# SelfGraphVQA: A Self-Supervised Graph Neural Network for Scene-based Question Answering
## Supplementary Material


Bruno Souza*
University of Campinas
b234837@dac.unicamp.br

Marius Aasan
University of Oslo
mariuaas@uio.no

Helio Pedrini
University of Campinas
helio@ic.unicamp.br

Adín Ramírez Rivera
University of Oslo
adinr@uio.no


Table A.1: Detailed statistics for the GQA dataset examined in our study compared to other possible statistics and the original paper dataset.

| | Answers Candidates |
|---|---|
| Ours | 1878 |
| Alternative [1, 10, 11] | 1533 |
| Original [3, 5] | 1878 |

## A. Datasets

We evaluate our SelfGraphVQA frameworks on the GQA dataset [5]. GQA is another large-scale effort (22M questions, each with one answer) that focuses on the compositionality of template-generated questions for real-world images. We use the official train/validation split of GQA.

The GQA dataset was selected for evaluation because it includes complex relational and spatial questions that require multiple reasoning skills, spatial understanding, and multi-step inference. These characteristics make it more challenging compared to previous visual question-answering datasets. Consequently, the GQA dataset is well-suited for evaluating the performance of scene graph models.

In contrast to prior studies [1, 11], our approach takes a simplistic approach by considering solely the ground truth distribution as a potential answer in the training dataset as a candidate for the answer distribution, without any filter techniques. Table A.1 provides detailed statistics for each dataset examined in our investigation.

Despite the substantial variations in the answering classes, we emphasize that our method proves to be effective and comparable to other existing approaches. In addition to the aforementioned points, this further highlights the fact that VQA is a complex and expansive challenge that lends itself to various approaches and needs continued exploration and refinement.

*Work carried out as Guest Researcher at UiO.

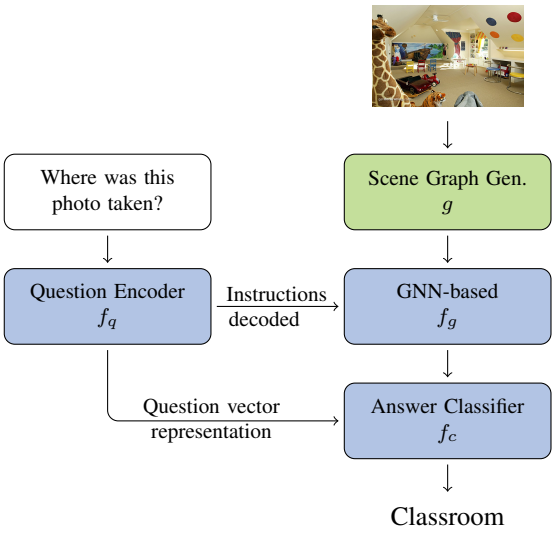


Figure B.1: The baseline architecture.

## B. Baseline Architecture

Figure B.1 depicts the overall components of our baseline architecture. The unbiased pre-trained scene graph generator model utilized in this study originates from the research paper authored by Knyazev et al. [7]. Applying a density-normalized edge loss to the model, the authors contend that the model is aware of the graph density and, therefore, generalizes better even to rare compositions.

In our project, the frozen weights pre-trained Scene Graph Generator takes the image information and generates a scene graph representation. The Question Encoder receives the instructions and provides them to the GNN-based encoder. Each layer of the module pays attention to these instructions in order to update its hidden node states. The Classifier then takes the graph representation and the question vector concatenates them, and predicts the correct answer.

We use the similar architecture of the state-of-the-art

graph-based GraphVQA model [8] and LRTA [9] over the GQA dataset as a baseline for our experiments, with some modifications in order to reduce the dependence on the annotated available data, as we aim to mitigate the limitations imposed by data availability and enhance the model's generalizability.

For practical purposes, the functional program instructions accompanying each question in the GQA dataset [5] are not necessarily available for inference on real-world data, so we train our decoder to decode the instructions from the question itself. These additional labels are processed by the reasoning module in the GraphVQA model which we explicitly omit in our baseline, as we are more interested in generalizability and real-world performance rather than expressively *solving* the GQA dataset.

In addition, we omit the pre-processing using the scene graph encoding module of the original GraphVQA, as the scene graph generation model $g$ was selected to extract high quality SG-representations. Here, our $f_g$ module is a graph attention network (e.g. GAT) [13].

In the GloVE embedding design, both the query encoder $f_q$ and the graph encoder $f_g$ designs are shared between the original baseline and our proposed modified model. Whereas in the BERT design, we only take the similarity of the graph encoder module $f_g$ design, as our query encoder $f_q$ and the language embedding is a BERT model. By adapting the similar SoTA architecture strategy to the specific design choices of each model, we aim to evaluate the performance and effectiveness of our proposed approach.

## C. Architecture Details

Within this section, we aim to provide additional details regarding all components of our implementation approaches.

To ensure clarity and facilitate better comprehension, we have divided this section into two subsections: one discussing the utilization of GloVE word embedding along with a transformer-based model for the question encoder, and the other focusing on the application of BERT for word embedding and the question encoder.

Table C.1 provides a comprehensive overview of the two approaches employed in this study.

It is worth mentioning that the scene graph generator module has its weights frozen in all training approaches, except when we employ the Distribution Link Representation Regularization technique.

### C.1. GloVE Word Embedding and Transformer-based Question Encoder

The images are fed through a pre-trained scene graph generator $g$ from [7] work that generates scene graphs from images on the fly.

Except for the pooled graph-level representation (i.e., the module that feeds the classifier), which has a dimension size of 512, all node and edge features have dimension size 300.

The word embedding for the transformed-based query encoder module $f_q$ has its initial weights initialized by using embeddings from GloVe [12]. Both hidden states and word embedding vectors have a dimension size of 300. The question representation is produced by the transformed-based question encoder.

Following [8, 9] work, we adopt a hierarchical sequence generation design, i.e., a transformer decoder model first parses the question into a sequence of $M$ instruction vectors, $[i_1, i_2, \ldots, i_M]$. The $i$-th instruction vector will correspond exactly to the $i$-th execution step processed by the GNN encoder $f_g$ module. In our experiments, we force $M$ equals five. We note that SelfGraphVQA does not require any explicit supervision on how to solve the instruction step from the question, and we only supervise the final answer prediction.

For the un-normalized contrastive approach, the MLP prediction head $h$ plays a crucial role in our model architecture. It comprises three fully connected layers, each followed by batch normalization and ReLU activation, except for the final layer. This setup ensures non-linearity and facilitates effective feature extraction. It is important to note that the MLP prediction head is exclusively utilized during the training phase and is subsequently discarded during inference, which aligns with prevailing practices in contemporary self-supervised training methods [2, 4].

The classification module $f_c$ is another integral component of our model. It is designed as a two-layer MLP with a dropout rate of $0.2$ and ELU activation.

As explained in Section 3, we independently apply the three self-supervised losses (i.e., local similarity, global similarity, and regularization for permutation equivariance) and compared performances. Our experimental choices were designed to minimize possible biases in the evaluation of our proposed framework.

Both anchored and augmented scene graphs along with the question ground on the scene feed our encoder model to infer a predicted answer. For a fair comparison, we train most of our model from scratch, except for the pre-trained scene graph generator $g$, whose weights are frozen.

### C.2. BERT Word Embedding and Question Encoder

In this case, we employ the BERT model as our word embedding approach and the question encoder, as being a more expressive language model.

Once again, the images are fed through a pre-trained scene graph generator $g$ from [7] work that generates scene graphs from images on the fly. In this particular case, all graph-level and node-level representations possess a dimen-

Table C.1: Detailed dimensions used in our study when employing the GloVE and BERT approaches.

| Methods | Word dim. | Question dim | Node Dim | Link Dim | Graph dim |
|---|---|---|---|---|---|
| GloVE+Transf | 300 | 300 | 300 | 300 | 512 |
| BERT | 756 | 512 | 512 | 512 | 512 |

sion size of 512, encompassing both node and edge features. This configuration is deliberately chosen to ensure that the dimensions of the representations closely align with the dimension yielded by BERT word embedding, which is 756. By maintaining consistency in the dimensionality across different components, we aim to facilitate seamless integration and compatibility with BERT-based models.

The word embedding for the BERT query encoder $f_q$ has its initial weights initialized by using embeddings from BERT [6]. Both hidden states and word embedding vectors have a dimension size of 512. The final question representation is derived by taking the average of all word embedding representations generated by BERT.

Following the same approach of [8, 9], we adopt a hierarchical sequence generation design, i.e., a transformer decoder module first parses the encoded question into a sequence of $M$ instruction vectors, $[i_1, i_2, \ldots, i_M]$. The $i$-th instruction vector will correspond exactly to the $i$-th execution step processed by the GNN encoder $f_g$ module. In our experiments, we force $M$ equals five. We note that SelfGraphVQA does not require any explicit supervision on how to solve the instruction step from the question, and we only supervise the final answer prediction.

In this scenario, we employ two self-supervised loss techniques: global similarity and regularization for permutation equivariance. Additionally, we incorporate the Distribution Link Representation Regularization method overall approaches performed in this case. It is important to note that the Distribution Link Representation Regularization is jointly executed with one of the self-supervised loss techniques.

As mentioned earlier, in this case, except for the object detector within the module, we have unfrozen the scene graph generator $g$ weights, allowing it to be trainable and to learn the representation and classification during the training process, merely according to the prediction answers. We have made deliberate experimental choices to mitigate potential biases and ensure an unbiased evaluation of our proposed framework.

For the un-normalized contrastive training step, we employ the MLP prediction head $h$. It comprises three fully connected layers, each followed by batch normalization and ReLU activation, except for the final layer. This setup ensures non-linearity and facilitates effective feature extraction. It is important to note that the MLP prediction head is exclusively utilized during the training phase and is subsequently discarded during inference, which aligns with prevailing practices in contemporary self-supervised training methods [2, 4].

The classification module $f_c$ is another integral component of our model. It is designed as a two-layer MLP with a dropout rate of $0.2$ and ELU activation.

Table D.1: Training details for the GloVE and BERT approaches employed in our study.

| Methods | Batch | Optimizer | lr | Epochs |
|---|---|---|---|---|
| GloVE+Transf | 64 | Adam | $10^{-4}$ | 50 |
| BERT | 32 | Adam Belief | $10^{-4}$ | 50 |

## D. Training Details

In this section, we provide further elaboration on our training approaches. Likewise, we have divided this section into two subsections: one with the utilization of GloVE word embedding along with a transformer-based model for the question encoder, and the other focusing on the application of BERT for word embedding and the question encoder.

### D.1. GloVe Word Embedding and Transformer-based Question Encoder

We train the models using the Adam optimizer with a learning rate of $10^{-4}$ and weight decay $10^{-4}$. We apply a batch size of 64, and a a linear learning rate schedule using a factor of $10^{-1}$ for every 20 epochs. All models are trained for 50 epochs. We emphasize that during training the weights of the scene graph generator $g$ are frozen, and do not receive weight updates.

### D.2. BERT for Word Embedding and Question Encoder

We train the models using the Belief Adam optimizer with a learning rate of $10^{-4}$ and weight decay $10^{-4}$. We apply a batch size of 32, and a linear learning rate schedule using a factor of $10^{-1}$ for every 10 epochs. All models are trained for 50 epochs. It is worth noting that in these cases, the weights of the scene graph generator $g$ are not frozen during training. This deliberate choice allows for continual updates and improvements, particularly in the edge representation, through the utilization of the Distribution Link Representation Regularization strategy.

## E. Self-Supervised implementation details

Table E.1 provides a comprehensive overview of the approach adopted in our study. It is worth noting that our training process was conducted sequentially and iteratively, allowing us to evaluate the performance of each approach before deciding on the subsequent implementation choice.

Table E.1: Detailed self-supervised implementation in our study by approaches.

| | SGG Methods | Baseline | Local Sim | Global Sim. | Self Sim |
|---|---|---|---|---|---|
| GloVE+Transf | Frozen SGG | ✓ | ✓ | ✓ | ✓ |
| | Link Regularizer | | | | |
| BERT | Frozen SGG | ✓ | | ✓ | |
| | Link Regularizer | | | ✓ | ✓ |

For instance, upon observing that the Local Similarity approach exhibited comparatively lower performance, albeit surpassing the baseline, we made the decision to discontinue its implementation on further research (i.e. with the BERT module and link distribution regularization approach). This strategy narrowed down the training possibilities, enabling us to focus solely on the most promising experiments. Another noteworthy example pertains to the utilization of BERT as our word embedding and query encoder module. Upon observing its positive impact on results, we exclusively applied the link distribution regularization technique with this architecture.

# F. Further Ablations

## F.1. Further Discussion on Language Bias

We elaborate on additional experiments aimed at evaluating the model's robustness when trained with the BERT module. In this case, the experiments investigate the impact of using a more expressive language model, such as BERT, on language biases in the VQA task and whether it harms the enhancement of visual information. We evaluate both how the biases convey not ideal information when using a more expressive language model such as BERT, and how the self-supervised approaches perform for robustness.

In this particular experiment, we augmented the images using various semantically-preserving techniques including Gaussian blur, Gaussian noise, color jitter with adjustments to brightness, contrast, and hue, as well as random rotation of up to 45 degrees. As for the questions, a similar approach was employed by randomly replacing up to 50% of the words with other words.ù

In this context, we emphasize that our approach maintains the semantic integrity of the image content. Consequently, the underlying model retains its fundamental objective of accurately predicting the correct response, despite the heightened complexity introduced.

Table F.1 demonstrated that even when employing a more expressive language model in the GQA dataset, the self-supervised learning still enhances the visual information for the predicted answer. Precisely, the results presented indicate that our approaches exhibit greater resilience to noise while maintaining the importance of visual information for the task.

We emphasize that the findings of this study demonstrate that despite the integration of a more expressive language model, such as BERT, the self-supervised learning method remains effective in leveraging visual data to classifier the predicted answers. Nevertheless, it is crucial to highlight that in this particular scenario, the results indicate a possible influence of language biases inherent in the dataset when utilizing a more advanced language model. See the results when the perturbation is employed solely on the scene graph compared to the non-perturbed one, in Table F.1. Additionally, when analyzing the results with full perturbation, the findings indicate an enhanced level of robustness when the self-supervision technique is combined with the model.

Table F.1: Sensitivity of accuracy (%) for bias analyzes of BERT module.

| **Setup** | **Methods** | | |
|---|---|---|---|
| Question + Scene Graph | BERT Baseline | BERTGlobal+link | BERTSelfSim+link |
| Noise + SG | 21.0 | 23.2 | 24.5 |
| Question + Noise | 42.4 | 41.8 | 42.8 |
| Noise + Noise | 19.8 | 21.7 | 21.3 |

## F.2. Does SelfGraphVQA have a few-shot learning capability?

We trained SelfGraphVQA with varying percentages of labeled data and found comparable performance to the GQA dataset, suggesting that adding self-supervised contrastive loss improves model generalization. We wanted to evaluate the different models on subsets of the full dataset. We tested reducing the ground truth labeling requirements and compared the performance when using SelfGraphVQA as opposed to directly training a fully supervised classification network.

In this case, we trained our SelfGraphVQA varying the percentage of labeled data, (i.e., 20%, 50%, and 100% of data) and evaluated it on the test dataset. As demonstrated in Fig. F.1, our proposal performs comparably with half of the GQA dataset evaluated on standard metrics. This insinuates that adding self-supervised un-normalized contrastive loss improves the generalization of the model.

Table F.2 shows how our proposal performs with the standard metrics when trained with 50% of training data, and we see that the three approaches perform on par with the baseline trained on the full dataset. In particular, the validity and plausibility metrics are consistent when compared to models trained on the full dataset.

Our intuition is that these metrics relate to linguistic bias and do not necessarily require large amounts of samples to converge, indicating that the model learns with little data what type of answer it should guess based on the type of question.

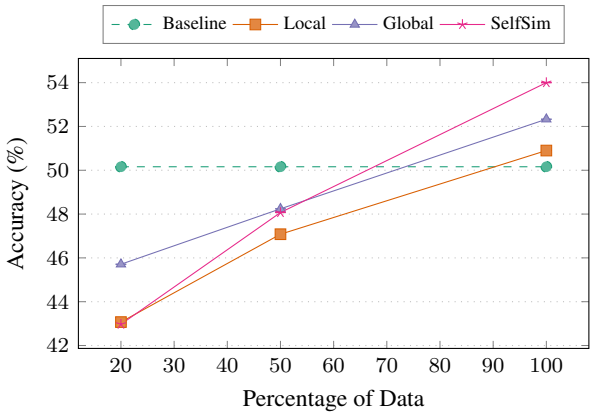


Figure F.1: Evaluation curve by percentage of data used in training on GQA dataset. The models obtain comparable results to baseline with 50% of the data. Note that we only illustrate the accuracy of the baseline trained on the full dataset for reference purposes.

Table F.2: Results (in %) evaluating by the standard metrics when training with 50% of GQA dataset.

| Method | Binary | Open | Consistency | Validity | Plausibility | Accuracy |
|---|---|---|---|---|---|---|
| Global | 63.5 | 27.6 | 54.1 | 94.8 | 90.1 | 48.2 |
| Local | 63.5 | 25.6 | 51.6 | 94.6 | 89.3 | 47.1 |
| SelfSim | 64.3 | 27.3 | 54.7 | 94.8 | 90.1 | 48.1 |

### F.3. More Examples

We present additional examples to illustrate how scene graphs can contribute to the explainability of AI in the context of VQA, Figure F.2. These examples highlight that VQA remains an open area of research and that the performance of a model should be evaluated beyond standard metrics. These examples serve as a reminder that there is room for further exploration and improvement in the field of VQA, extending beyond conventional evaluation metrics.

All examples were predicted by the SelfSim framework. In the following discussion, the additional examples demonstrate both the problem of low agreement of VQA question answers due to ambiguity and the usefulness of scene graphs in providing more explainable AI for this task.

For instance in example 1, the model accurately predicts the answer, and the detection of the airplane in the scene graph is easily visualized. Conversely, in example 2, the model correctly do not detect the object mentioned in the question, leading to a correct answer of 'No'.

The benefits of using scene graphs for visual question answering become more evident in examples 3 and 4. In example 3, the model provides an objectively correct answer despite a different ground truth answer in the dataset. This discrepancy is explained by the scene graph, which highlights that the extracted object related to the question is 'flowers' rather than 'flowers'. In example 4, the model correctly classifies the link that relates the chair located to the right of the curtains in the scene graph, enabling the model to predict the correct answer.

In example 5, the acceptance of the model's answer 'liquid' as opposed to the ground truth 'beverage' is subjective and depends on the evaluator's opinion. This demonstrates that the model's response may fail to precisely evaluate the question, emphasizing the inherent challenges in VQA.

Overall, these examples highlight the potential benefits of incorporating scene graphs in visual question answering, offering insights into the model's reasoning and contributing to more interpretable AI systems.

## References

[1] Aishwarya Agrawal, Ivana Kajic, Emanuele Bugliarello, Elnaz Davoodi, Anita Gergely, Phil Blunsom, and Aida Nematzadeh. Reassessing evaluation practices in visual question answering: A case study on out-of-distribution generalization. In *Conf. European Ch. Assoc. Comput. Ling. (EACL)*, pages 1171–1196, 2023. 1

[2] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 15750–15758, 2021. 2, 3

[3] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 6904–6913, 2017. 1

[4] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Adv. Neural Inf. Process. Sys. (NeurIPS)*, 33:21271–21284, 2020. 2, 3

[5] Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. *10.48550/arxiv.1902.09506*, 2019. 1, 2

[6] Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. *Adv. Neural Inf. Process. Sys. (NeurIPS)*, 31, 2018. 3

[7] Boris Knyazev, Harm de Vries, Cătălina Cangea, Graham W Taylor, Aaron Courville, and Eugene Belilovsky. Graph density-aware losses for novel compositions in scene graph generation. *arXiv preprint arXiv:2005.08230*, 2020. 1, 2

[8] Weixin Liang, Yanhao Jiang, and Zixuan Liu. GraghVQA: Language-guided graph neural networks for graph-based visual question answering. In *Wksp. Multimodal Artif. Intell. (ACLW)*, pages 79–86, Mexico City, Mexico, June 2021. Association for Computational Linguistics. 2, 3

[9] Weixin Liang, Feiyang Niu, Aishwarya Reganti, Govind Thattai, and Gokhan Tur. LRTA: a transparent neural-symbolic reasoning framework with modular supervision for visual question answering. *arXiv preprint arXiv:2011.10731*, 2020. 2, 3

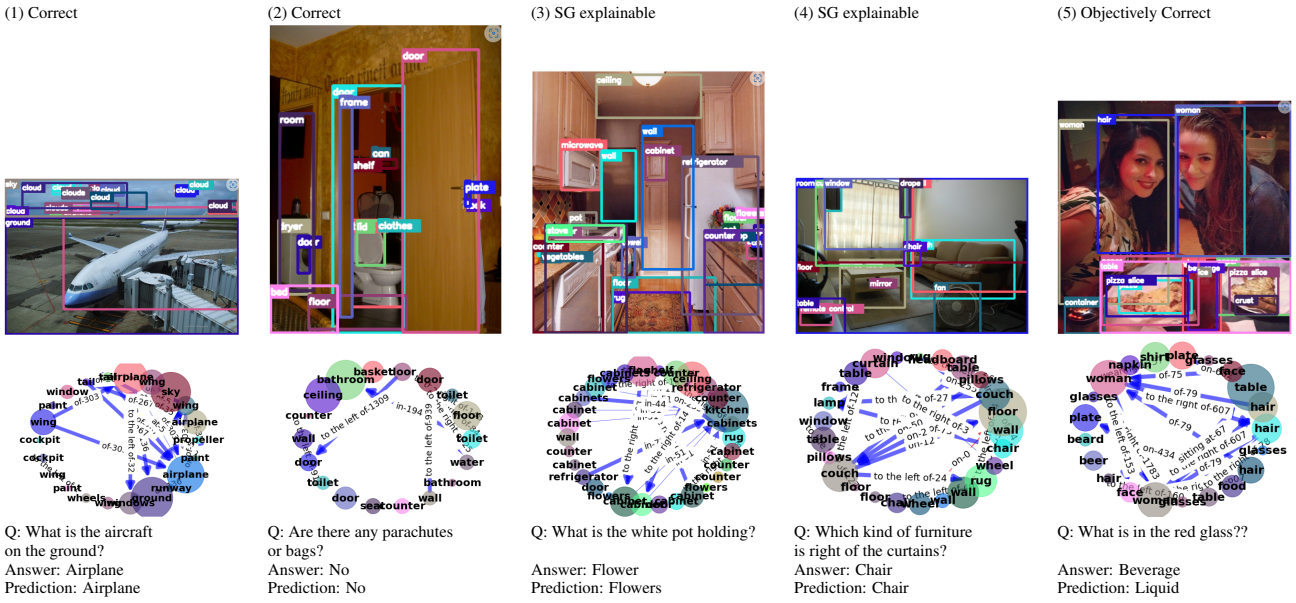


Figure F.2: Examples to demonstrate the complexity of VQA and the explainability of the scene graph. All example is predicted by the SelfSim framework.


[10] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Adv. Neural Inf. Process. Sys. (NeurIPS)*, volume 32, 2019. 1

[11] Binh X Nguyen, Tuong Do, Huy Tran, Erman Tjiputra, Quang D Tran, and Anh Nguyen. Coarse-to-fine reasoning for visual question answering. In *IEEE/CVF Inter. Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 4558–4566, 2022. 1

[12] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. In *Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, pages 1532–1543, 2014. 2

[13] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. 2