Conditioning Language Models for Image Paragraph Captioning

Yan Pan

Mentor: Prof. Louis-Philippe Morency, CMU LTI

Introduction

In the image paragraph captioning task, the model is given an image and is asked to output a descriptive paragraph of the image that describes the main objects in this image, including their attributes and relationships. While there are vision models that can detect the objects in an image accurately, it is a challenging task for the model to generate a coherent, well-organized long paragraph. We study how we can condition visual information for pretrained language models such as GPT-2, that are traditionally conditioned on texts, to utilize their power of generating long text.

Method

We propose a simple yet effective method that combines scene graph and GPT-2. The scene graph of an image is a graph, where the vertices are the objects, and the edges tells the relationship between any pair of objects. There are models that can generate scene graphs accurately, such as Graph R-CNN [4].

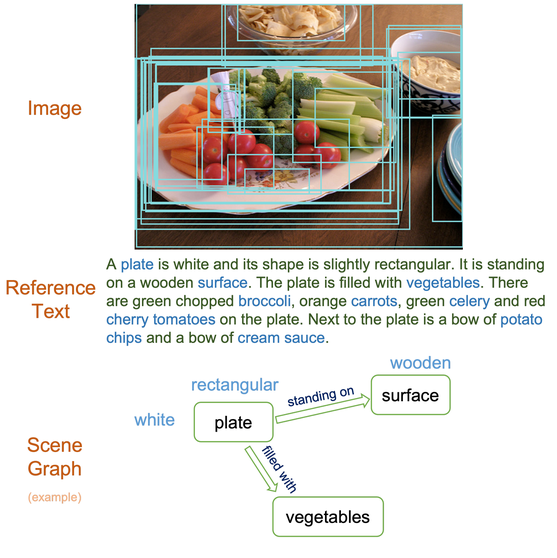

Figure 1:An example of image paragraph captioning, including images with bounding boxes, reference text, and scene graph.

We can convert the scene graphs into keywords by concatenating all the relationships, and then train a GPT-2 to generate text based on the keywords.
To further improve the result, we separate the training and testing stage for the model. Instead of training GPT-2 to learn the output, we train it to be a fill-in-the-blank language model.

- **Training:** We adapt the ProGen method of [1], that generates keywords from the reference text using TF-IDF. Then, the GPT-2 model is trained to fill in the blanks of the keywords to match the reference text.
- **Testing:** The GPT-2 model will fill in the blanks of the keywords extracted from scene graphs.

Experiments

We test our method on the VisualGenome image paragraph captioning dataset. We run experiments with our method with and without separating training and testing, as well as a baseline methods of [2] that maps visual features to text space. We also compare our result to the state-of-the-art result of [3] in terms of METEOR score on the test dataset. The result is shown in Table 1. Our result are close to the state-of-the-art model in METEOR score.

Model & Parameter	METEOR
Frozen [2]	0.073474
CAE-LSTM [3]	**0.1882**
SG	0.135464
ProGen+SG (0.3)	**0.181274**
ProGen+SG (0.5)	0.176456
ProGen+SG (0.7)	0.170432
ProGen+SG (0.85)	0.163415

Table 1:Comparison of METEOR scores for different methods, where ProGen is for training with [1], SG is for scene graph, the number in parenthesis is the rate for ProGen.

Effect of ProGen Rate: The rate decides the threshold of which words are classified as keywords. With larger rate, the keywords will be longer, resulting in more details from scene graphs, but will make generation of text harder. This is a trade-off between "accurate description" and "better text style."

Better Keyword Extraction: The main problem with using different training and testing stage is that the model might suffer distribution shift, that the distribution of keywords in training and testing are different. There are some heuristics to improve the keyword extraction. Taking a smaller subgraph of the scene graph can filter out unnecessary objects. Removing some duplicate texts about the same object also improves the result.

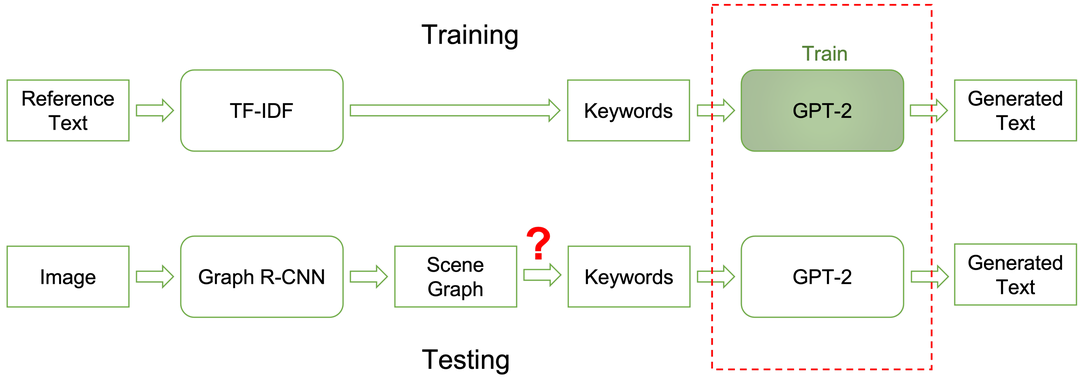

Figure 2:Flow diagram of our proposed method.

Output: A man's neck, and the lower portion of his face are visible in this image. The man is wearing a black suit, and a purple tie. The man has a slight smirk on his face, and he is sitting in front of a white wall.

Reference: The image is a portrait of a man. The man is wearing a gray suit jacket. The shirt is blue. He is wearing a patterned necktie. The man has grey and brown hair. He is smiling at the camera. The man is wearing eyeglasses. On his finger, there are two rings. The man is standing behind a blue leather chair. The background is gray.

Figure 3:An example of generated text and reference text.

Conclusion

We propose a new method that combines scene graphs and GPT-2 used as fill-in-the-blank language models. We conduct experiments on the VisualGenome dataset and compare the performance of several different methods, and show that our method achieves top performance in METEOR scores.

Acknowledgements

I sincerely thank my collaborators Yudong Liu and Paul Liang for their help in this project.

References

[1] B. Tan, Z. Yang, M. Al-Shedivat, E. Xing, and Z. Hu. Progressive generation of long text with pretrained language models. In *NAACL-HLT*, pages 4313–4324, 2021.

[2] M. Tsimpoukelli, J. L. Menick, S. Cabi, S. Eslami, O. Vinyals, and F. Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

[3] J. Wang, Y. Pan, T. Yao, J. Tang, and T. Mei. Convolutional auto-encoding of sentence topics for image paragraph generation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 940–946, 2019.

[4] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh. Graph r-cnn for scene graph generation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 670–685, 2018.