GAMER: Graphs As Multi-modal intErmediate Representations; An exploration of a high-resolution, graph-based multimodal approach

Daniel Volkov **Dekel Yehezkel** **Shahar Glam** **Neri Nigberg**

{danielvolkov, dekely1, shaharglam, nerinigberg}@mail.tau.ac.il

Abstract

Lorem ipsum dolor sit amet consectetur adipiscing elit. Quisque faucibus ex sapien vitae pellentesque sem placerat. In id cursus mi pretium tellus duis convallis. Tempus leo eu aenean sed diam urna tempor. Pulvinar vivamus fringilla lacus nec metus bibendum egestas. Iaculis massa nisl malesuada lacinia integer nunc posuere. Ut hendrerit semper vel class aptent taciti sociosqu. Ad litora torquent per conubia nostra inceptos himenaeos. Lorem ipsum dolor sit amet consectetur adipiscing elit. Quisque faucibus ex sapien vitae pellentesque sem placerat.

1 Introduction

TODO: I feel like clarity is lacking, as well as proper emphasis on the question we're talking. Revise

Representing multi-modality has been a matter of great interest within the past few years. Being able to integrate different modalities into a singular model enables us to capture complex relationships between them and solve practical tasks such as Visual Question Answering [1], Image-Captioning [8], Autonomous-Driving [24], Text-to-Audio generation [10] and the list goes on across many important fields.

The rise of transformers [21] has set a new State-of-the-Art (SotA) standard for generating high-quality embeddings by utilizing a cross-attention mechanism, these models can capture both intra-model relationships [4, 6, 9] and cross-modal relationships [17, 14, 18] with great success. Traditionally, modalities have been represented in simpler forms—text as token sequences, images as region features or patch embeddings, and audio as spectrogram frames before being merged. Fusion has typically been achieved either by concatenating these embeddings into a single stream [14], by linking separate encoders through cross-attention [15], or by mapping modalities into a shared latent space with contrastive objectives [17].

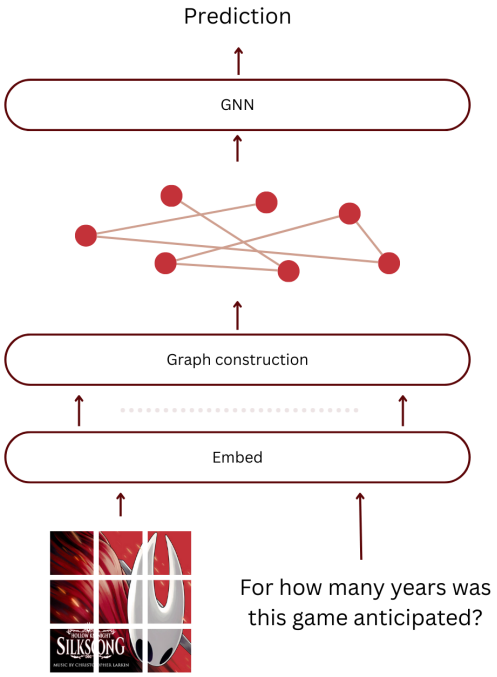


Figure 1: A high-level illustration of our methodology. Each modality is broken down into smaller atomic units (text → tokens, image → patches) which are then embedded. A graph is constructed with these embeddings as it's node features. A GNN is trained to produce predictions conditioned on these graphs.

However, recent papers have shown these approaches are not without fault. Recent evidence shows modern SotA models have difficulty with fine-grained tasks [27], provide sub-human results in visual-spatial tasks [26] and even struggle with basic reasoning [20]. This opens up questions about better ways to represent these multi-modal interactions.

In this paper, we explore alternative approaches on representing multi-modality. We hypothesize the aforementioned techniques often fail to capture the *relationship* between modalities, and examine the potential of capturing these relationships using graphs, utilizing GNN's as our neural network architecture.

Probably one more paragraph providing the high-level details (MMG for tests, Cayley for baseline, GPS, pooling probably doesn't matter too much but you'll decide Daniel)

2 Prior work

Xu et al. [25] surveys the popular approaches taken with handling a multi-modal input and the different methods of applying self-attention to generate a token which expresses the interaction between the modalities. Ultimately, all methods boil down to applying a sequence of summation/concatenation/transformer operations on the uni-modal embeddings. This means the way uni-modal tokens are processed in a modality-agnostic way, such that all modalities have the same structure and are handled identically.

There is research about multi-modal representation in GNN's, however it is heavily centered around specific, often niche use-cases, such as emotion-recognition in conversations [12], recommender systems [16] and other such cases where the topology of the graph is either known, or can be rather trivially constructed.

For cases where the graph isn't so easily deduced, strides have been made in the realm of topology-prediction [7, 3] - where the structure of the graph is learned via a separate down-stream task, reminiscent of meta-learning.

However, despite success within these two realms, to the best of our knowledge, there's been no research attempting to maintain the shape of the original embeddings for a general, complex, multi-modal use-case.

3 Method

To achieve high-resolution multi-modal modeling, we choose to break down each modality into smaller atomic units – tokens for the text modality, and patches for the image modality.

As illustrated in figure 1, our approach consists of 3 main components: 1) The embedding used for the atomic units. 2) The graph construction method, used to construct graphs from the resulting embeddings. 3) The GNN used for prediction.

While our approach could be generalized to a wide array of multi-modal tasks, we provide a proof-of-concept, and focus our experimentation on Visual Question Answering (VQA). The VQA task, as introduced in [1], consists of ⟨Question,Image⟩ pairs, where the questions are about the images. The model is tasked to predict the answer to the question, conditioned on both image and question.

3.1 Embeddings

Since our approach leverages the token and patch embeddings as the initial features for our graphs, we aim to get meaningful token/patch-level embeddings. To test various aspects of our approach, we experiment with different types of embeddings:

Unaligned embeddings. Uni-modal encoders are used to embed each modality *'in a vacuum'*. By using unaligned embeddings as the initial node features of our input graphs, we essentially force our GNN to learn multi-modal fusion as it optimizes for the task at hand. For the text modality we choose the celebrated BERT encoder [5], providing token embeddings that are contextualized within their sentence. To achieve a similar contextualization effect for our image patches, we use the BEiT encoder [2]. This encoder is trained using Masked-Image-Modeling, analogously to the way BERT is trained, and achieves contextualized patch embeddings.

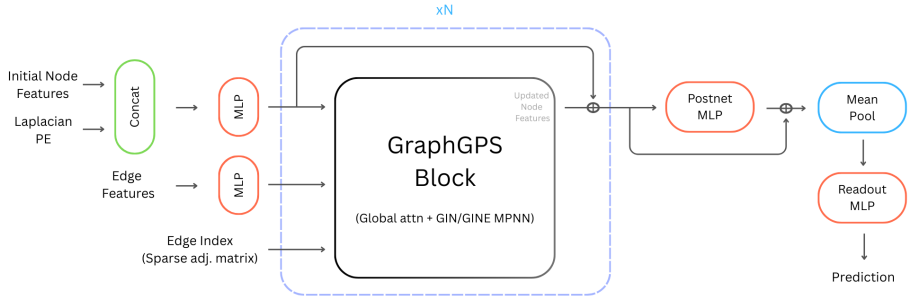
Aligned embeddings A multi-modal encoder is used to embed all modalities together, into a shared latent space, where the resulting embeddings are aligned across modalities. Using aligned embeddings, the model already has access to connections between the two modalities, and needs to learn a simpler mapping from initial features to predictions. We experiment with two multi-modal encoders: the first is the celebrated CLIP encoder [17], which is trained on contrastive pairs, and provides high quality aligned embeddings that were trained to capture global semantic features. The second is the BLIP encoder [13], which is fine-tuned on VQA, and provides aligned embeddings that are tuned specifically for question answering, while capturing fine-grained as well as global semantic features.

3.2 Graph Construction

To represent and introduce connections between the different modalities, we leverage the graph as an intermediate representation of both modalities as a single entity. The constructed graph carries the embeddings of the aforementioned *atomic units* as it's node features. We experiment with two methods of graph construction, each emphasizing different aspects of our desired properties for the graph structure.

Cayley Graphs. The authors of [23] suggest incorporating Cayley graphs for message passing, to prevent over-squashing and bottlenecks in-

Figure 2: An Illustration of our GNN architecture. PE stands for Positional Encoding. Laplacian PE and edge features are optional and may not be passed. The $\oplus$ symbol represents a residual connection. The mean pool receives the final node embeddings $X \in \mathbb{R}^{|V| \times d_{\text{hidden}}}$ and computes their mean over the first dimension, resulting in a single d_{hidden}-dimensional vector, which is then used for graph-level prediction. Each GraphGPS block uses either GINE – when edge features are present, or GIN – when they are not.



duced by general input graphs – leveraging the mathematical properties of Cayley graphs as expander graphs. We adopt a similar approach, and experiment with a Cayley graph structure as a baseline for our approach. Given token embeddings $T \in \mathbb{R}^{N_{\text{tokens}} \times d_{\text{embed}}}$ and patch embeddings $P \in \mathbb{R}^{N_{\text{patches}} \times d_{\text{embed}}}$, we stack them across the first dimension, resulting in the stacked embedding matrix $\texttt{Embeds} \in \mathbb{R}^{(N_{\text{tokens}}+N_{\text{patches}}) \times d_{\text{embed}}}$. We then proceed to construct a Cayley graph with the minimal number of nodes greater than (or equal to) $N_{\text{tokens}} + N_{\text{patches}}$. We extend our Embeds matrix with zero-vector embeddings to match the number of nodes in the constructed Cayley graph (i.e. the additional *virtual nodes'* features are initialized to zero-vectors). We adopt the publicly available code[1] by the authors of [23] for Cayley graph construction.

MMGs. We design a graph structure, specifically tailored to represent multi-modal data and introduce connections intra and inter-modally. **TODO: THIS** (model each modality naively, then hierarchy of virtual nodes and multiple pathways for propogation)

3.3 GNN Architecture

The nodes (alongside their features) of our graph, being tokens and image patches, are connected in ways that require attention to both global and local phenomena, both intra and inter-modally. To achieve these properties, as-well as considering the graph as a strong source of relevant inductive biases, we choose to implement a model using the GraphGPS architecture [19], which utilizes global attention alongside MPNNs. Our base architecture is detailed in figure 2.

Graph Pooling. While the base GraphGPS template offers impressive expressive power (as illustrated in **ADD EXAMPLES**), using it with a mean-pool for graph classification essentially forces all final node representations to participate in the graph-level readout. Following the *'fusion nodes'* motivation from our MMGs, we experiment with Self-Attention Graph Pooling (SAGPool) [11]. The SAGPool layer uses a MPNN as a node-level classifier, computing a score for each node, and filtering out a fraction of the total nodes with the lowest scores. For the MPNN used by the SAGPool layers we use the celebrated GAT architecture [22]. Inspired by [11], we experiment with two ways to incorporate SAGPool layers into our base architecture: 1) Global SAGPool – A single SAGPool layer right after the postnet MLP and before the mean pool. This method acts as a *'smarter pooling'*, allowing the model to choose which nodes will be used in classification for the graph-level task, and allowing message passing and global attention to *'concentrate'* on specific nodes for classification. 2) Hierarchical SAGPool – Analogous to the way max-pooling is used in CNNs. Multiple SAGPool layers, each after the GraphGPS block on a chosen subset of layers. This method reduces the graph iteratively, with MPNN and global attention steps in-between, allowing the model to essentially *'throw out'* un-important nodes early on in the forward-pass.

4 Experimental Details

TODO

Setup. VQA Classification setup. Evaluation setup. Dataset by pingzhili. Explain the test splits.

[1] https://github.com/josephjwilson/cayley_graph_propagation

Experiment No.	Graph Structure		SAGPool Method			VQA Subset	
	MMG	Cayley	N	G	H	test-dev	validation
1	✓	-	✓	-	-	$44.43_{\pm 0.02}$	$44.015_{\pm 0.025}$
2	✓	-	-	✓	-	$\mathbf{45.305}_{\pm 0.015}$	$\mathbf{44.825}_{\pm 0.035}$
3	✓	-	-	-	✓	$43.85_{\pm 0.01}$	$43.255_{\pm 0.015}$
4	-	✓	✓	-	-	$\mathbf{45.29}_{\pm 0.01}$	$\mathbf{44.605}_{\pm 0.005}$
5	-	✓	-	✓	-	$44.205_{\pm 0.035}$	$43.475_{\pm 0.005}$
6	-	✓	-	-	✓	$44.365_{\pm 0.005}$	$43.655_{\pm 0.035}$

Table 1: The results for BERT/BEiT embeddings. For the test-dev split we report the results obtained by submission to the official VQA challenge, for the validation split we manually calculate the VQA evaluation metric and average over the validation split. In the SAGPool method column, N stands for None, G is Global SAGPool, and H is Hierarchical SAGPool (As mentioned in section 3.3). The reported result is the mean of the best (w.r.t the VQA accuracy metric) and last checkpoints of the relevant experiment, where $\pm$ denotes the standard deviation. The best result for each graph structure is written with a **bold font**, the best result per split (across both graph structures) is highlighted with .

Experiment No.	Method Description	Embedding type			VQA Subset	
		CLIP	BLIP	BERT/BEiT	test-dev	test-std
1	Our MMG, Global SAGPool	✓	-	-	$38.65_{\pm 0.36}$	$yy.yy$
2		-	✓	-	$44.375_{\pm 0.005}$	$yy.yy$
3 (Baseline)		-	-	✓	$\mathbf{45.305}_{\pm 0.015}$	$yy.yy$
3	Cayley Graph, No SAGPool	✓	-	-	$41.51_{\pm 0.01}$	$y.yy$
4		-	✓	-	$43.355_{\pm 0.005}$	$yy.yy$
5 (Baseline)		-	-	✓	$45.29_{\pm 0.01}$	$y.yy$

Table 2: The results for multi-modal aligned embeddings. The reported results are generated and denoted as in table 1. Experiments No. 3 and 5 are the experiments from table 1, mentioned as a baseline for this experiment. For test-std we only report the result of the best performing checkpoint on test-dev, since submissions to test-std are limited. Best results (per split) are highlighted.

Training details. Model config. Hyperparams. Optimizer-scheduler combo. Mention our code+wandb+checkpoints on hf. Don't forget we did a couple of lrs and lowered until training was stable and didn't diverge.

Experimental design. We first experiment with shitty generic unaligned embeddings (BERT/BEiT) to evaluate the contributions of the graph structure and GNN model to performance on VQA. This also tests the ability of our approach to learn multimodal fusion alltogether. These first experiments are the baseline ($V1.0$). Then we atke the best performing model for each graph structure and try to push it to SOTA by swapping the generic embeddings to something that is aligned multimodally.

The VQA test-std split is the official leaderboard challenge, and as such submission to it is limited. We thus only report the test-std scores for the aligned embeddings, as they are our shot as SOTA. For both sets of experiments we report test-dev which is also an official test split, without a submission limit.

5 Results & discussion

TODO

6 Future work

TODO

Limitations

Shit about compute and resources innit. I believe this is typically written in key-words but I'll have to double check.

Ethics Statement

All practices were in-line with Tel Aviv University's ethics guidelines and all data used is free for public use.

References

[1] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. 2016. Vqa: Visual question answering. *Preprint*, arXiv:1505.00468.

[2] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. 2022. Beit: Bert pre-training of image transformers. *Preprint*, arXiv:2106.08254.

[3] Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Preprint*, arXiv:2006.13009.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. *Preprint*, arXiv:2010.11929.

[7] Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. 2020. Learning discrete structures for graph neural networks. *Preprint*, arXiv:1903.11960.

[8] Taraneh Ghandi, Hamidreza Pourreza, and Hamidreza Mahyar. 2023. Deep learning approaches on image captioning: A review. *ACM Computing Surveys*, 56(3):1–39.

[9] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *Preprint*, arXiv:2106.07447.

[10] Rongjie Huang, Jiawei Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiang Yin, and Zhou Zhao. 2023. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. *Preprint*, arXiv:2301.12661.

[11] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-attention graph pooling. *Preprint*, arXiv:1904.08082.

[12] Jiang Li, Xiaoping Wang, Guoqing Lv, and Zhigang Zeng. 2023. Graphmft: A graph network based multimodal fusion technique for emotion recognition in conversation. *Neurocomputing*, 550:126427.

[13] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. *Preprint*, arXiv:2201.12086.

[14] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *Preprint*, arXiv:1908.03557.

[15] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Preprint*, arXiv:1908.02265.

[16] Feng Mo, Lin Xiao, Qiya Song, Xieping Gao, and Eryao Liang. 2024. Multimodal graph neural network for recommendation with dynamic de-redundancy and modality-guided feature de-noisy. *Preprint*, arXiv:2411.01561.

[17] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *Preprint*, arXiv:2103.00020.

[18] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision. *Preprint*, arXiv:2212.04356.

[19] Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. 2023. Recipe for a general, powerful, scalable graph transformer. *Preprint*, arXiv:2205.12454.

[20] Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. 2022. Winoground: Probing vision and language models for visio-linguistic compositionality. *Preprint*, arXiv:2204.03162.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

[22] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. *Preprint*, arXiv:1710.10903.

[23] JJ Wilson, Maya Bechler-Speicher, and Petar Veličković. 2025. Cayley graph propagation. *Preprint*, arXiv:2410.03424.

[24] Yi Xiao, Felipe Codevilla, Akhil Gurram, Onay Urfalioglu, and Antonio M. Lopez. 2022. Multimodal end-to-end autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):537–547.

[25] Peng Xu, Xiatian Zhu, and David A. Clifton. 2023. Multimodal learning with transformers: A survey. *Preprint*, arXiv:2206.06488.

[26] Jihan Yang, Shusheng Yang, Anjali W. Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. 2025. Thinking in space: How multimodal large language models see, remember, and recall spaces. *Preprint*, arXiv:2412.14171.

[27] Yuhui Zhang, Alyssa Unell, Xiaohan Wang, Dhruba Ghosh, Yuchang Su, Ludwig Schmidt, and Serena Yeung-Levy. 2024. Why are visually-grounded language models bad at image classification? *Preprint*, arXiv:2405.18415.

A Illustrations and plots

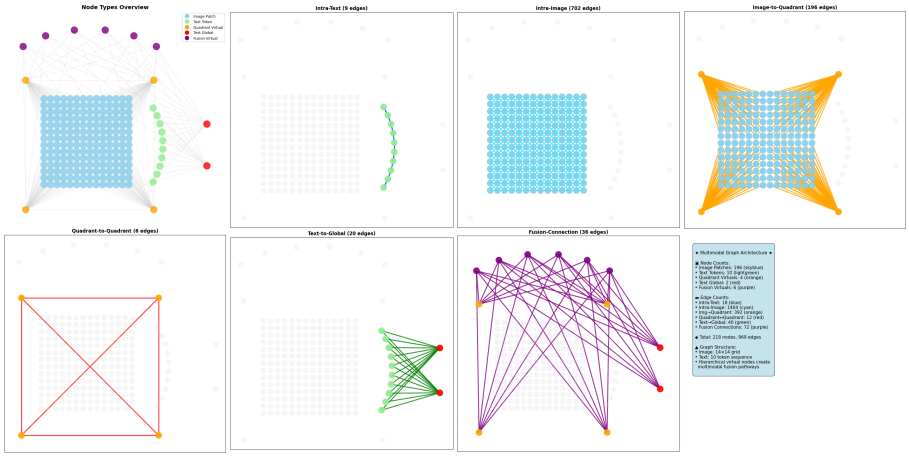


Figure 3: An illustration of our proposed MMG structure. The top-left sub-plot showcases the different node types present in our MMGs. All other sub-plots showcase specific nodes and the connections between them: 1) Text token nodes and intra-text edges. 2) Image patch nodes and intra-image edges. 3) Image patch to quadrant conections. 4) Image Quadrant to Quadrant connections. 5) Text token to text global connections. 6) Image quadrants and text global nodes to fusion nodes connections.