Ensuring Readability and Data-fidelity using Head-modifier Templates in Deep Type Description Generation

Jiangjie Chen, Ao Wang, Haiyun Jiang, Suo Feng, Chenguang Li and Yanghua Xiao*
School of Computer Science, Fudan University, Shanghai, China

Introduction

Motivations

1) Previous generative methods pay little attention to the grammatical structure, resulting in grammatical or factual mistakes.
2) Head-modifier rule. As a kind of short text, different type descriptions have similar grammatical structures and follows the head-modifier rule. E.g. **street** (head component) in **Paris, France** (modifier component); **library** in **New York, America**; **lake** in **Siberia, Russia**; etc.
3) Head-modifier template. Generalize infinite type descriptions into finite templates by replacing head and modifier words w/ placeholders (hed and mod).

Contributions

1) We propose a new head-modifier template-based method to improve the readability and data fidelity of generating type descriptions, which is also the first attempt of integrating head-modifier rule into neural generative models.
2) We apply copy and context gate mechanism to enhance the model's ability of choosing contents with the guidance of templates
3) We propose a new dataset with two new automatic metrics for this task. Experiments show that our method achieves SOTA performance on both datasets.

Solution

Template Annotation for Training

1) The **root** of a noun compound is always the head component in a head-modifier template.
2) Acquire templates using Dependency parsing technique w/ Stanford CoreNLP toolkit.

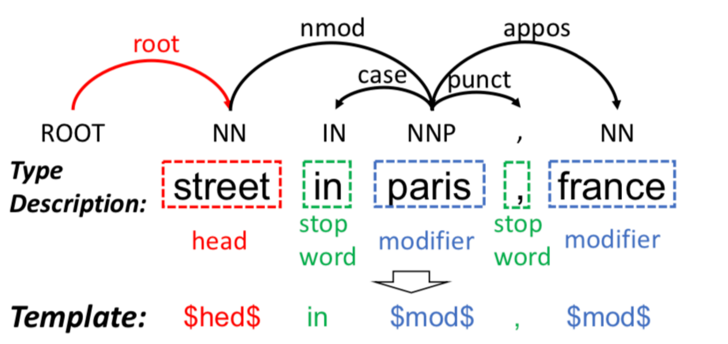

Figure 1: An example of extracting head-modifier template from type description by dependency parsing.

Two-stage Generation

1) In Stage 1, the model takes as input an infobox and generates a head-modifier template.
2) In Stage 2, the model takes as input the previously encoded infobox and the output template, and produces a type description.
   - Copy mechanism
   - Context gate mechanism

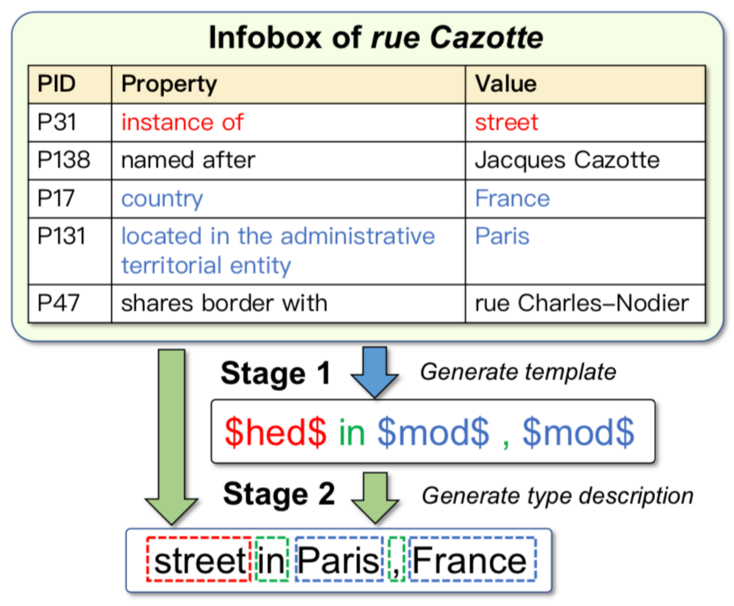

Figure 2: An example of the two-stage generation of our head-modifier template-based method. hed and mod are the placeholder for head and modifier components in the template.

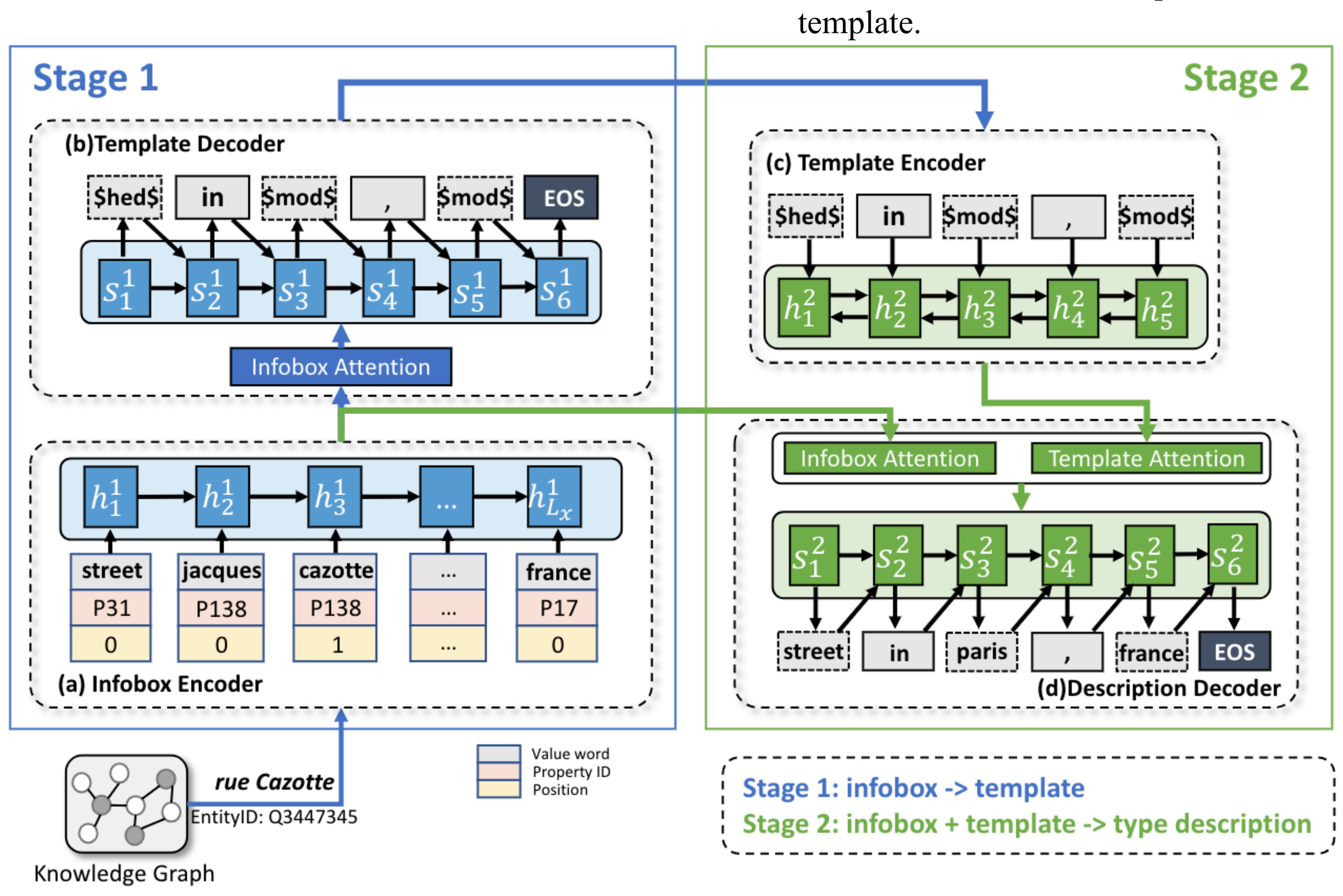

Figure 3: Overall architecture of our method. In Stage 1, the model generates a template from infobox of entity *rue Cazotte* (the entity can be found at Wikidata by EntityID), then in Stage 2 the model completes this template by reusing the infobox and generates a type description for this entity.

Loss of both stages

$$\mathcal{L} = -\sum_{i=1}^{L_t} \log p(t_i|t_{<i}, \mathcal{I}) - \sum_{i=1}^{L_y} \log p(y_i|y_{<i}, \mathcal{I}, \mathcal{T})$$

Evaluation

Dataset

- ***Wiki10K***: proposed by Bhowmik et al (2018), unevenly sampled from *Wikidata*.
- ***Wiki200K***: the proposed larger dataset, evenly sampled from *Wikidata*.

Datasets	Wiki10K	Wiki200K
# entities	10,000	200,000
# properties	480	900
vocabulary size	28,785	130,686
# *avg* statement	8.90	7.96
Copy(%)	88.24	71.30

Table 1: Statistics for both datasets. "Copy(%)" denotes the copy ratio in the golden type descriptions excluding stopwords, which is similar to the metric **ModCopy**.

Evaluation Metrics

- **Traditional:** BLEU-(1-2), ROUGE-L, METEOR, CIDEr
- **ModCopy (new):** Measures to what extent the informative words are preserved in the modifiers of model's output.
- **HedAcc (new):** Measures model's ability of predicting the right type of the entity.

Baseline Models

Models	Description
AttnS2S	Seq2Seq w/ attention mechanism (Luong et al., 2015).
Ptr-Gen	Pointer-Generator (See et al., 2017), baseline of copy mechanism. We further construct it into a non-template version of our method.
Tranformer	Competitive baseline in Seq2Seq task (Vaswani et al., 2017).
DGN	Previously SOTA method regarding this task (Bhowmik et al., 2018).

Automatic Evaluation

Wiki10K							
Model	B-1	B-2	RG-L	METEOR	CIDEr	ModCopy	HedAcc
AttnS2S	53.96	47.56	55.25	29.95	2.753	69.45	52.82
Ptr-Gen	64.24	57.11	65.37	36.42	3.536	83.88	67.92
Transformer	61.63	54.93	63.14	35.01	3.400	75.37	61.13
DGN	63.24	57.52	64.50	35.92	3.372	77.53	64.65
Our work	**65.09**	**58.72**	**66.92**	**37.55**	**3.717**	**86.04**	**70.68**
Wiki200K							
Model	B-1	B-2	RG-L	METEOR	CIDEr	ModCopy	HedAcc
AttnS2S	66.15	61.61	70.55	37.65	4.105	49.59	79.76
Ptr-Gen	70.13	66.21	75.21	41.38	4.664	**58.27**	85.38
Transformer	69.78	66.07	75.60	41.52	4.654	53.85	85.55
DGN	62.60	57.86	69.30	34.84	3.815	48.30	81.31
Our work	**73.69**	**69.59**	**76.77**	**43.54**	**4.847**	58.14	**85.81**

Table 2: Evaluation results of different models on both datasets.

Human Evaluation

- Evaluates Readability based on Grammar Accuracy (G.A.) and Overall Accuracy (O.A.)

Model	*G.A.*	*O.A.*	M.C.	H.A.
AttnS2S	92.25	50.50	51.53	80.27
Ptr-Gen	90.00	65.00	**62.50**	88.01
Transformer	95.25	58.00	55.70	89.67
DGN	89.50	56.00	47.29	81.37
Our work	**96.50**	**66.25**	61.32	**90.29**

Table 3: Results of manual evaluation as well as two proposed metrics.

Effects of Templates

Entity ID: Q859415	Entity ID: Q18758590
Gold: **commune** in paris, france	Gold: italian **architect** and **teacher**
Template 1: hed in mod, mod	Template 1: mod hed and hed
Output 1: **commune** in paris, france	Output 1: italian **architect** and *architect*
Template 2: mod hed	Template 2: mod hed
Output 2: **commune** in france	Output 2: italian **architect**
Template 3: hed mod	Template 3: hed mod and mod
Output 3: **commune**	Output 3: *italy* and **teacher**

Table 4: Examples of replacing templates. Template 1's are the inital generated templates, while the remaining ones are produced by the authors. We use **bold** to denote the heads and use *italic red* to denote mistaken words.

Conclusion

1) We propose a head-modifier template-based type description generation method, with a larger dataset and two metrics designed for this task.
2) Experimental results demonstrate that our method achieves state-of-the-art performance over baselines on both datasets while ensuring data fidelity and readability in generated type descriptions.
3) Further experiments regarding the effect of templates show that our model is not only controllable through templates, but resilient against wrong templates and able to correct itself.

Future Work

Aside from such syntax templates, in the future, we aim to explore how semantic templates contribute to type description generation.