Scene Graph Parsing by Attention Graph

Martin Andrews
Red Dragon AI
Singapore
martin@reddragon.ai

Yew Ken Chia
Red Dragon AI
Singapore
ken@reddragon.ai

Sam Witteveen
Red Dragon AI
Singapore
sam@reddragon.ai

Abstract

Scene graph representations, which form a graph of visual object nodes together with their attributes and relations, have proved useful across a variety of vision and language applications. Recent work in the area has used Natural Language Processing dependency tree methods to automatically build scene graphs.

In this work, we present an 'Attention Graph' mechanism that can be trained end-to-end, and produces a scene graph structure that can be lifted directly from the top layer of a standard Transformer model.

The scene graphs generated by our model achieve an F-score similarity of 52.21% to ground-truth graphs on the evaluation set using the SPICE metric, surpassing the best previous approaches by 2.5%.

1 Introduction

In recent years, there have been rapid advances in the capabilities of computer systems that operate at the intersection of visual images and Natural Language Processing - including semantic image retrieval [1, 2], image captioning [3–6], visual question answering [7–9], and referring expressions [10–12].

As argued in [1], and more recently [13], encoding images as scene graphs (a type of directed graph that encodes information in terms of objects, attributes of objects, and relationships between objects) is a structured and explainable way of expressing the knowledge from both textual and imaged-based sources, and is able to serve as an expressive form of common representation. The value of such scene graph representations has already been proven in a range of visual tasks, including semantic image retrieval [1], and caption quality evaluation [14].

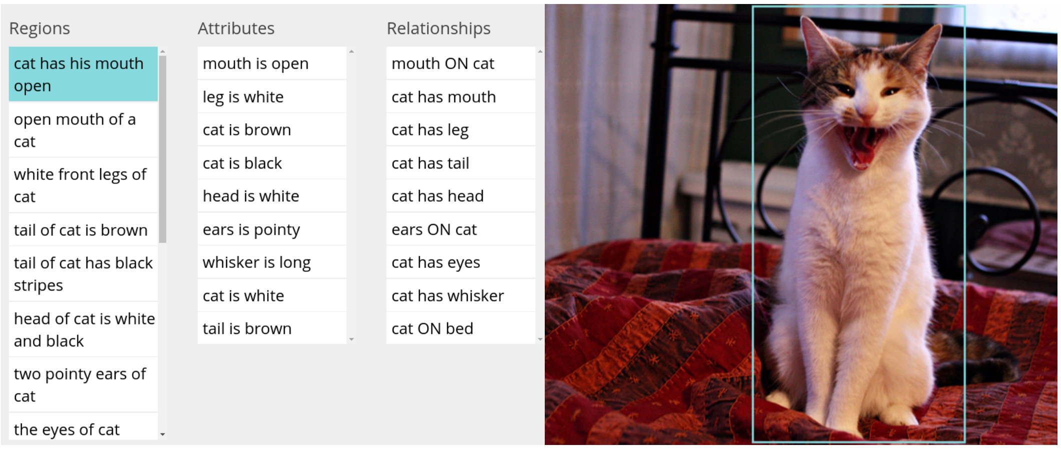
One approach to deriving scene graphs from captions / sentences is to use NLP methods for dependency parsing. These methods extend the transition-based parser work of [15], to embrace more complex graphs [16], or more sophisticated transition schemes [13].

Recently, an alternative to the sequential state-based models underlying transition-based parsers has gained popularity in general NLP settings, with the Transformer model of [17] leading to high performance Language Models [18], and NLP models trained using other, innovative, criteria [19].

In this paper, we suppliment a pre-trained Transformer model with additional layers that enable us to 'read off' graph node connectivity and class information directly. This allows us to benefit from recent advances in methods for training Language Models, while building a task-specific scene graph creation model. The overall structure allows our graph elements to be created 'holistically', since the nodes are output in a parallel fashion, rather than through stepwise transition-based parsing.

Based on a comparison with other methods on the same Visual Genome dataset [20] (which provides rich amounts of region description - region graph pairs), we demonstrate the potential of this graph-building mechanism.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.



Regions

cat has his mouth open
open mouth of a cat
white front legs of cat
tail of cat is brown
tail of cat has black stripes
head of cat is white and black
two pointy ears of cat
the eyes of cat

Attributes

mouth is open
leg is white
cat is brown
cat is black
head is white
ears is pointy
whisker is long
cat is white
tail is brown

Relationships

mouth ON cat
cat has mouth
cat has leg
cat has tail
cat has head
ears ON cat
cat has eyes
cat has whisker
cat ON bed

Figure 1: Example from data exploration site for [20]. For this region, possible graph objects would be {cat, mouth}, attributes {brown←cat, black←cat, white←cat open←mouth}, and relationships {cat←has←mouth, mouth←ON←cat}.

2 Region descriptions and scene graphs

Using the same notation as [13], there are three types of nodes in a scene graph: object, attribute, and relation. Let $\mathcal{O}$ be the set of object classes, $\mathcal{A}$ be the set of attribute types, and $\mathcal{R}$ be the set of relation types. Given a sentence s, our goal in this paper is to parse s into a scene graph:

$$G(s) = \langle O(s), A(s), R(s) \rangle \tag{1}$$

where $O(s) = \{o_1(s), \ldots, o_m(s)\}, o_i(s) \in \mathcal{O}$ is the set of object instances mentioned in the sentence s, $A(s) \subseteq O(s) \times \mathcal{A}$ is the set of attributes associated with object instances, and $R(s) \subseteq O(s) \times \mathcal{R} \times O(s)$ is the set of relations between object instances.

To construct the graph $G(s)$, we first create object nodes for every element in $O(s)$; then for every (o, a) pair in $A(s)$, we create an attribute node and add an unlabeled arc $o \leftarrow a$; finally for every (o_1, r, o_2) triplet in $R(s)$, we create a relation node and add two unlabeled arcs $o_1 \leftarrow r$ and $r \leftarrow o_2$.

2.1 Dataset pre-processing and sentence-graph alignment

We used the same dataset subsetting, training/test splits, preprocessing steps and graph alignment procedures as [13], thanks to their release of runnable program code[1].

2.2 Node labels and arc directions

In this work, we use six node types, which can be communicated using the CONLL file format:

- SUBJ The node label for an object in $\mathcal{O}$ (either standalone, or the subject of a relationship). The node's arc points to a (virtual) ROOT node
- PRED The node label for a relationship $\mathcal{R}$, The node's arc points to SUBJ
- OBJT The node label for an object in $\mathcal{O}$ that is the grammatical object of a relationship, where the node's arc points to the relevant PRED
- ATTR The arc label from the head of an object node to the head of an attribute node. The node's arc points to an object in $\mathcal{O}$ of node type SUBJ or OBJT
- same This label is created for nodes whose label is a phrase. For example, the phrase "in front of" is a single relation node in the scene graph. The node's arc points to the node with which this node's text should be simply concatenated
- none This word position is not associated with a node type, and so the corresponding model output is not used to create an arc

[1] Upon publication, we will also release our code, which includes some efficiency improvements for the preprocessing stage, as well as the models used.

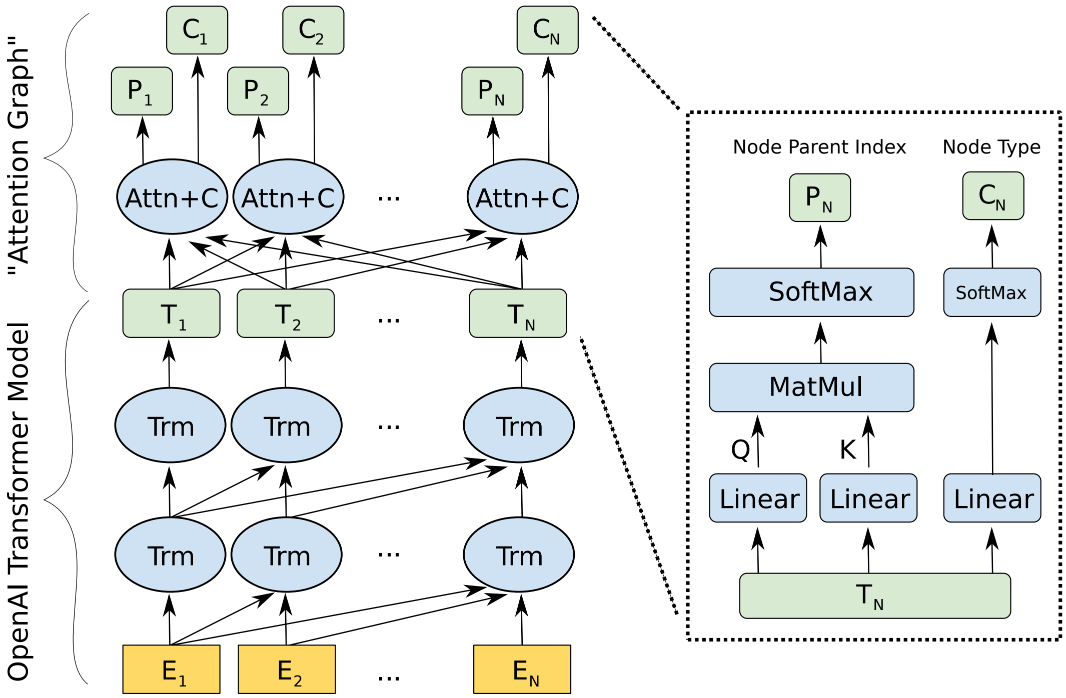


Figure 2: Model architecture illustrating Attention Graph mechanism

3 Attention Graph Model

The OpenAI Transformer [18] Language Model was used as the foundation of our phrase parsing model (see Figure 2). This Transformer model consists of a Byte-Pair Encoded subword [21] embedding layer followed by 12-layers of "decoder-only transformer with masked self-attention heads" [17], pretrained on the standard language modelling objective on a corpus of 7000 books.

The Language Model's final layer outputs were then fed in to a customised "Attention Graph" layer, which performed two functions : (i) classifying the node type associated with each word; and (ii) specifying the parent node arc required *from* that node.

The Attention Graph mechanism is trained using the sum of two cross-entropy loss terms against the respective target node types and parent node indices, weighted by a factor chosen to approximately equalise the contributions to the total loss of the classification and Attention Graph losses. For words where the target node type is `none` (e.g: common conjunctions), the cross-entropy loss due to that node's connectivity is multiplied by zero, since its parent is irrelevant.

To convert a given region description to a graph, the BPE-encoded form is presented to the embedding layer E_i in Figure 2, and the node types and node arc destinations are read from C_i and P_i respectively. No post-processing is performed : If the attention mechanism suggests an arc that is not allowed (e.g.: an `OBJT` points to a word that is not a `PRED`) the arc is simply dropped.

4 Experiments

We train and evaluate our scene graph parsing model on (a subset of) the Visual Genome [20] dataset, in which each image contains a number of regions, with each region being annotated with a region description and a (potentially empty) region scene graph. Our training set is the intersection of Visual Genome and MS COCO [22] `train2014` set (34,027 images & 1,070,145 regions), with evaluations split according to the MS COCO `val2014` set (17,471 images & 547,795 regions).

We also tested the performance of the 'Oracle' (an algorithmic alignment system between the region descriptions and the ground-truth graph tuples) - including a regime where the number of tuples was limited to the number of words, excluding `{a, an, the, and}`, in the region description.

Table 1: SPICE metric scores for the Oracle (using code released by [13]) and our method, under the base assumptions, and also where the number of tuples is bounded above by the number of potentially useful words in the region description

Parser	F-score reported in [13]	F-score (our tests)	F-score (limited tuples)
Attn. Graph (ours)		0.5221	**0.5750**
Oracle	0.6985	0.6630	**0.7256**

Table 2: SPICE metric scores between scene graphs parsed from region descriptions and ground truth region graphs on the intersection of Visual Genome [20] and MS COCO [22] validation set.

Parser	F-score
Stanford [23]	0.3549
SPICE [14]	0.4469
Custom Dependency Parsing [13]	0.4967
Attention Graph (ours)	**0.5221**
Oracle (as reported in [13])	0.6985
Oracle (as used herein)	0.6630

The model Q and K vectors were of length 768, consistent with the rest of the Transformer model. We use an initial learning rate of 6.25×10^{-5} and Adam optimizer [24] with b_1/b_2 of $0.9/0.999$ respectively. Training was limited to 4 epochs (about 6 hours on a single Nvidia Titan X).

5 Results

The F_1 scores given in Table 1 indicate that there might be significant room for improving the Oracle (which, as the provider of training data, is an upper bound on any trained model's performance). However, examination of the remaining errors suggests that an F_1 near 100% will not be achievable because of issues with the underlying Visual Genome dataset. There are many instances where relationships are clearly stated in the region descriptions, where there is no corresponding graph fragment. Conversely, attributes don't appear to be region-specific, so there are many cases (as can be seen in Figure 1) where a given object (e.g. 'cat') has many attributes in the graph, but no corresponding text in the region description.

Referring to Table 2, our Attention Graph model achieves a higher F_1 than previous work, despite the lower performance of the Oracle used to train it[2]. The authors also believe that there is potential for further gains, since there has been no hyperparameter tuning, nor have variations of the model been tested (such as adding extra fully bidirectional attention layers).

6 Discussion

While the Visual Genome project is inspirational in its scope, we have found a number of data issues that put a limit on how much the dataset can be relied upon for the current task. Hopefully, there are unreleased data elements that would allow some of its perplexing features to be tidied up.

The recent surge in NLP benchmark performance has come through the use of large Language Models (trained on large, unsupervised datasets) to create contextualised embeddings for further use in downstream tasks. As has been observed [25], the ability to perform transfer learning using NLP models heralds a new era for building sophisticated systems, even if labelled data is limited.

The Attention Graph mechanism, as introduced here, also illustrates how NLP thinking and visual domains can benefit from each other. Although it was not necessary in the Visual Genome setting, the Attention Graph architecture can be further extended to enable graphs with arbitrary connectivity to be created. This might be done in several distinct ways, for instance (a) Multiple arcs could leave each node, using a multi-head transformer approach; (b) Instead of a SoftMax single-parent output P_i, multiple directed connections could be made using independent ReLU weight-factors; and (c) Potentially untie the correspondence that the Transformer has from each word to nodes, so that it becomes a Sequence-to-Graph model. Using attention as a way of deriving structure is an interesting avenue for future work.

[2] This needs further investigation, since the Oracle results are a deterministic result of code made available by the authors of [13]

References

[1] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3668–3678, 2015.

[2] Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. *arXiv preprint arXiv:1511.06361*, 2015.

[3] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*, 2014.

[4] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. Cs231n: Convolutional neural networks for visual recognition. *University Lecture*, 2015.

[5] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[6] Chenxi Liu, Junhua Mao, Fei Sha, and Alan L Yuille. Attention correctness in neural image captioning. In *AAAI*, pages 4176–4182, 2017.

[7] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[8] Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4995–5004, 2016.

[9] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 39–48, 2016.

[10] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4555–4564, 2016.

[11] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11–20, 2016.

[12] Chenxi Liu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, and Alan L Yuille. Recurrent multimodal interaction for referring image segmentation. In *ICCV*, pages 1280–1289, 2017.

[13] Yu-Siang Wang, Chenxi Liu, Xiaohui Zeng, and Alan Yuille. Scene graph parsing as dependency parsing. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 397–407. Association for Computational Linguistics, 2018. doi: 10.18653/v1/N18-1037. URL `http://aclweb.org/anthology/N18-1037`.

[14] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *European Conference on Computer Vision*, pages 382–398. Springer, 2016.

[15] Eliyahu Kiperwasser and Yoav Goldberg. Simple and accurate dependency parsing using bidirectional lstm feature representations. *arXiv preprint arXiv:1603.04351*, 2016.

[16] Peng Qi and Christopher D Manning. Arc-swift: A novel transition system for dependency parsing. *arXiv preprint arXiv:1705.04434*, 2017.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[18] Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. *Technical report, OpenAI*, 2018.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[20] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. 2016. URL `https://arxiv.org/abs/1602.07332`.

[21] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[23] Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D Manning. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In *Proceedings of the fourth workshop on vision and language*, pages 70–80, 2015.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Sebastian Ruder. *NLP's ImageNet moment has arrived*, 2018 (accessed November 1, 2018). URL `https://thegradient.pub/nlp-imagenet/`.