겨울 세미나

Hallucination Mitigation in EMNLP 2023

서재형

2024.01.11

EMNLP 2023

메인 기준으로 Hallucination Paper는 10편 이상..?

그런데 대부분은 Evaluation, Detection, 그리고 벤치마크 생성에 집중함..

완화를 다루는 논문으로 3개 논문이 있었으며, 그 중에서 2개를 선택

Hallucination Mitigation in Natural Language Generation from Large-Scale Open-Domain Knowledge Graphs

Xiao Shi, Zhengyuan Zhu, Zeyu Zhang, Chengkai Li
Department of Computer Science and Engineering
The University of Texas at Arlington
Arlington, TX, USA

Overview

기존 그래프는 **small-scale, human-annotated datasets with limited variety of graph shapes**
→ **Graph-to-text task**에 있어서 **not more realistic large-scale open-domain settings**

제안 **1: GraphNarrative 데이터셋, Graph-to-Text for open-domain setting**
→ **Finetuned Transformers**가 **SOTA** 성능을 거두긴 했는데 여기서 **Hallucination** 문제 발견

제안 **2: Trimming the sentence to eliminate portions that are not present in corresponding graph using dependency parse tree**

Graph-to-Text

Graph to text generation task entails, given a subgraph $\mathcal{G} \subset G$ (a small fragment of triples), generating a token sequence $(y_1, \ldots, y_n)$ to describe G.

In graph-to-text generation, the preciseness and naturalness of the textual narration of graph fragments is important.

WebNLG, EventNarrative, TEKGEN.. 현실적인 세팅이 아니며, 상당히 단조로운 형태임

→ **This paper introduces GraphNarrative, a new dataset that fills the aforementioned gap between graph-to-text models and real-world needs.**

GraphNarrative

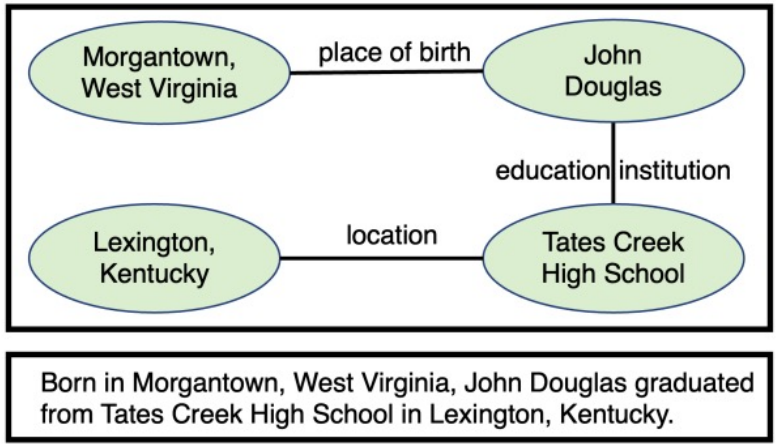

Figure 1: A graph-sentence pair in GraphNarrative

GraphNarrative consists of around 8.7 million (input graph, output text) pairs
→ **Wikipedia sentence**
→ **Freebase (2008)**의 **entities**와 **relationships**로 문장을 설명함.
→ 양이 더 많고 복잡하게 구성해서 더 **real-world**에 가깝다…
ex) 87만개의 페어, **7,920**개의 **topological shapes**를 지님. 그 중에서 **22%**는 **star shape**이며, 다른 **KG**에서는 **94, 96%**를 차지함.

How to make?

Wikipedia sentence W
the corresponding subgraph G in Freebase
to form a graph-sentence pair (G, W)

Step 1. Entity Linking
Mapping a span of tokens in W to an entity e in Freebase
- 4,408,115 (one-to-one mapping)
- Using coreference resolution (McCarthy and Lehnert, 1995), wikification (Csomai and Mihalcea, 2008), and Wikipedia-to-Freebase entity mapping

Step 2. Edge Detection
if Freebase contains **only one edge** between them, our simple method assumes the corresponding relationship is described in W

If Freebase has **multiple edges** between them, we include the edge whose **label tokens overlap** with W.. **그래도 남아있다**? we include the edge that is most **frequent** in Freebase.

특별한 점?

1) Scale and variety of entities and relations.
- GraphNarrative contains 8,769,634 graph-sentence pairs, 1,853,752 entities, 15,472,249 triples, and 1,724 relations from 84 Freebase domains

2) Linguistic variation
- A model to learn from many Wikipedia authors' diverse narrations

3) Graph structure complexity
- 7,920 distinct topological shapes based on graph isomorphism.
- Furthermore, only 22% of the instance graphs are star graphs

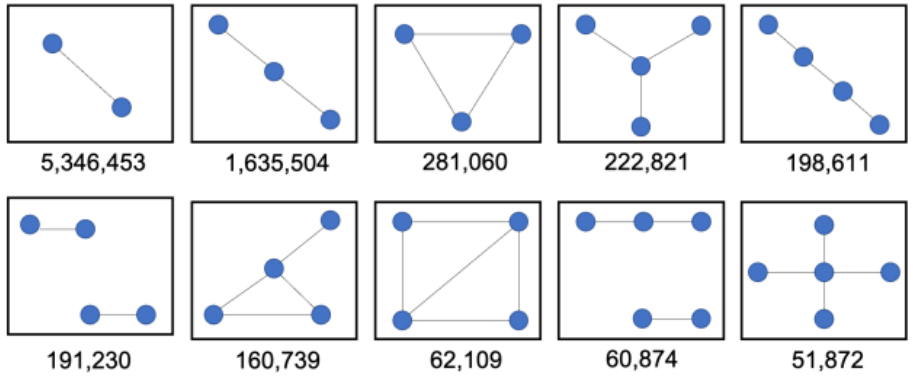

Figure 2: 10 most frequent graph shapes in GraphNarrative, with instance counts

Hallucination in G2T

Hallucination 발생!!
기존 **SOTA PLM**을 쓰니까 생기더라
트리플이 주어졌음에도 전혀 존재하지 않는 **(extrinsic) hallucination**이 발생

g: {(Neff Maiava, date of birth, 01 May 1924), (Neff Maiava, date of death, 21 April 2018)}
→ t: "Neff Maiava (1 May 1924 - 21 April 2018) was an Albanian actor"

문제**1**: 생성하라고도 안 했는데 만듦 **(Faithfulness hallucination)**
문제**2**: 사실이 아닌 것을 만들어 버림 **(Factuality hallucination)**

[IITP 2024 과제 계획서]
Faithfulness Hallucination(or intrinsic / user input-conflict / context-conflict) : User Input (e.g., instruction, task, retrieved knowledge, etc)에 대해서 주어진 정보와 불일치한 결과, 스스로 생성한 결과에 대해서 **inconsistency, instruction** 불이행 등

Factuality Hallucination(or extrinsic / fact-conflicting): User Input으로는 판단하기 어려운 신뢰할 수 없거나, 불확실한 정보, 실제 사실과는 다른 내용을 생성하는 결과 등

Hallucination in G2T

데이터 퀄리티가 영향을 주기는 함. **(Data-related)**

Hallucinated facts are seldom found in the clean, manually-crafted WebNLG but are present in automatically extracted graph-text pairs in TEKGEN due to extraction errors.

Data-related 해결책으로, **One is to improve our graph-text alignment method**
→ The graph extracted from a piece of text during alignment may miss certain entities or relationships due to either extraction errors or disparities between the text corpus and the knowledge graph.

그렇지만, **this method has an inherent limitation—since a knowledge graph in real-world is often far from complete**

기존엔 어떻게 했는가**?**
(1) further fine-tuning PLMs on WebNLG after fine-tuning on noisier automatically extracted datasets.

(2) filtering out training instances when the ROUGE-1 scores between the input and the output fall below a certain threshold

그러나, 명확히 환각을 줄일 수 있다는 검증은 없으며, 정량적으로 확인할 수도 없음.

Sentence Trimming

Algorithm 1: Sentence Trimming

Input: W: A co-reference resolved Wikipedia sentence; G: The graph for W based on graph-text alignment

Output: W_{trim}: The trimmed text sequence

$M \leftarrow \{\}$
foreach $(s, p, o) \in G$ **do**
  $s' \leftarrow s.remove(special\_tokens)$
  $o' \leftarrow o.remove(special\_tokens)$
  $W \leftarrow W.replace((s, o), (s', o'))$
  $M[s'], M[o'] \leftarrow s, o$
$W_{tree} \leftarrow W.dependency\_parsing()$
$min\_pos, max\_pos \leftarrow W.length, 0$
foreach $(s, p, o) \in G$ **do**
  $sdp \leftarrow shortest\_path(W_{tree}, s', o')$
  foreach $node \in sdp$ **do**
    $min\_pos \leftarrow \min(min\_pos, node.start)$
    $max\_pos \leftarrow \max(max\_pos, node.end)$
$W_{trim} \leftarrow W[min\_pos : max\_pos]$
foreach $k \in M.keys()$ **do**
  $W_{trim} \leftarrow W_{trim}.replace(k, M[k])$
return W_{trim}

W를 **subgraph** G에 없는 일부를 삭제하면서 **main idea**는 유지할 수 있도록 **Trimming**

1) **Parsing W to generate its dependency parse tree** W_{tree} **using spaCy**

2) **Each triple**에 대해서 **shortest dependency path (SDP) between** s **and** o

3) 모든 트리플 내의 **SDP**에서 모든 토큰 중에서 **sentence** W 기준으로 **leftmost position index (**$\min\_pos$**)**와 **rightmost position index (**$\max\_pos$**)**를 찾음

4) W는 시퀀스 형태로 $\min_pos$와 $\max\_pos$ 까지의 범위를 지니도록 **trimming**

Sentence Trimming

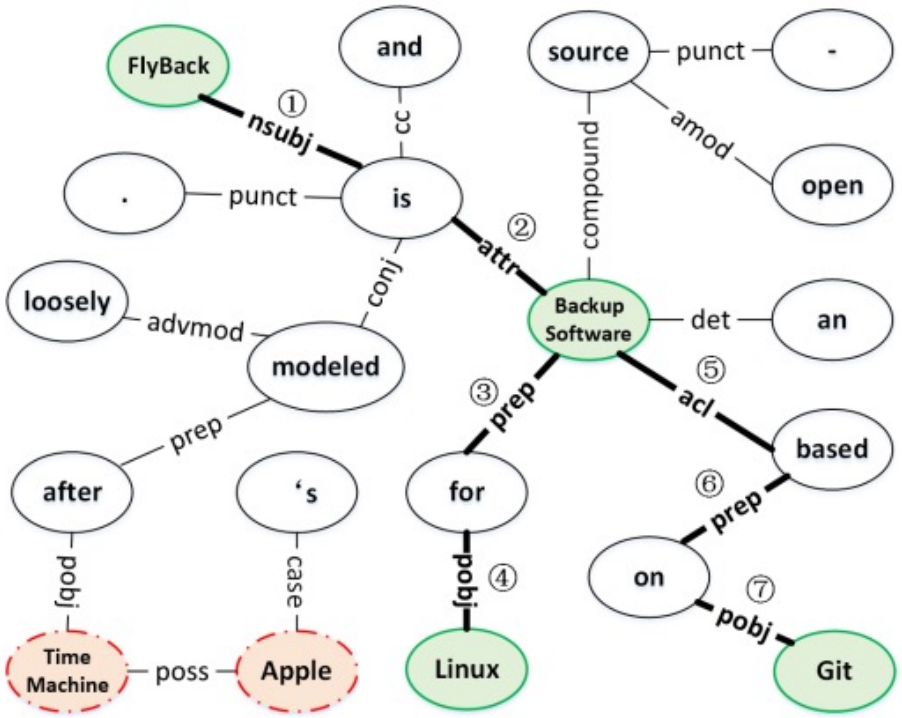

Figure 3: Dependency parse tree of sentence "FlyBack is an open-source Backup Software for Linux based on Git and modeled loosely after Apple's Time Machine."

W: FlyBack is an open-source Backup Software for Linux based on Git and modeled loosely after Apple's Time Machine.

G: {(FlyBack, software_genre, Backup Software), (FlyBack, operating_system, Linux),
(FlyBack, basis, Git)}

→ Apple과 Time machine은 이미 mapping 단계에서 사라짐.

SDPs: (1,2) (1,2,3,4), (1,2,5,6,7)

min_pos: Flyback
max_pos: Git

W_trim: FlyBack is an open-source Backup Software for Linux based on Git ~~and modeled loosely after Apple's Time Machine.~~

일반적인 DPT라면
Back Software도 분리한 head entity
그러나 여기서 적용한 DPT는 별도의 전처리 과정을 거쳐서 단일 토큰화

Results

ST	Hallucinated Entities	Missed Entities	Hallucinated Relations	Missed Relations	Grammar
w/o	1.163	0.003	1.340	0.040	4.793
w/	0.306	0.003	0.453	0.083	4.613

Table 2: Human evaluation of GraphNarrative quality

HE: 그래프에는 없었으나 문장으로 만들어진 엔티티 **(Factuality Hallucination)**
ME: 그래프에는 있었으나 문장에는 없는 엔티티 **(Faithfulness Hallucination)**
HR: 그래프에는 없었으나 문장에는 언급된 관계 **(Factuality Hallucination)**
MR: 그래프에는 있었으나 문장에는 없는 관계 **(Faithfulness Hallucination)**

G: 문장 당 평균 문법 오류 수
→ **a scale of 1-5: 5 (no errors), 4 (one error), 3 (two to three errors), 2 (four to five errors), and 1 (more than five errors).**

Results

Model	ST	BLEU			METEOR			chrF++		
		all	seen	unseen	all	seen	unseen	all	seen	unseen
BART-base	w/o	33.18	33.33	27.52	17.18	17.26	14.63	36.56	36.74	30.75
BART-base	w/	**46.49**	**46.77**	36.67	24.43	24.53	21.30	49.92	50.12	43.29
BART-large	w/o	32.35	32.48	27.56	17.45	17.53	15.07	37.12	37.29	31.58
BART-large	w/	46.04	46.18	40.98	24.35	24.41	22.17	49.69	49.85	44.72
T5-small	w/o	19.48	19.53	17.34	15.78	15.85	13.79	33.92	34.08	28.90
T5-small	w/	43.72	43.87	38.11	23.40	23.48	21.10	48.15	48.31	42.65
T5-base	w/o	16.89	16.95	14.63	16.23	16.30	14.10	35.37	35.54	29.84
T5-base	w/	42.18	42.29	37.85	24.20	24.27	21.94	49.63	49.80	44.18
T5-large	w/o	22.22	22.26	20.41	17.16	17.23	15.02	36.78	36.95	31.40
T5-large	w/	45.12	45.16	**43.40**	**24.77**	**24.84**	**22.54**	**50.44**	**50.60**	**45.21**

Table 3: Model performance on GraphNarrative

Critic-Driven Decoding for Mitigating Hallucinations in Data-to-text Generation

Mateusz Lango and **Ondřej Dušek**
Charles University, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Prague, Czech Republic

Overview

기존에 **NLI-based metrics (Honovich et al., 2021; Dušek and Kasner, 2020)** 중에서 **text classifiers**를 활용한 접근이 많았음.

그러나, **Hallucination** 정도를 **Detect**하거나 결과물을 **re-ranking**하는 것에 그침.

Combining the probabilistic output of a generator language model (LM) with the output of a special "text critic" classifier를 제안

→ Guiding the generation by assessing the match between the input data and the text generated without any changes to the underlying LM's architecture or training procedure

Critic-decoding

LM 모델의 조건부확률 계산식에 **Critic 'c'**를 포함하여 계산
The output of the critic c can be seen as a binary variable, equal to 1 if the text matches the input data and 0 otherwise

$$P(y|x,c) = \prod_{i=1}^{n} P(y_i|y_{\leq i-1}, x, c)$$

Generation of text 'y'는 **representation x**와 **critic c**를 바탕으로 함.
Using simple probability transformations을 통해서 아래의 독립변수의 확률 곱을 얻을 수 있음

$$P(y_i|y_{\leq i-1}, x, c) \propto P(c|y_{\leq i}, x)P(y_i|y_{\leq i-1}, x)$$

Left: The probability of the match between the text and the data as evaluated by the critic model

Right: The probability of a standard conditional LM

Critic-decoding

$$
\begin{aligned}
&P(y_i|y_{\leq i-1}, x, c) = \\
&\quad = \frac{P(y_i, y_{\leq i-1}, x, c)}{P(y_{\leq i-1}, x, c)} \\
&\quad = \frac{P(c|y_i, y_{\leq i-1}, x)P(y_i|y_{\leq i-1}, x)P(y_{\leq i-1}, x)}{P(y_{\leq i-1}, x, c)} \\
&\quad = P(c|y_{\leq i}, x)P(y_i|y_{\leq i-1}, x)\frac{P(y_{\leq i-1}, x)}{P(y_{\leq i-1}, x, c)} \\
&\quad = P(c|y_{\leq i}, x)P(y_i|y_{\leq i-1}, x)P(c|y_{\leq i-1}, x)^{-1} \\
&\quad \propto P(c|y_{\leq i}, x)P(y_i|y_{\leq i-1}, x)
\end{aligned}
$$

i th token y를 생성하는 경우, 이전 토큰인 **(i-1) th token y**까지의 값은 고정, **the critic's score for the previous tokens P(c|y≤i−1, x) is a constant** (독립임)

Critic-decoding

Binary classifier는 **each decoding step**마다 적용되며, 현재까지의 생성 결과가 타당하다고 가정하고 이후 생성에 대한 평가가 이루어짐. **Critic** 모델은 **LM** 모델의 훈련과는 별도로 훈련 가능함 **(**조건부 독립**).**
→ **P(c|y≤i, x)**에 근사할 수 있는 별도로 학습된 **Additional critic model**을 활용함.

Implementation operates on logarithms rather than raw probabilities and uses an additional weight λ
→ 가중치 조정을 통해서 **critic** 모델의 영향도를 조절하기 위하고 디코딩 단계에서 계산 단순화를 위함.

$$\ln P(y_i|y_{\leq i-1}, x, c)$$
$$\propto \lambda \ln P(c|y_{\leq i}, x) + \ln P(y_i|y_{\leq i-1}, x)$$

Critic-model Training

Positive instances for the critic's training are constructed from examples (x, y) in the underlying LM's dataset as prefixes: (x, y1),(x, y≤2),(x, y≤3), …,(x, y≤n).

Negative examples must be synthesized and are crucial for training the critic, as they teach it how to detect that the generated text starts deviating from the input data

1. base – the negative examples are generated with random words

   "The Cruises", "The A-Rosa operated", "The A-Rosa Luna located", ...

2. base with full sentences - a sentence or a token from the reference is replaced with random sentence/token and all possible negative examples are generated

   "The Cruises", "The Cruises Luna", "The Cruises Luna is", ..., "The A-Rosa operated", "The A-Rosa operated is", ...

3. vanilla LM – the incorrect next words are sampled from the five most probable tokens according to (unconditioned) LM

   "The United", "The A-Rosa is", "The A-Rosa Luna powers", ...

4. fine-tuned LM with full sentences – for a given data the NLG system generated the following output: "The A-Rosa Luna is 125.8m long and is powered by MTU Friedrichsburger", which is used to generate negative examples by comparing it against the reference

   "The A-Rosa Luna is 125.8m", "The A-Rosa Luna is 125.8m long", "The A-Rosa Luna is 125.8m and", "The A-Rosa Luna is 125.8m and is", ...

5. fine-tuned LM – the incorrect next words are sampled from the five most probable tokens according to data-conditioned LM

   "The A-Rosa Luna is 125.8m", "The A-Rosa Luna is supplied", "The A-Rosa Luna is powered with", ...

Results

critic model	accuracy	F1
1. base	0.969	0.970
2. base w/full sent.	0.984	0.975
3. vanilla. LM	0.931	0.798
4. fine-tuned LM	0.920	0.718
5. fine-tuned LM w/full sent.	0.929	0.714

Table 1: The classification performance of different

decoding approach	BLEU	MET EOR	BERT Score	NLI all	NLI ood	NLI ind	BLEURT all	BLEURT ood	BLEURT ind
baseline	45.09	0.373	0.911	0.841	0.783	0.889	0.128	-0.026	0.257
1. critic (base)	45.48	**0.377**	**0.913**	0.855	0.801	0.901	**0.155***	**0.010***	**0.277***
2. critic (base with full sentences)	44.90	0.371	**0.913**	**0.868***	**0.820***	**0.909**	0.153*	0.007*	0.274
3. critic (vanilla LM)	45.44	**0.377**	**0.913**	0.859*	0.811	0.900	0.139	-0.002	0.258
4. critic (fine-tuned LM)	45.41	0.373	0.911	0.834	0.772	0.886	0.128	-0.021	0.254
5. critic (fine-tuned LM w. full sentences)	**45.59**	0.374	0.912	0.839	0.779	0.889	0.136	-0.013	0.261

Table 2: Results of automatic evaluation on the WebNLG test set. NLI and BLEURT are reported for the whole test set (*all*) as well as separately for its out-of-domain (*ood*) and in-domain (*ind*) parts. "*" marks statistical significance at $\alpha = 0.05$ level (NLI: exact test for proportions, BLEURT: unpaired t-test).

Results

	BLEU	METEOR	BERTScore	NLI	BLEURT
baseline	11.74	0.149	0.775	0.748	-0.933
1. critic (base)	9.67	0.137	0.771	**0.796**	**-0.905**
2. critic (base with full sentences)	**11.88**	**0.151**	**0.776**	0.754	-0.920
3. critic (vanilla LM)	10.37	0.139	0.763	0.713	-0.980
4. critic (fine-tuned LM)	10.76	0.143	0.768	0.739	-0.964
5. critic (fine-tuned LM with full sentences)	11.41	0.149	0.771	0.712	-0.956

Table 3: Results of automatic evaluation on the OpenDialKG test set.

decoding approach	min. hal.	maj. hal.	omi.	disfl.	rep.	avg. rank
baseline	0.22	0.40	0.25	0.20	0.08	3.61
1. critic (base)	0.21	0.30	**0.20**	0.17	**0.04**	**3.38**
2. critic (base with full sentences)	0.21	**0.29**	0.27	**0.11**	0.08	3.43
3. critic (vanilla LM)	**0.18**	**0.29**	0.23	0.19	0.05	3.54
4. critic (fine-tuned LM)	0.22	0.37	0.26	0.21	0.07	3.53
5. critic (fine-tuned LM with full sentences)	0.20	0.37	0.26	0.18	0.07	3.54

Table 5: Results of manual evaluation on a sample of 100 examples from the WebNLG test set (percentage of examples with minor and major hallucinations, omissions, disfluencies, repetitions; average relative ranking).

Takeaways

1. Mitigation 연구 상태가…**?**
- 특정 벤치마크나 케이스 **/ LLM..? /** 평가 방법과 중복 **/** 정량평가와 **Confounder / 3**편 모두 **KG**

2. Data + Training <-> Inference
- 애초에 안 나오게 하는 연구**?** 나온 것을 고치는 연구

3. Mitigation은 아직 할 것이 많아 보임. 연구한다면, 정량화 방법을 반드시 고려