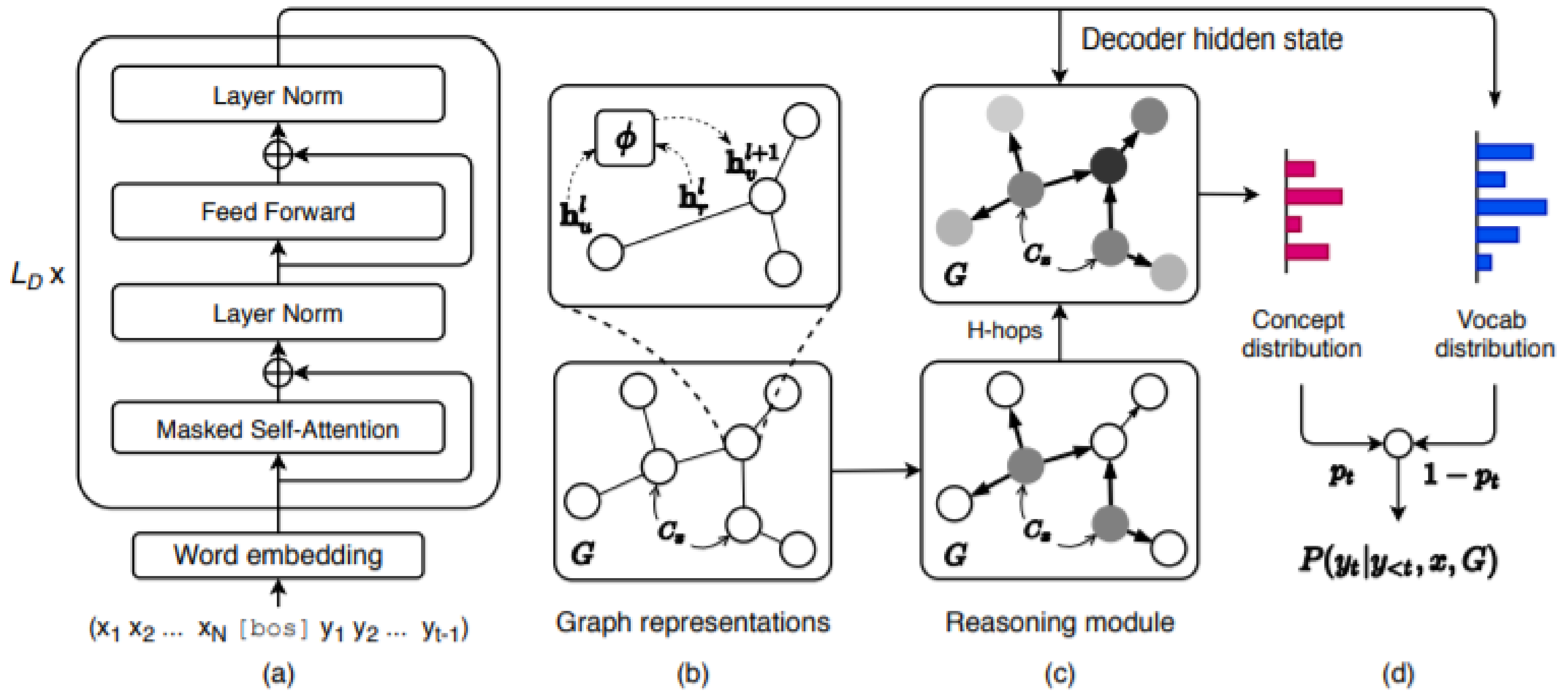
Motivation

- Previous work transfers commonsense knowledge into pre-trained language models by utilizing triple information in commonsense KB.
- However, this method has two drawbacks, (1) recovering knowledge triples at the post training stage hardly enables the model to utilize the encoded knowledge in fine-tuning generation tasks (2) it ignores the abundant structural relational relevance of the concepts in the graph

Method

- In this paper, we propose Generation with MultiHop Reasoning Flow (GRF), a generation model that performs multi-hop reasoning on the external knowledge graph for knowledge-enriched language generation
- It constructs sub-graph extended from the concepts in the input text
- It first encodes the multi-relational graph with compositional operation to obtain graph-aware representations for the concepts and the relations
- Then, the multi-hop reasoning module performs dynamic reasoning via aggregating triple evidence along multiple relational paths
- Finally, the generator combines the probability of copying the concept from knowledge graph and choosing a word from the vocabulary.

Framework



Decoder hidden state
Layer Norm
Feed Forward
Layer Norm
Masked Self-Attention
L_D x
Word embedding
$(x_1\ x_2 \dots\ x_N$ [bos] $y_1\ y_2 \dots\ y_{t-1})$
(a)
ϕ
$\mathbf{h}_v^{l+1}$
$\mathbf{h}_u^l$
$\mathbf{h}_r^l$
G
C_s
Graph representations
(b)
H-hops
Reasoning module
(c)
Concept distribution
Vocab distribution
p_t
$1 - p_t$
$P(y_t|y_{<t}, x, G)$
(d)

Problem formulation

- We focus on text generation tasks where reasoning over exernal commonsense knlowedge is required.
- the input source is a text sequence $\boldsymbol{x} = (x_1, x_2, .., x_n)$**.** which may consist of several sentences. The output is another text sequence $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$**,**
- To facilitate the reasoning process, we resort to an external commonsense knowledge graph $G = (\mathcal{V}, \mathcal{E})$ **,** where $\mathcal{V}$ is the concept set and $\mathcal{E}$ is the relations connecting the concepts
- We extract a sub-graph G = (V, E) given the input text where $V \subset \mathcal{V}$ and $E \subset \mathcal{E}$ The sub-graph consists of inter-connected H-hop paths starting from the source concepts C_x extracted from the input text.

Problem formulation

- The task is then formulated as generating the best hypothesis $\hat{\boldsymbol{y}}$ which maximizes the following conditional probability:

Static Multi-Relational Graph Encoding

- We use N layers of GCN, follow Vashishth et al. (2020) and use a non-parametric compositional operation $\phi(\cdot)$ to combine the node embedding and the relation embedding.

$$\boldsymbol{o}_v^l = \frac{1}{|\mathcal{N}(v)|} \sum_{(u,r)\in\mathcal{N}(v)} \mathbf{W}_N^l \phi(\boldsymbol{h}_u^l, \boldsymbol{h}_r^l),$$

$$\boldsymbol{h}_v^{l+1} = \mathrm{ReLU}\Big(\boldsymbol{o}_v^l + \mathbf{W}_S^l \boldsymbol{h}_v^l\Big),$$

- We define the compositional operation as $\phi(\boldsymbol{h_u}, \boldsymbol{h_r}) = \boldsymbol{h}_u - \boldsymbol{h}_r$,the relation embeddung is also uodated with another linear function:

$$\boldsymbol{h}_r^{l+1} = \mathbf{W}_R^l \boldsymbol{h}_r^l.$$

- Finally ,we obtain node embeddings $\boldsymbol{h}_v^{L_G}$ and relation embeddings $\boldsymbol{h}_r^{L_G}$

Context Modeling with Pre-Trained Transformer

- We use a GPT-2 model to model the contextual dependency of the text sequence. The input to the model is the concatenation of the source and the target sequence: $\boldsymbol{s} = (x_1, x_2, \ldots, x_n, [BOS], y_1, y_2, .., y_m)$.
- The final hidden state at the t-th time step $\boldsymbol{h}_t^{L_D}$ is used used as the input to the multi-hop reasoning module.

Dynamic Multi-Hop Reasoning Flow

- we devise a dynamic reasoning module that utilizes both structural patterns of the knowledge graph and contextual information to propagate evidence along relational paths at each decoding step.
- the module broadcasts information on G by updating the score of outer nodes with their visited neighbours for multiple hops until all the nodes on G are visited. Initially, nodes correspond to the concepts in C_x are given a score of 1 while other nodes are assigned with 0 .

Dynamic Multi-Hop Reasoning Flow

- For unvisited node v, its node score $ns(v)$ is computed by aggregating evidence from $\mathcal{N}_{in}(v)$.

$$ns(v) = \underset{(u,r)\in\mathcal{N}_{in}(v)}{f}\Big(\gamma\cdot ns(u)+R(u,r,v)\Big), \quad (8)$$

- Where γ is a discount factor that controls the intensity of the information flow from the previous hops. $R(u,r,v)$ is the triple relevance that reflects the relevancy of the evidence given by the triplet under current context:

$$R(u,r,v) = \sigma(\boldsymbol{h}_{u,r,v}^{\mathrm{T}}\mathbf{W}_{sim}\boldsymbol{h}_t^{L_D}), \qquad (9)$$
$$\boldsymbol{h}_{u,r,v} = [\boldsymbol{h}_u^{L_G};\boldsymbol{h}_r^{L_G};\boldsymbol{h}_v^{L_G}]. \qquad (10)$$

After H hops, the final distribution over nodes is obtained by:

$$P(c_t|\boldsymbol{s}_{<t},G) = \mathrm{softmax}_{v\in V}(ns(v)),$$

Generation Distribution with Gate Control

- The final generation distribution combines the distribution over the concepts and the standard distribution over vocabulary. We use a soft gate probability g to decide whether to copy or generate:

$$g_t = \sigma\Big(\mathbf{W}_{gate}\boldsymbol{h}_t^{L_D}\Big).$$

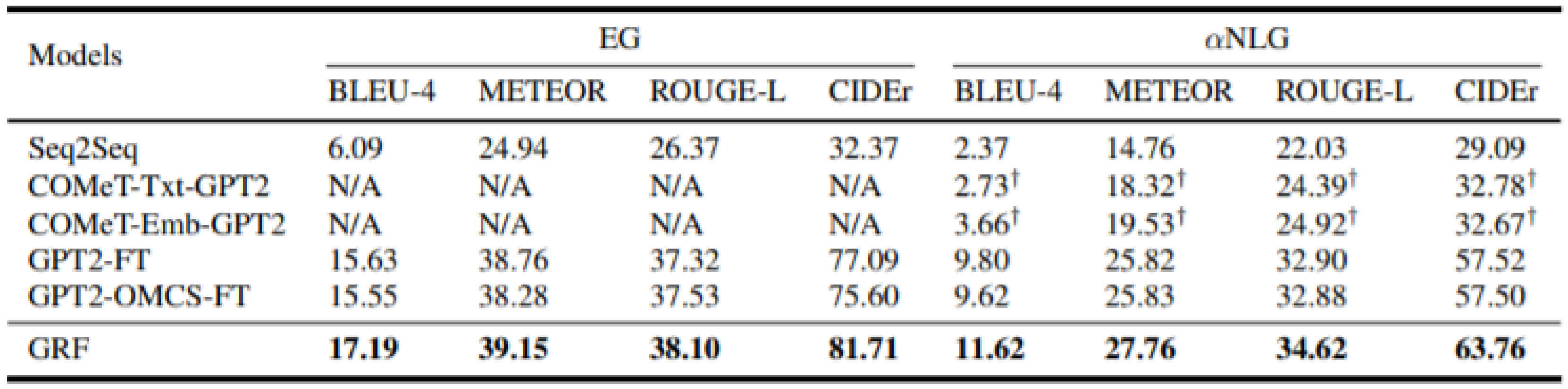
- The final distribution is:

$$\begin{aligned} P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x}, G) = {} & g_{t+N} \cdot P(c_{t+N}|\boldsymbol{s}_{<t+N}, G) \\ & + (1 - g_{t+N}) \cdot P(s_{t+N}|\boldsymbol{s}_{<t+N}), \end{aligned} \tag{13}$$

Experiments

- Task 1:Abductive NLG (αNLG) is to generate an explanatory hypothesis given two observations: O1 as the cause and O2 as the consequence. We use the official data split2
- Task 2:Explanation Generation (EG) is to generate an explanation given a counter-factual statement for sense-making

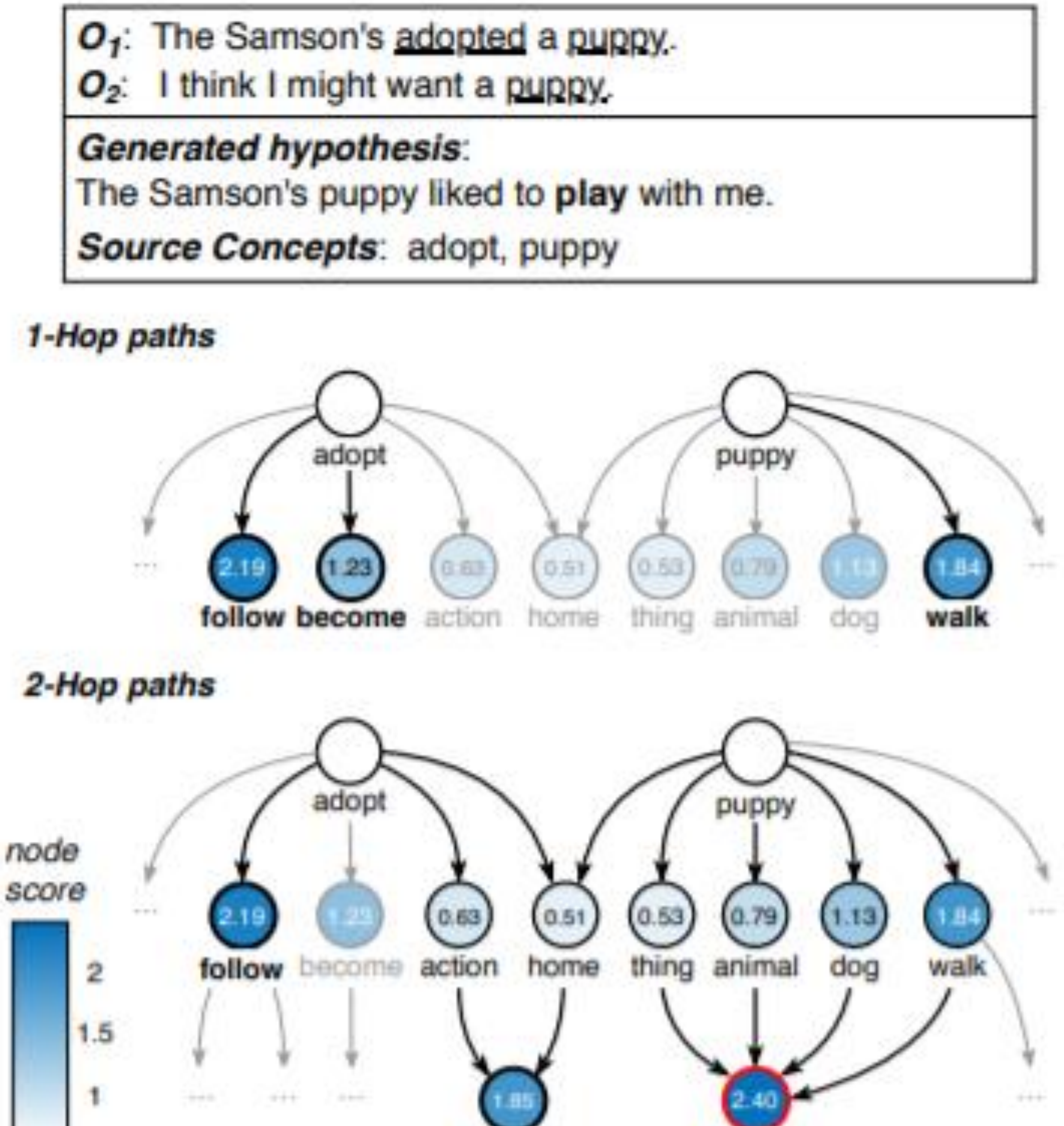
Experiments



Models	EG				αNLG			
	BLEU-4	METEOR	ROUGE-L	CIDEr	BLEU-4	METEOR	ROUGE-L	CIDEr
Seq2Seq	6.09	24.94	26.37	32.37	2.37	14.76	22.03	29.09
COMeT-Txt-GPT2	N/A	N/A	N/A	N/A	2.73†	18.32†	24.39†	32.78†
COMeT-Emb-GPT2	N/A	N/A	N/A	N/A	3.66†	19.53†	24.92†	32.67†
GPT2-FT	15.63	38.76	37.32	77.09	9.80	25.82	32.90	57.52
GPT2-OMCS-FT	15.55	38.28	37.53	75.60	9.62	25.83	32.88	57.50
GRF	**17.19**	**39.15**	**38.10**	**81.71**	**11.62**	**27.76**	**34.62**	**63.76**

Table 3: Automatic evaluation results on the test set of EG and αNLG. Entries with N/A mean the baseline is not designated for this task. $\dagger$: we use the generation results from Bhagavatula et al. (2020).

Experiments



O₁: The Samson's adopted a puppy.

O₂: I think I might want a puppy.

Generated hypothesis:

The Samson's puppy liked to **play** with me.

Source Concepts: adopt, puppy