Multimodal Graph Neural Architecture Search Under Distribution Shifts

Jie Cai [1], Xin Wang [2,3*], Haoyang Li [2], Ziwei Zhang [2], Wenwu Zhu[2,3*]

[1]Tsinghua-Berkeley Shenzhen Institute, Tsinghua University
[2]Department of Computer Science and Technology, Tsinghua University
[3]Beijing National Research Center for Information Science and Technology, Tsinghua University
caij20@tsinghua.org.cn, {xin_wang, zwzhang, wwzhu}@tsinghua.edu.cn, lihy18@mails.tsinghua.edu.cn

Abstract

Multimodal graph neural architecture search (MGNAS) has shown great success for automatically designing the optimal multimodal graph neural network (MGNN) architecture by leveraging multimodal representation, crossmodal information and graph structure in one unified framework. However, existing MGNAS fails to handle distribution shifts that naturally exist in multimodal graph data, since the searched architectures inevitably capture spurious statistical correlations under distribution shifts. To solve this problem, we propose a novel Out-of-distribution Generalized Multimodal Graph Neural Architecture Search (OMG-NAS) method which optimizes the MGNN architecture with respect to its performance on decorrelated OOD data. Specifically, we propose a multimodal graph representation decorrelation strategy, which encourages the searched MGNN model to output representations that eliminate spurious correlations through iteratively optimizing the feature weights and controller. In addition, we propose a global sample weight estimator that facilitates the sharing of optimal sample weights learned from existing architectures. This design promotes the effective estimation of the sample weights for candidate MGNN architectures to generate decorrelated multimodal graph representations, concentrating more on the truly predictive relations between invariant features and ground-truth labels. Extensive experiments on real-world multimodal graph datasets demonstrate the superiority of our proposed method over SOTA baselines.

1 Introduction

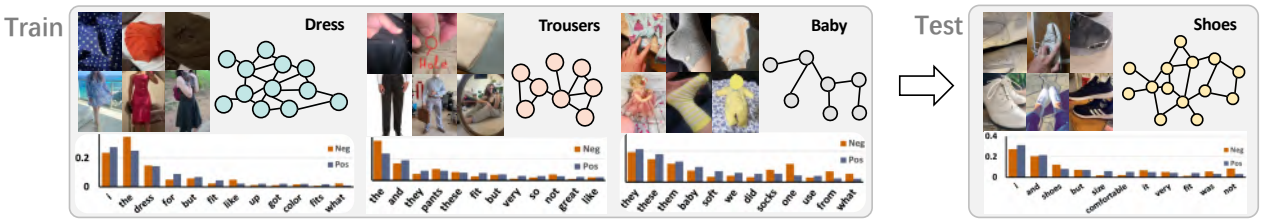
Multimodal graph data is ubiquitous in various real-world applications such as social media (Li et al. 2019b; Tao et al. 2020; Li et al. 2021a), biomedicine (Wen et al. 2022), and health (Gao et al. 2021a). Accordingly, the development of a capable multimodal graph neural architecture search (MG-NAS) algorithm is of significant importance to effectively process this complex data type for various tasks and data distributions. MG-NAS aims at automatically designing multimodal graph neural networks (MGNN) and getting more powerful models, which achieves great success in various multimodal graph tasks (Cai et al. 2022) under the identically distributed (ID) assumption, where the training and testing multimodal graphs are sampled from the same distribution. Different MGNN architectures have been proven to have different performances on different tasks and data distributions because of the diverse message-passing mechanisms (Li et al. 2022b) and crossmodal interaction modes.

However, distribution shifts in multimodal graph data - which indicate changes in the statistical properties of data across domains or over time, are very common in real-world applications. Distribution shifts can arise from various factors, such as modifications in real-world conditions, variations in data collection processes, or the presence of confounding variables. Especially in the case of multimodal graph data, where each graph or node is characterized by multiple modalities, may be impacted by distinct factors. Furthermore, these factors always interact with each other in intricate and complex ways. Furthermore, distribution shifts in multimodal graph data occur not only in single-modal or multimodal interactions within a node but also in the overall distribution of the graph. As shown in Figure 1, the distribution of multimodal reviews on Amazon varies across different commodity categories, such as dress, trousers and shoes.

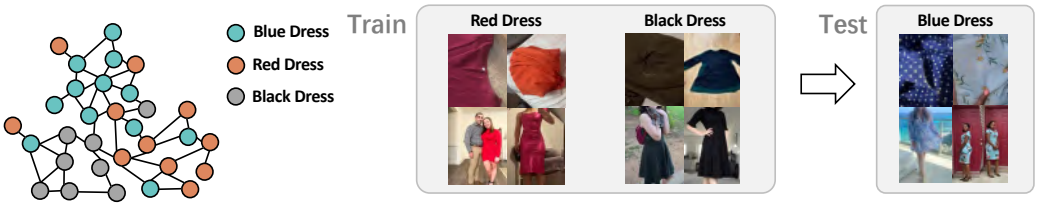
Existing MGNAS or NAS approaches are based on the ID assumption. When there is a distribution shift, each MGNN is trained on the training dataset and evaluated on the validation dataset. The best-performing model on the validation dataset is then selected for the new data distribution. However, it is important to note that the selected MGNN is more prone to overfitting the training distribution, resulting in the over-exploitation of spurious features and disregard of invariant features. Consequently, the best MGNN architecture derived by MGNAS may be sub-optimal in out-of-distribution (OOD) scenarios due to the mistakenly learned features and the inadequate evaluation strategy.

The problem of distribution shifts can be particularly challenging, and developing effective methods to handle distribution shifts in multimodal graph data is an essential research topic. The first challenge in addressing the OOD generalization problem of MGNAS is how to model the distribution shifts across graph structures, modalities, and their complex interactions. Given the complex nature of feature interactions in multimodal graphs, distinguishing between invariant and variant features can pose significant challenges. The second challenge lies in automating the search for the best model in multimodal graph OOD situations. It will be ben-

*Corresponding authors



(a) Distribution Shift on Multimodal Domain



(b) Distribution Shift on Singlemodal Domain (Color Shift)

Figure 1: Distribution shifts in the Amazon dataset. Different node colors represent different node distributions and variations in graph density correspond to different graph structure distributions. In Figure (a), the MGNNs are trained on Dress, Trousers and Baby but tested on Shoes, leading to distribution shifts in both multimodal node features and graph structures. In Figure (b), the MGNNs are trained with red and black dresses but tested with blue dresses with a single-modal distribution shift.

eficial if we leverage the diverse characteristics of searched models to enhance the OOD generalization capabilities. It is also important to minimize the inconsistent performance between the validation and the training dataset.

As an effort towards enhancing the generalization capability of MG-NAS, we propose a method named Out-of-distribution Generalized Multimodal Graph Neural Architecture Search (OMG-NAS) that searches for architectures with both maximal predictive performance and maximal generalization ability. The overview of the proposed OMG-NAS is illustrated in Figure 2. Firstly, to address the different distribution shift patterns of different modalities and their complex interactions at their core, we disentangle the multimodal features into singalmodal contributions. Notably, we observe that without this disentanglement, the performances of MGNNs are significantly compromised. Secondly, we employ sample reweighting and random Fourier features (RFF) to decorrelate multimodal graph features and alleviate the impact of intricate non-linear dependencies during the learning process. The global multimodal sample weight estimator (GMSWE), the sampled MGNN model, and the controller are optimized iteratively in an end-to-end manner. Finally, to fully leverage the diverse models exploited by OMG-NAS, we train and maintain the global weights across different architectures. The global weights are model agnostic and allow for efficient warm-starting of new architectures, leading to a more stable searching process.

Our contributions are summarized as follows:

- To the best of our knowledge, we are the first to formulate the problem of OOD graph neural architecture search from multiple modalities. We introduce three novel multimodal graph-OOD datasets to evaluate the generalization ability of the proposed method.
- We propose an OMG-NAS method that automatically searches for the best MGNN model with the best OOD generalization ability by using a novel GMSWE module to optimize the global weights that decorrelate multi-modal invariant and variant features.
- We construct extensive experiments that demonstrate the superiority of OMG-NAS over previous SOTA methods in both graph classification and node classification tasks..

2 Related Works

Multimodal Graph Learning. Given the success of graph learning in information aggregation and transmission (Li et al. 2019b), some scholars focus on multimodal graph learning to effectively utilize the dependencies and relationships across multiple modalities in information dissemination. Multimodal graph neural network (MGNN) aims to represent multimodal graph-structured data in an end-to-end manner (Peng et al. 2017; Wu et al. 2020), taking into account both multimodal information aggregation and message passing. MGNNs also provide an expressive and flexible strategy to leverage interdependencies in multimodal datasets (Gao et al. 2020a). Although these multimodal graph learning methods have made great success, it is worth noting that they are designed for ID conditions. That is, they can not be directly applicable in OOD scenarios due to the lack of generalization ability.

Out-of-Distribution Generalization. In real-world scenarios, the distribution of training graphs may differ from that of testing graphs, leading to unstable inference across different testing environments (Zhu et al. 2021; Ding et al. 2021). To tackle the OOD problem on graphs, researchers have proposed various methods including disentanglement-based graph models (Fan et al. 2022; Li et al. 2022c, 2021b), causality-based graph models (Li et al. 2022a; Chen et al. 2022) and graph invariant learning (Li et al. 2023, 2022d; Wu et al. 2022a). Sample reweighting is an effective tool for addressing distribution shift problems. (Shen et al. 2020) introduce an online reweighting method that utilizes a set of unbiased clean validation examples for sample reweighting. Similarly, (Fang et al. 2020) propose to automatically learn

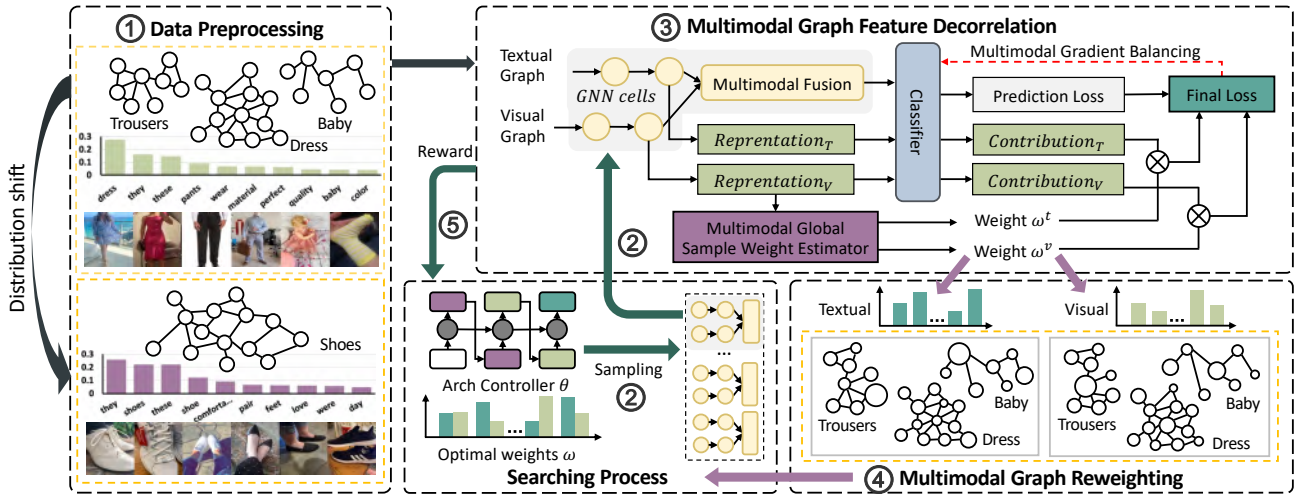


Figure 2: Overview of the OMG-NAS method. After ① data preprocessing, we ② sample an MGNN architecture using the controller, then we ③ optimize the MGNN model with Multimodal Graph Feature Decorrelation (MGFD) and ④ get the optimal global multimodal sample weights using Global Multimodal Sample Weight Estimator across Architectures (GMSWE). The performance of the optimized MGNN acts as the ⑤ reward to guide the training of the controller in the next optimization cycle.

an explicit loss-weight function that is parameterized by an MLP.

Graph Neural Architecture Search. Graph neural architecture search (GraphNAS) intends to automatically search for the most effective model architecture without human intervention (Qin et al. 2022b; Zhang et al. 2022b, 2023). Existing GraphNAS methods can be categorized into reinforcement learning-based methods (Gao et al. 2021b; Zhou et al. 2022), evolutionary algorithms (Nunes and Pappa 2020; Shi et al. 2022), and differentiable methods (Zhao et al. 2020a,b). In recent years, scholars also explore how NAS performs under distribution shifts (Bai et al. 2021; Qin et al. 2022a). However, there are still significant challenges when dealing with multimodal graph data. An aspect worthy of attention is that NAS methods naturally generate a diverse set of models that have proven effective for OOD settings (Pagliardini et al. 2022; Teney et al. 2022; Rame et al. 2022).

3 Methodology

3.1 Preliminaries

Notations Consider a multimodal graph represented as $G = (\mathcal{U}, \mathcal{E})$, where $\mathcal{U} = \{u_1, \cdots, u_N\}$ is the vertex set with size N and $\mathcal{E} = \{\langle u_i, u_j \rangle | 1 \leq i, j \leq N\}$ is the edge set with $|\mathcal{E}|$ edges. Each node $u_i \in \mathcal{U}$, u_i corresponds to multimodal node feature $X_i = [X_i^t, X_i^v]$ including the textual feature X_i^t and the visual feature X_i^v. For textual modality, each node can represent either words, sentences, or paragraphs, while for visual modality, each node can represent either a part of a picture or a whole picture. For the node classification task, the dataset contains graphs with N nodes $\mathcal{U} = \{(u_i, y_i) | i = 1, \cdots, N\}$ where y_i is the label of node u_i. For the graph classification task, the dataset contains a set of graphs $D = \{(G_j, y_j) | j = 1, \cdots, M\}$ where y_j is the label of graph G_j. This paper focuses on the node classification task and graph classification task. However, our method can also be easily extended to other multi-modal graph learning tasks. MG-NAS aims to find the best architecture $A^* \in \mathcal{A}$ that maximizes the prediction accuracy given the pre-defined search space $\mathcal{A}$.

OOD problem of MG-NAS Given a training multimodal graph G from the distribution $P_{tr}(G, Y)$, MG-NAS needs to handle the testing multimodal graph G from a new distribution $P_{te}(G, Y)$, where $P_{tr}(G, Y) \neq P_{te}(G, Y)$ and $P_{te}(G, Y)$ is unknown during the training process. In this scenario, MG-NAS faces the issue of over-fitting the training data, leading to sub-optimal MGNN architectures.

The goal of this paper is to address the issue of overfitting in the training data and leverage the diverse models generated by MG-NAS to develop a re-weighting approach that effectively decorrelates the multimodal graph information acquired from the training set. Given an input multimodal graph G for the node classification task (or a set of multimodal graphs $\{G\}$ for the graph classification task), we aim to optimize the following objective:

$$(A^*, W^*, \omega^*) = \underset{(A,W,\omega) \in (\mathcal{A}, \mathcal{W}, \Omega)}{\arg\min} \mathcal{L}_{train}(f_{cls}(\Phi_{A,W}(G; \omega))), \quad (1)$$

where A^*, W^* are the best MGNN architecture and optimal trainable model parameters, respectively. ω^* is the best sample weight for multimodal graph feature decorrelation, $\Phi_{A,W}(G; \omega)$ denotes the MGNN encoder under weights ω, f_{cls} represents the classifier, $\mathcal{L}_{train}$ represents the loss function. We use two GNN encoders $\phi^t(\theta^t, \cdot)$ and $\phi^v(\theta^v, \cdot)$ to extract textual and visual features. We denote $Z_n^t = \phi^t(\theta^t, u_n^t) = [Z_{n1}^t, \cdots, Z_{nm_t}^t] \in \mathbb{R}^{N \times m_t}$, $Z_n^v = \phi^v(\theta^v, u_n^v) = [Z_{n1}^v, \cdots, Z_{nm_v}^v] \in \mathbb{R}^{N \times m_v}$ as the uni-modal representations of node u_n.

3.2 Multimodal Graph Feature Decorrelation

We aim to identify the optimal weights of training samples to eliminate the dependency between features and remove the reliance on spurious features in the multimodal graph representation space. However, simply weighting each

sample will fail to achieve OOD generalization in the multimodal graph OOD settings. To tackle this problem, we propose a multimodal graph feature decorrelation (MGFD) method. MGFD separates the features into unimodal components and multimodal interactions, then utilizes the approximation prediction generated by the unimodal sub-networks to decorrelate features within each modality.

We first introduce how to obtain textual intra-modal weights ω^t and the process is similar for visual intra-modal weights ω^v. We eliminate dependence between sub-features in the single-modality representation space by measuring their relevance based on the sample data. We adopt the squared Frobenius norm of the partial cross-covariance matrix $\|\hat{\Sigma}_{Z^t_{*i},Z^t_{*j}}\|^2_F$ as a way of quantifying the degree of independence, inspired by (Zhang et al. 2021; Li et al. 2022a):

$$\hat{\Sigma}_{Z^t_{*i},Z^t_{*j}} = \frac{1}{N-1}\sum_{n=1}^{N}[(h(Z^t_{ni}) - \overline{h}(Z^t_{*i}))^\top \cdot (g(Z^t_{nj}) - \overline{g}(Z^t_{*j}))], \quad (2)$$

where Z^t_{ni} and Z^t_{nj} denote the value of textual random variables Z^t_{*i} and Z^t_{*j} given the input node u_n, N is the number of training samples. $h(\cdot)$ and $g(\cdot)$ are random Fourier features (RFF) mapping functions and we select Q functions from the RFF function space $\mathcal{H}_{RFF}$. $\overline{h}(Z^t_{*i})$ and $\overline{g}(Z^t_{*j})$ are the mean values of vectors $h(Z^t_{*i}) = [h(Z^t_{1i}), \cdots, h(Z^t_{Ni})]^\top$ and $g(Z^t_{*j}) = [g(Z^t_{1j}), \cdots, g(Z^t_{Nj})]^\top$.

Given textual intra-modal weights, the re-weighted partial cross-covariance matrix can be calculated as

$$\hat{\Sigma}^{\omega^t}_{Z^t_{*i},Z^t_{*j}} = \frac{1}{N-1}\sum_{n=1}^{N}[(\omega^t_n h(Z^t_{ni}) - \overline{h}_{\omega^t}(Z^t_{*i}))^\top \cdot (\omega^t_n g(Z^t_{nj}) - \overline{g}_{\omega^t}(Z^t_{*j}))], \quad (3)$$

where $\overline{h}_{\omega^t}(Z^t_{*i})$ and $\overline{g}_{\omega^t}(Z^t_{*j})$ are weighted average of vectors $h(Z^t_{*i})$ and $g(Z^t_{*j})$ with weights $\omega^t = [\omega^t_1, \cdots, \omega^t_N]^\top$.

To eliminate the dependence between representations, we optimize ω^t by minimizing the squared Frobenius norm of the partial cross-covariance matrix:

$$\omega^{t*} = \arg\min_{\omega^t \in \Delta} \sum_{1 \le i < j \le m^t_Z} \|\hat{\Sigma}^{\omega^t}_{Z^t_{*i},Z^t_{*j}}\|^2_F, \quad (4)$$

where $\Delta = \{\omega^t \in \mathbb{R}^N_+ | \sum_{n=1}^N \omega^t_n = N\}$, m^t_Z is the dimension of Z^t.

Afterwards, we iteratively optimize the multimodal feature weights $\omega = [\omega^t, \omega^v]$, MGNN encoder $\Phi = \{\phi^t, \phi^v, \phi^f\}$ and classifier f_{cls} by minimizing the following MGFD loss function $\mathcal{L}_w$:

$$\mathcal{L}_w = \sum_{n=1}^{N} \omega^t_n L(f_{cls}(\phi^t(x^t_n), y_n)) + \omega^v_n L(f_{cls}(\phi^v(x^v_n), y_n)), \quad (5)$$

where L denotes the cross-entropy loss, ω^t, ω^v are the textual and visual intra-modal weights with length of N, indicating the importance of unimodal training features. The overall loss function consists of two terms: the first term represents the cross-entropy loss function and the second term represents the MGFD loss:

$$\mathcal{L}_{train} = -\frac{1}{N}\sum_{n=0}^{N}(y_n \log \hat{y}_n - (1-y_n)\log(1-\hat{y}_n)) + \mathcal{L}_w. \quad (6)$$

When one of the modalities is more susceptible to be influenced by spurious features, MGNN tends to learn the spurious feature of this modality over the other modality. To fully learn from the two GNN encoders of MGNN, we also incorporate the OGM-GE method (Peng et al. 2022) as a plug-in module to prevent over-reliance on the spurious features of a single modality.

3.3 Global Multimodal Sample Weight Estimator across Architectures

In Equation 4, our objective is to learn unique weights for each sample's unimodal contribution. However, different MGNN models offer diverse multimodal graph feature spaces for learning these feature weights. To address this issue, we propose a novel approach called the global multimodal sample weight estimator (GMSWE). During the training of the OMG-NAS controller, we employ a saving-reloading-finetuning method.

We have observed that the learned weights have limited dependence on the sampled model architectures in the architecture search phase because they are associated with the distribution of input multimodal graph features. Consequently, these learned weights can be effectively transferred and generalized across architectures. Our findings yield two important insights: Firstly, the optimal weights learned from one MGNN architecture can be used as a less biased multimodal graph feature reweighting scheme for another architecture. Secondly, we notice that the learning speed of the optimal weight varies across different MGNN architectures, with some models struggling to effectively learn decorrelated features. Based on these insights, we recommend the adoption of a global weight across different model architectures. Specifically, after the training of an MGNN architecture, we retain the best global weight and use it as a warm start for the next MGNN architecture training process. For the i-th sampled architecture, the formula is shown below:

$$\begin{aligned} (A^{(i)}, W^{(i)}, \omega^{(i)}) &= \arg\min_{A,W,\omega} \mathcal{L}_{train} | \omega^*, \\ \omega^* = \omega^{(i)}, \mathcal{L}^*_{val} &= \mathcal{L}^{(i)}_{val}, \text{ if } \mathcal{L}^{(i)}_{val} < \mathcal{L}^*_{val} \end{aligned}, \quad (7)$$

where $A^{(i)}$, $W^{(i)}$ and $\omega^{(i)}$ represent the best architecture, optimized model parameters and sample weights at the i-th step, $\mathcal{L}^{(i)}_{val}$ refers to the best validation loss at the i-th step, ω^* and $\mathcal{L}^*_{val}$ represent the best global sample weights and the best global validation loss prior to the i-th step.

3.4 Search Algorithm

In this work, we employ a reinforcement learning-based search algorithm, which is a widely adopted strategy in many popular NAS algorithms. We utilize a recurrent neural network (RNN) as the controller to generate MGNN architectures. Once an architecture is generated, we construct and train an MGNN model based on this architecture and record its highest accuracy on the validation dataset. Subsequently, we optimize the parameters of the RNN controller, enabling it to generate better architectures over time.

Train the controller. Let $P(A;\theta)$ denote the distribution of architecture A parameterized by the choice of con-

troller θ, the goal is to maximize the expected accuracy $\mathcal{E}_{P(A;\theta)}[\mathcal{R}(A(W^*, G))]$, while minimizing the training loss $\mathcal{L}_{train}(A(W, G))$. This process can be formulated as a three-level optimization problem outlined below:

$$\begin{aligned} &\max_{A\in\mathcal{A}} \mathcal{E}[\mathcal{R}_{val}(A(W^*, G); \omega^*)], \\ s.t.\ \ &W^* = \arg\min_{W} \mathcal{L}_{train}(A(W, G); \omega^*), \\ &\omega_t^* = \arg\min_{\omega_t} \sum_{1\le i<j\le m_Z^t} \|\hat{\Sigma}^{\omega^t}_{Z^t_{*i}, Z^t_{*j}}\|_F^2, \\ &\omega_v^* = \arg\min_{\omega_v} \sum_{1\le i<j\le m_Z^v} \|\hat{\Sigma}^{\omega^v}_{Z^v_{*i}, Z^v_{*j}}\|_F^2, \end{aligned} \tag{8}$$

where $\mathcal{A}$ represents the search space of the neural architectures, W^* represents the optimal trainable parameters for architecture A. $\mathcal{R}_{val}$ measures the performance (e.g., accuracy) of architecture A on the validation dataset, which is used as the reward in reinforcement learning.

4 Experiment

In this section, we perform various experiments to verify the effectiveness of the proposed OMG-NAS method.

4.1 Datasets

We evaluate our OMG-NAS on three challenging real-world multimodal graph OOD datasets: Tencent dataset, Amazon review dataset and Recipe dataset. More details about datasets are provided in Appendix.

Tencent dataset: We extract the articles spreading network from Tencent WeChat official accounts, where each node represents an article and has two modalities: visual head images and textual titles. We establish connections between two articles if at least one user has viewed both of them. The objective of this task is to identify and detect low-quality articles for different network domains.

Amazon review dataset: We extract both user-generated reviews and product images from the famous Amazon e-commerce platform[1]. We classify ratings equal to or greater than 4 as positive feedback and ratings less than 2 as negative feedback. Each review in the graph has two modalities and is connected to other reviews based on whether they belong to the same or similar products. The task is to categorize each review as either positive or negative.

For Tencent dataset and Amazon dataset, we use the open-source implementations (Wolf et al. 2020) of pre-trained Bert (Devlin et al. 2018) to extract the textual features and pre-trained Vision Transformer (ViT) (Dosovitskiy et al. 2020) to extract the visual features.

Recipe dataset: We collect recipes data from 3 popular cooking websites[2][3][4] with relevant text and images. The extracted text includes titles, lists of ingredients, and cooking instructions, while the images showcase raw materials, manufacturing processes, and finished products. We partition each image into 16*16 small blocks as visual nodes and divide the text into words as textual nodes (Huang et al. 2019; Han et al. 2022). We aim to classify each recipe into the corresponding food label such as cakes and beverages.

[1]https://www.Amazon.com

[2]https://www.simplyrecipes.com/

[3]https://www.allrecipes.com/

[4]https://www.thespruceeats.com/

4.2 Experimental Settings

Evaluation Tasks and Metrics. We consider Tencent dataset, Amazon review dataset for node classification task, and Recipe dataset for graph classification task. The evaluation metric is the classification accuracy of the test datasets.

OOD Settings. For each task, we perform experiments in both Multi-OOD and Single-OOD settings since the occurrence of both Multi-OOD and Single-OOD is common in the real world. According to existing works that focus on text-OOD (Yang et al. 2022; Wang et al. 2021a), image-OOD (Wang et al. 2021b; Zhang et al. 2022a), and multimodal-OOD scenarios (Sun et al. 2022), we identify two situations where OOD could potentially occur:

1. Multimodal OOD (Multi-OOD): This term refers to the scenario where the training dataset and testing dataset come from different domains, causing distribution shifts in both modalities. For Amazon review dataset, different domains correspond to different types of products. For Recipe dataset, these domains could be determined based on the subcategories of food. For instance, one can train a model using chocolate cakes and evaluate its performance on fruit cakes.
2. Singlemodal OOD (Single-OOD): This term refers to the situation where one modality of the test data exhibits a different distribution than that of the training data. For instance, distribution shifts in the visual modality can occur due to changes in color, background, or shape, whereas in the textual modality, distribution shifts can be attributed to variations in words, named entities, or sentiments.

Baselines. We compare our model with baselines from the following three different categories.

- Manually designed MGNNs: we include the MGNNs in our search space as baselines, i.e., GCN, GAT and MGAT (Tao et al. 2020).
- OOD generalization methods for GNNs: we consider Mixup (Wang et al. 2021c), OOD-GNN (Li et al. 2022a), EERM (Wu et al. 2022a), DIR (Wu et al. 2022b), and CIGA (Chen et al. 2022) along with manually designed MGNNs as baselines.
- Neural Architecture Search: we consider three baselines, Random Search and GraphNAS(Gao et al. 2019) with a single-modal search space using similar GNN cells, and MG-NAS (Cai et al. 2022) with the MGNN search space designed in this paper.

Implementation Details.

For the Tencent dataset, we set the number of epochs to 200, the learning rate to 0.001, and the dimensions of the representations and hidden layers to 768 for both the text modality and visual modality. For the Amazon dataset, we set the number of epochs to 100, choose the learning rate from $\{0.001, 0.005, 0.01\}$, and set the dimensions of the representations and hidden layers to 128 for both the text

Methods	Tencent	Amazon review				
	Multi-OOD	Multi-OOD-S	Multi-OOD-B	Multi-OOD-D	Multi-OOD-T	Single-OOD
GCN	55.88±0.96	79.42±7.68	79.79±9.01	55.82±2.01	60.90±1.83	57.87±4.11
GAT	55.89±5.50	78.09±2.79	80.42±3.78	55.60±3.44	56.16±2.08	59.38±2.38
MGAT	59.83±4.60	67.83±9.17	72.71±7.50	48.00±4.06	53.60±7.01	61.35±3.16
Mixup	58.08±1.43	57.82±0.56	75.00±0.42	64.50±0.54	<u>70.55±2.72</u>	**76.12±1.31**
SRGNN	46.36±0.01	47.21±0.15	59.38±0.83	49.62±2.59	60.35±0.49	68.45±1.29
EERM	53.73±0.42	60.74±0.16	55.54±0.15	56.42±0.10	41.83±0.01	64.93±0.41
OOD-GNN + GCN	61.00±1.25	65.96±8.95	74.82±3.75	58.92±3.15	52.80±9.20	59.02±3.17
OOD-GNN + GAT	56.49±3.26	73.14±5.45	78.51±5.81	56.60±5.57	50.19±7.61	58.13±1.81
OOD-GNN + MGAT	60.06±7.99	66.30±5.17	74.54±6.63	60.26±7.69	56.55±5.28	62.67±2.50
Random Search	60.98±2.53	80.13±5.21	82.61±4.12	60.60±5.82	63.86±4.57	65.52 ±6.91
GraphNAS	62.41±3.35	81.89±5.32	83.47±3.98	61.49±5.34	63.51±4.89	67.10 ±7.59
MG-NAS	<u>64.25±3.45</u>	<u>85.13±4.68</u>	<u>86.75±3.32</u>	<u>64.97±4.97</u>	68.55±4.12	68.22±8.80
OMG-NAS (ours)	**66.82±1.35**	**88.40±2.13**	**88.47±2.82**	**68.46±1.27**	**71.65±1.93**	<u>75.56±5.41</u>

Table 1: Classification accuracy (%) on the Tencent dataset and Amazon review dataset. In each column, the boldfaced score denotes the best result and the underlined score represents the second-best result. ± denotes standard deviation. For Amazon review dataset, we select one domain as the target domain and the other three domains serve as source domains. We use the first letter to represent each target domain in a concise way, , where S stands for Shoes, B for Baby, D for Dress, and T for Trousers.

Methods	Recipe	
	Multi-OOD	Single-OOD
GCN / Edge	61.60±4.90	73.14±4.19
GAT / Mr	53.15±6.55	69.75±3.08
MGAT / Sage	66.80±1.90	75.00±5.56
Mixup	<u>69.54±2.21</u>	75.39±1.56
DIR	60.14±2.75	73.90±1.57
CIGA	67.82±1.98	74.38±1.42
OOD-GNN + GCN / Edge	61.65±5.05	75.45±1.91
OOD-GNN + GAT / Mr	53.73±3.24	71.57±1.82
OOD-GNN + MGAT / Sage	65.30±1.15	<u>76.02±2.49</u>
Random Search	64.64±5.05	72.41±5.80
GraphNAS	64.91±5.92	72.07±5.43
MG-NAS	68.84±4.52	75.82±4.12
OMG-NAS (ours)	**75.70±2.48**	**76.53±2.73**

Table 2: Classification accuracy (%) on Recipe dataset.

modality and visual modalities. For the Recipe dataset, the number of epochs is set to be 50, the batch size is selected from {8,16,64}, and the learning rate is chosen from {0.001, 0.005, 0.01}. The dimensions of the representations and hidden layers are set to 200 for the text modality and 128 for the visual modality. The number of epochs for learning weights in MGFD is set to be 30 for all datasets. We utilize a two-layer MLP classifier. We report the mean values with standard deviations from 5 repeated experiments.

4.3 Results Analysis and Comparison

Table 1 and Table 2 display a comparison of our proposed method OMG-NAS with the baseline methods on three real-world datasets. The results reveal that OMG-NAS achieves state-of-the-art performance in both Single-OOD and Multi-OOD settings. Firstly, due to the limited ability to learn domain invariant features, fixed MGNN models demonstrate relatively poor performance and high instability, as evidenced by lower accuracy and higher standard deviation. Secondly, when utilizing distribution generalization methods like OOD-GNN, it is important to consider the inconsistency between different modality distribution shifts. Failing to account for the mixed distribution shift of different modalities may result in worse outcomes as the model becomes susceptible to variant features. Furthermore, we conduct a comparison between OMG-NAS and various NAS techniques, namely random search, GraphNAS and MG-NAS. While these methods are successful in selecting the optimal architecture on the validation dataset, they often underperform on the test dataset due to the presence of spurious features. This can lead to suboptimal performance on the test dataset for the chosen architecture. These results demonstrate that OMG-NAS enhances the OOD generalization ability of MGNN models through automatic exploration of both architectures and multimodal weights.

We also conduct a comparison of OMG-NAS with baseline methods under unbalanced settings in Table 3. Three domains are used as source domains, while the remaining one is used as the target domain. We select Dress as the dominant source domain and adjust the ratio of data from the dominant domain and the other two domains, Trousers and Shoes. OMG-NAS consistently achieves the best performance under all ratios. These findings indicate that the statistical correlations between relevant and irrelevant features are strong enough to hinder generalization across domains when the size of domains is unbalanced. However, OMG-NAS is able to learn the true connections between features and labels by eliminating these correlations.

4.4 Ablation Study

We compare OMG-NAS against three variations in Table 4 to investigate the impact of different components of OMG-NAS, as detailed below:

- MGNN + SFD: We adopt a fixed MGNN that learns weights for each sample instead of learns weights for different modalities. Specifically, we concatenate the outputs of the multimodal GNN encoders and optimize the sample weights via Equation 3 to Equation 6.

Methods	Amazon Review				
	MD2	MD4	MD6	MD8	MD10
GCN	71.91±5.21	68.48±7.14	65.44±5.10	64.32±4.51	67.54±17.86
GAT	69.84±6.46	53.91±6.41	56.48±4.91	51.59±8.13	52.62±0.93
MGAT	70.21±8.32	58.14±6.98	67.13±9.24	59.15±2.41	76.17±1.85
OOD-GNN + GCN	69.91±5.83	64.48±4.74	68.98±6.87	59.01±3.61	52.13±6.93
OOD-GNN + GAT	60.39±7.90	61.84±5.61	57.91±5.31	62.67±4.63	60.96±6.33
OOD-GNN + MGAT	67.17±5.19	79.93±4.92	68.03±5.92	71.43±5.72	66.50±9.41
Random Search	83.61±4.32	83.58±6.70	76.25±5.19	73.23±6.30	71.79±9.27
GraphNAS	84.91±3.95	84.66±7.71	79.82±5.39	72.64±6.21	77.16±10.87
MG-NAS	86.55±3.06	85.33±2.93	82.17±6.51	86.15±1.97	78.79±9.21
OMG-NAS (ours)	**88.93±1.89**	**87.92±2.91**	**85.94±2.07**	**89.04±0.51**	**83.11±0.82**

Table 3: Predictive performance on Amazon-Review dataset (Unbalanced). MD2 indicates that the ratio of Dress, Trousers, and Shoes is 2:1:1 in both the training and validation data, and other notations with 'MD' are similar.

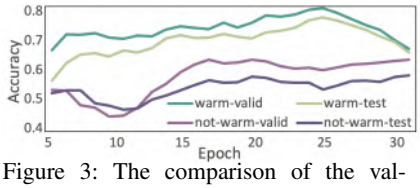


Figure 3: The comparison of the validation/test accuracy between transfer and non-transfer settings on the Recipe dataset.

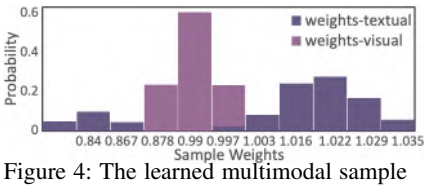


Figure 4: The learned multimodal sample weights distribution on Recipe dataset.

- MGNN + MGFD: We consider a fixed MGNN model with the MGFD method by learning intra-modal weights, as described in Section 3.2.
- OMG-NAS w/o GMSWE: On the basis of OMG-NAS, we incorporate a NAS method that eliminates the transfer of sample weights across architectures, as discussed in Section 3.3.

Methods	Tencent	Amazon	Recipe
GCN	55.88±0.96	60.90±1.83	61.60±4.90
GCN + SFD	61.00±1.25	52.80±9.20	61.65±5.05
GCN + MGFD	55.26±2.24	61.85±3.51	70.63±2.10
GAT	55.89±5.50	56.16±2.08	53.15±6.55
GAT + SFD	56.49±3.26	50.19±7.61	53.73±3.24
GAT + MGFD	62.68±4.45	56.85±8.28	54.98±1.47
MGAT	59.83±4.60	53.60±7.01	66.80±1.90
MGAT + SFD	60.06±7.99	56.55±5.28	65.30±1.15
MGAT + MGFD	65.66±1.41	58.30±4.97	65.93±0.60
OMG-NAS w/o GMSWE	62.54±3.11	63.28±6.53	72.85±3.54
OMG-NAS (ours)	**66.82±1.35**	**71.65±1.93**	**75.70±2.48**

Table 4: Ablation experiments on Multi-OOD dataset.

Architectures	WT	w/o WT	Improvement
GCN → GCN	65.19±9.31	64.03±9.21	**+1.8%**
GCN → GAT	70.55±6.50	65.80±5.93	**+7.22%**
GCN → MGAT	74.15±3.90	73.82±2.50	**+4.40%**
GAT → GCN	54.68±8.13	58.27±4.10	-5.80%
GAT → GAT	69.25±4.42	58.85±6.51	**+17.7%**
GAT → MGAT	73.35±7.46	70.13±6.69	**+4.60%**
MGAT → GCN	74.68±0.47	72.85±1.25	**+2.51%**
MGAT → GAT	69.00±10.9	64.13±9.10	**+7.60%**
MGAT → MGAT	73.28±4.74	72.38±1.62	**+1.25%**

Table 5: Weight transfer experiments on Recipe dataset. WT means with-transfer setting and w/o WT means without-transfer setting.

Firstly, MGNN+MGFD outperforms MGNN+SFD in all datasets and MGNN architectures. This result highlights the effectiveness of the multimodal graph feature decorrelation component in OMG-NAS, which effectively distinguishes distribution shifts among different modalities and resolves the issue of overlapping spurious features between modalities. Secondly, OMG-NAS outperforms OMG-NAS w/o GMSWE in terms of all datasets, indicating the effectiveness of our proposed global sample weight estimator across architectures. We also demonstrate the effectiveness of weight transfer in Figure 3 and Table 5. In Figure 3, we compare the validation accuracy and test accuracy between transfer and not-transfer settings on Recipe dataset. In the transfer setting, we use the optimal weights obtained from training GAT as the initial sample weights of MGAT. In the NOT-transfer setting, we use random weight initialization for the sample weights of MGAT. To further investigate the effectiveness of the multimodal graph reweighting, we present the distribution of learned weights in OMG-NAS on the Recipe dataset. Figure 4 demonstrates that OMG-NAS successfully acquires meaningful weights, while the distribution of weights are obviously different for different modalities. In summary, OMG-NAS outperforms all the variations considered in this ablation study, demonstrating the importance of every components of our approach in achieving superior performance.

5 Conclusion

In this paper, we propose a novel OMG-NAS method to improve the OOD generalization ability of MGNAS. OMG-NAS disentangles multimodal features and reweights samples using random Fourier features. Additionally, it utilizes the diverse features of the searched models. All of these designs aim to eliminate spurious correlations and enable OOD generalization ability. Extensive experiments show the significant contribution of OMG-NAS in addressing the challenging generalization problem of MGNAS.

6 Acknowledgments

This work was supported by the National Key Research and Development Program of China No. 2020AAA0106300, National Natural Science Foundation of China (No. 62250008, 62222209, 62102222), Beijing National Research Center for Information Science and Technology under Grant No. BNR2023RC01003, BNR2023TD03006, and Beijing Key Lab of Networked Multimedia.

References

Abavisani, M.; Wu, L.; Hu, S.; Tetreault, J.; and Jaimes, A. 2020. Multimodal categorization of crisis events in social media. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Bai, H.; Zhou, F.; Hong, L.; Ye, N.; Chan, S.-H. G.; and Li, Z. 2021. Nas-ood: Neural architecture search for out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Cai, J.; Wang, X.; Guan, C.; Tang, Y.; Xu, J.; Zhong, B.; and Zhu, W. 2022. Multimodal continual graph learning with neural architecture search. In *Proceedings of the ACM Web Conference 2022*.

Chen, Y.; Zhang, Y.; Bian, Y.; Yang, H.; Kaili, M.; Xie, B.; Liu, T.; Han, B.; and Cheng, J. 2022. Learning causally invariant representations for out-of-distribution generalization on graphs. *Advances in Neural Information Processing Systems*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ding, M.; Kong, K.; Chen, J.; Kirchenbauer, J.; Goldblum, M.; Wipf, D.; Huang, F.; and Goldstein, T. 2021. A closer look at distribution shifts and out-of-distribution generalization on graphs.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Fan, S.; Wang, X.; Mo, Y.; Shi, C.; and Tang, J. 2022. Debiasing graph neural networks via learning disentangled causal substructure. *Advances in Neural Information Processing Systems*.

Fang, T.; Lu, N.; Niu, G.; and Sugiyama, M. 2020. Rethinking importance weighting for deep learning under distribution shift. *Advances in neural information processing systems*.

Gao, D.; Li, K.; Wang, R.; Shan, S.; and Chen, X. 2020a. Multi-Modal Graph Neural Network for Joint Reasoning on Vision and Scene Text. arXiv:2003.13962.

Gao, J.; Lyu, T.; Xiong, F.; Wang, J.; Ke, W.; and Li, Z. 2021a. Predicting the survival of cancer patients with multimodal graph neural network. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Gao, Y.; Yang, H.; Zhang, P.; Zhou, C.; and Hu, Y. 2019. Graphnas: Graph neural architecture search with reinforcement learning. *arXiv preprint arXiv:1904.09981*.

Gao, Y.; Yang, H.; Zhang, P.; Zhou, C.; and Hu, Y. 2020b. Graph Neural Architecture Search. In *IJCAI*.

Gao, Y.; Yang, H.; Zhang, P.; Zhou, C.; and Hu, Y. 2021b. Graph neural architecture search. In *International joint conference on artificial intelligence*. International Joint Conference on Artificial Intelligence.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*.

Han, K.; Wang, Y.; Guo, J.; Tang, Y.; and Wu, E. 2022. Vision gnn: An image is worth graph of nodes. *arXiv preprint arXiv:2206.00272*.

Huang, L.; Ma, D.; Li, S.; Zhang, X.; and Wang, H. 2019. Text level graph neural network for text classification. *arXiv preprint arXiv:1910.02356*.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Li, G.; Muller, M.; Thabet, A.; and Ghanem, B. 2019a. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE/CVF international conference on computer vision*.

Li, H.; Cui, P.; Zang, C.; Zhang, T.; Zhu, W.; and Lin, Y. 2019b. Fates of microscopic social ecosystems: Keep alive or dead? In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Li, H.; Wang, X.; Zhang, Z.; Ma, J.; Cui, P.; and Zhu, W. 2021a. Intention-aware sequential recommendation with structured intent transition. *IEEE Transactions on Knowledge and Data Engineering*.

Li, H.; Wang, X.; Zhang, Z.; Yuan, Z.; Li, H.; and Zhu, W. 2021b. Disentangled contrastive learning on graphs. *Advances in Neural Information Processing Systems*.

Li, H.; Wang, X.; Zhang, Z.; and Zhu, W. 2022a. Ood-gnn: Out-of-distribution generalized graph neural network. *IEEE Transactions on Knowledge and Data Engineering*.

Li, H.; Wang, X.; Zhang, Z.; and Zhu, W. 2022b. Out-of-distribution generalization on graphs: A survey. *arXiv preprint arXiv:2202.07987*.

Li, H.; Zhang, Z.; Wang, X.; and Zhu, W. 2022c. Disentangled graph contrastive learning with independence promotion. *IEEE Transactions on Knowledge and Data Engineering*.

Li, H.; Zhang, Z.; Wang, X.; and Zhu, W. 2022d. Learning invariant graph representations for out-of-distribution generalization. In *Advances in Neural Information Processing Systems*.

Li, H.; Zhang, Z.; Wang, X.; and Zhu, W. 2023. Invariant Node Representation Learning under Distribution Shifts with Multiple Latent Environments. *ACM Transactions on Information Systems*.

Nunes, M.; and Pappa, G. L. 2020. Neural architecture search in graph neural networks. In *Intelligent Systems: 9th Brazilian Conference, BRACIS 2020, Rio Grande, Brazil, October 20–23, 2020, Proceedings, Part I 9*. Springer.

Pagliardini, M.; Jaggi, M.; Fleuret, F.; and Karimireddy, S. P. 2022. Agree to disagree: Diversity through disagreement for better transferability. *arXiv preprint arXiv:2202.04414*.

Peng, N.; Poon, H.; Quirk, C.; Toutanova, K.; and Yih, W.-t. 2017. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*.

Peng, X.; Wei, Y.; Deng, A.; Wang, D.; and Hu, D. 2022. Balanced multimodal learning via on-the-fly gradient modulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Qin, Y.; Wang, X.; Zhang, Z.; Xie, P.; and Zhu, W. 2022a. Graph neural architecture search under distribution shifts. In *International Conference on Machine Learning*. PMLR.

Qin, Y.; Zhang, Z.; Wang, X.; Zhang, Z.; and Zhu, W. 2022b. NAS-Bench-Graph: Benchmarking graph neural architecture search. *Advances in Neural Information Processing Systems*.

Rame, A.; Kirchmeyer, M.; Rahier, T.; Rakotomamonjy, A.; Gallinari, P.; and Cord, M. 2022. Diverse weight averaging for out-of-distribution generalization. *arXiv preprint arXiv:2205.09739*.

Shen, Z.; Cui, P.; Zhang, T.; and Kunag, K. 2020. Stable learning via sample reweighting. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Shi, M.; Tang, Y.; Zhu, X.; Huang, Y.; Wilson, D.; Zhuang, Y.; and Liu, J. 2022. Genetic-GNN: Evolutionary architecture search for graph neural networks. *Knowledge-Based Systems*.

Sun, T.; Wang, W.; Jing, L.; Cui, Y.; Song, X.; and Nie, L. 2022. Counterfactual reasoning for out-of-distribution multimodal sentiment analysis. In *Proceedings of the 30th ACM International Conference on Multimedia*.

Tao, Z.; Wei, Y.; Wang, X.; He, X.; Huang, X.; and Chua, T.-S. 2020. Mgat: Multimodal graph attention network for recommendation. *Information Processing & Management*.

Teney, D.; Abbasnejad, E.; Lucey, S.; and Van den Hengel, A. 2022. Evading the simplicity bias: Training a diverse set of models discovers solutions with superior ood generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, T.; Sridhar, R.; Yang, D.; and Wang, X. 2021a. Identifying and mitigating spurious correlations for improving robustness in nlp models. *arXiv preprint arXiv:2110.07736*.

Wang, T.; Zhou, C.; Sun, Q.; and Zhang, H. 2021b. Causal Attention for Unbiased Visual Recognition. *CoRR*, abs/2108.08782.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*.

Wang, Y.; Wang, W.; Liang, Y.; Cai, Y.; and Hooi, B. 2021c. Mixup for node and graph classification. In *Proceedings of the Web Conference 2021*.

Wen, H.; Ding, J.; Jin, W.; Wang, Y.; Xie, Y.; and Tang, J. 2022. Graph neural networks for multimodal single-cell data integration. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, 38–45.

Wu, Q.; Zhang, H.; Yan, J.; and Wipf, D. 2022a. Handling distribution shifts on graphs: An invariance perspective. *arXiv preprint arXiv:2202.02466*.

Wu, Y.-X.; Wang, X.; Zhang, A.; He, X.; and Chua, T.-S. 2022b. Discovering invariant rationales for graph neural networks. *arXiv preprint arXiv:2201.12872*.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

Yang, L.; Zhang, S.; Qin, L.; Li, Y.; Wang, Y.; Liu, H.; Wang, J.; Xie, X.; and Zhang, Y. 2022. GLUE-X: Evaluating Natural Language Understanding Models from an Out-of-distribution Generalization Perspective. *arXiv preprint arXiv:2211.08073*.

Zhang, C.; Zhang, M.; Zhang, S.; Jin, D.; Zhou, Q.; Cai, Z.; Zhao, H.; Liu, X.; and Liu, Z. 2022a. Delving deep into the generalization of vision transformers under distribution shifts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Zhang, X.; Cui, P.; Xu, R.; Zhou, L.; He, Y.; and Shen, Z. 2021. Deep stable learning for out-of-distribution generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Zhang, Z.; Wang, X.; Guan, C.; Zhang, Z.; Li, H.; and Zhu, W. 2022b. Autogt: Automated graph transformer architecture search. In *The Eleventh International Conference on Learning Representations*.

Zhang, Z.; Wang, X.; Zhang, Z.; Shen, G.; Shen, S.; and Zhu, W. 2023. Unsupervised graph neural architecture search with disentangled self-supervision. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhao, Y.; Wang, D.; Bates, D.; Mullins, R.; Jamnik, M.; and Lio, P. 2020a. Learned low precision graph neural networks. *arXiv preprint arXiv:2009.09232*.

Zhao, Y.; Wang, D.; Gao, X.; Mullins, R.; Lio, P.; and Jamnik, M. 2020b. Probabilistic dual network architecture search on graphs. *arXiv preprint arXiv:2003.09676*.

Zhou, K.; Huang, X.; Song, Q.; Chen, R.; and Hu, X. 2022. Auto-gnn: Neural architecture search of graph neural networks. *Frontiers in big Data*.

Zhu, Q.; Ponomareva, N.; Han, J.; and Perozzi, B. 2021. Shift-Robust GNNs: Overcoming the Limitations of Localized Graph Training Data. arXiv:2108.01099.

We provide detailed information about the complexity analysis, search space of MG-NAS and OMG-NAS in Appendix B, training procedures of OMG-NAS in Appendix C, datasets in Appendix D, parameter settings in Appendix E and the OMG-NAS algorithm in Algorithm 1.

A Complexity Analysis

We analyze the computational complexity of OMG-NAS as follows. Let $|V|, |E|$ denote the total number of nodes and edges, d is the multimodal hidden dimension, T is the number of evaluated architectures, C is the number of cells and d_c is the hidden dimension of LSTM. The time complexity of OMG-NAS contains MGNN evaluation $O(T(|E|d+|V|d^2))$, GMSWE $O(T|V|)$ and controller learning $O(Cd_c^2)$. The overall time complexity of OMG-NAS is $O(T(|V| + |E|d + |V|d^2) + Cd_c^2)$. MGNNs has $O(d^2)$ parameters, the GMSWE has $O(|V|)$ parameters and the controller has $O(d_c^2)$ parameters. Thus, the total number of learnable parameters is $O(d^2 + d_c^2 + |V|)$. In summary, our method is on par or more computationally efficient compared to existing NAS-OOD works (Qin et al. 2022a; Bai et al. 2021) and GNNs.

B Details of Search Space

The MGNN model in our OMG-NAS consists of three components - GNN cells, multimodal fusion cell and prediction layer.

B.1 GNN Cells

In GNN cells, we model the information propagation and aggregation under different modalities. We represent the data from each modality as a graph $G_m = (\mathcal{U}_m, E_m)$ with edge set $E_m = \{\langle i, j\rangle\}$. $m \in \mathcal{M} = \{v, t\}$ denote the visual and textual modality, respectively.

The GNN cell in the l-th layer of modality m updates node feature $h_{u,m}$ for each node u by aggregating its neighborhoods as

$$a_{u,m}^{(l)} = \text{AGGREGATE}(h_{v,m}^{(l-1)} : v \in \mathcal{N}(u)), \tag{9}$$

$$h_{u,m}^{(l)} = \text{ACTIVATE}(W_m^{(l)} \cdot a_{u,m}^{(l)}), \tag{10}$$

where AGGREGATE is the aggregation function including attention computation and aggregation operation, ACTIVATE is the activation function. $W_m^{(l)}$ is the network weight, $h_{v,m}^{(l-1)}$ is the output of the last layer or the input feature for $l = 1$, $\mathcal{N}$ is the receptive field (the set of neighboring nodes) of the node u.

For AGGREGATE function, we firstly calculate attention coefficients $e_{u,v,m}^{(l)}$ (also called correlation coefficients in some papers) for each node $v \in \mathcal{N}(u)$, then we aggregate the information from neighborhoods. Formally,

$$a_{u,m}^{(l)} = Agg(\{e_{u,v,m}^{(l)} h_{v,m}^{(l-1)} : v \in \mathcal{N}(u)\}), \tag{11}$$

where Agg is the message computation operator that aggregates information from neighborhood. Any permutation invariant operators, such as mean, max, sum and mlp can be used, and non-linear transformations are applied before and/or after the aggregation to increase expressive power (Gao et al. 2020b). We consider several attention coefficients $e_{u,v,m}^{(l)}$ and list several primitive operations as follows. Note that we use σ to represent the aggregation operation for the following formulas.

Primitive operations.

(1) GNN. The key idea of GNN is to generate node embeddings based on local neighborhoods. GNN firstly averages neighbors' previous layer embeddings, then adds the previous layer embedding of node u. After a nonlinear function we will get the l-th layer embedding of node u:

$$h_u^{(l)} = \sigma(W^{(l)} \sum_{v \in \mathcal{N}(u)} \frac{h_v^{(l-1)}}{|\mathcal{N}(u)|} + B^{(l)} h_u^{(l-1)}). \tag{12}$$

(2) GCN. GCN is one type of GNN based on the neighborhood aggregation idea (Kipf and Welling 2016). Instead of simple averaging, GCN calculates various normalization coefficients across neighbors and uses the same transformation matrix for node u and neighbor embeddings.

$$h_u^{(l)} = \sigma(W^{(l)} \sum_{v \in \mathcal{N}(u) \cup u} \frac{h_v^{(l-1)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}}) \tag{13}$$

(3) GAT. For GAT(Veličković et al. 2017) , the attention coefficients are computed on the pair-wise attention

$$h_u^{(l)} = \sigma(W^{(l)} \sum_{v \in \mathcal{N}(u)} \alpha_{uv} h_v^{(l-1)}), \tag{14}$$

where the neighbor nodes of the central node are normalized using softmax function

$$\alpha_{uv} = \frac{exp(e_{uv})}{\sum_{k \in \mathcal{N}(u)} exp(e_{uk})}. \tag{15}$$

GAT also uses a one-layer feed-forward neural network to calculate the importance of neighbor nodes to central nodes:

$$e_{uv} = \text{LeakyReLU}(W_a h_u^{(l-1)} + W_b h_v^{(l-1)}). \tag{16}$$

(4) MGAT. Existing methods fail to learn the propagation patterns of multimodal graphs from a multimodal perspective. To this end, according to Tao et al. (Tao et al. 2020), we introduce the gated attention mechanism into information transmission:

$$h_{u,m}^{(l)} = \sigma(\sum_{v \in \mathcal{N}} f_a(u, v) f_g(u, v) W^{(l)} h_{v,m}), \tag{17}$$

where $f_a(u, v)$ and $f_g(u, v)$ represent the attention part and the gate part of the gated attention network respectively. The former represents the contribution of node v to node u, and the latter determines whether information is propagated from node v to node u. We use inner product gate to represent the close relationship between node u and v:

$$f_g(u, v) = \sigma(\frac{h_{u,m}^T h_{v,m}}{\sqrt{d_u}}), \tag{18}$$

where d_u is the out degree of node u.

Primitive operations for visual graph convolution.

(1) Max-Relative GraphConv (Li et al. 2019a) (Mr). This method reduces the limitations of GCNs in processing non-Euclidean distance data and significantly im-

Algorithm 1: OMG-NAS: Out-of-distribution Generalized Multimodal Graph Neural Architecture Search

Input: Multimodal graph dataset $\mathcal{D}^{(tr)}, \mathcal{D}^{(val)}$
Output: Optimal MGNN model (A^*, W^*), optimal global weights ω^*;

```
for t ← 1 to T do
    while not converge do
        Controller samples architectures set 𝒜 from search space;
        for A in 𝒜 do
            ω ← Init_with(ω*);
            for e ← 1 to Epoch do
                for sampled minibatch do
                    for e' ← 1 to Epoch_Reweight do
                        Optimize the weights by minimizing Equation 4;
                    end
                    Back propagate with weighted prediction loss Equation 6;
                    Modulate gradient as Equation 39 and Equation 40;
                    Save features and weights;
                end
            end
            Update ω* as Equation 7;
        end
        Train controller;
    end
    W*, ω* ← Train MGNN with architecture A* and global weights ω*;
end
```

proves performance in point cloud semantic segmentation tasks. To construct a network that allows GCN to stack deeper layers without suffering from gradient vanishing, three deep GCN algorithms have been proposed: residual/dense connections, and dilated convolutions.

$$h_u^{(l)} = \sigma(W^{(l)}[h_u^{(l-1)}, max(\{h_v - h_u | v \in \mathcal{N}\})]) \quad (19)$$

(2) EdgeConv(Wang et al. 2019) (Edge). EdgeConv applies a channel-wise operation on the edge features associated with all the edges emanating from each vertex. The following is an asymmetric edge function which explicitly combines global shape structure, captured by the coordinates of the patch centers with local neighborhood information:

$$h_u^{(l)} = \sigma(\Theta^{(l)}[h_u^{(l-1)}, h_v - h_u] | v \in \mathcal{N}(u)]), \quad (20)$$

where $\Theta = (\theta_1, \cdots, \theta_M)$ encodes the weights of M different filters.

(3) GraphSAGE (Sage). Hamilton et al.(Hamilton, Ying, and Leskovec 2017) propose a general inductive learning framework called GraphSAGE, which does not directly learn node embedding vectors as previous GNNs did, but instead learns functions for sampling and aggregating from neighboring nodes, results in the ability of inductive learning. We concatenate self-embeddings with neighbor embeddings, and adopt the maximum operation as the general aggregator.

$$h_u^{(l)} = \sigma([W^{(l)} max(h_v^{(l-1)} | v \in \mathcal{N}(u)), B_k h_u^{(l-1)}]) \quad (21)$$

(4) GIN. Xu et al.(Xu et al. 2018) prove the strict upper limit of the expressive power of GNN variants that perform neighborhood aggregation (also known as message passing), and design the most powerful GNN named GIN under this framework:

$$h_u^{(l)} = \sigma(\mathrm{MLP}^{(l)}((1+\epsilon^{(l)})h_u^{(l-1)} + \sum_{v \in \mathcal{N}(u)} h_v^{(l-1)})). \quad (22)$$

B.2 Multimodal Fusion Cells

While GNN cells learn the graph propagation mode of a single mode, multimodal fusion cells learn how different modalities interact with each other and lead to better multimodal GNN models. Consequently, the integration of multimodal fusion cells can enhance the performance of MGNN models. In each such fusion layer, two inputs are combined, namely, the output from single modal GNN cells for modality m_1, and the output from single modal GNN cells for modality m_2:

$$h_f^{(l)} = \mathrm{FUSION}(h_{m_1}, h_{m_2}). \quad (23)$$

Primitive operations. All the primitive FUSION operations take two tensor inputs h_{m_1}, h_{m_2}, and outputs a tensor $\widetilde{h}_f$.

(1) Cross-attention. A modality may contain uninformative or even misleading information, resulting in negative messaging. Cross-attention (Abavisani et al. 2020) can filter the uninformative and misleading components from a weak modality. Firstly, we use a full connection layer to project the single-mode m feature:

$$h'_{m_i} = \mathrm{FC}(W_{m_i} h_{m_i} + b_{m_i}). \quad (24)$$

Secondly, we calculate an attention mask α_{m_i} that completely dependent on single modality:

$$\alpha_{m_1} = \sigma(W_{m_1}^T [h'_{m_1} | h'_{m_2}] + b_{m_1}), \quad (25)$$

$$\alpha_{m_2} = \sigma(W_{m_2}^T[h'_{m_1}|h'_{m_2}] + b_{m_2}). \quad (26)$$

We then obtain the weighted feature representations:

$$\widetilde{h}_{m_1} = \sum \alpha_{m_1} h'_{m_1}, \widetilde{h}_{m_2} = \sum \alpha_{m_2} h'_{m_2}. \quad (27)$$

Finally, we concatenate $\widetilde{h}_{m_1}$ and $\widetilde{h}_{m_2}$:

$$\text{FUSION}(h_{m_1}, h_{m_2}) = [\widetilde{h}_{m_1}|\widetilde{h}_{m_2}]. \quad (28)$$

(2) Co-attention. Different from cross-attention, co-attention jointly performs visual attention and image attention to model visual and textual information simultaneously. The attention mask is calculated as follows:

$$\alpha_{m_1} = \sigma(W_{m_1}^T h_{m_1} + b_{m_1}), \quad (29)$$

$$\alpha_{m_2} = \sigma(W_{m_2}^T h_{m_2} + b_{m_2}). \quad (30)$$

The rest of the operation steps are similar to cross-attention.

B.3 High Order Spread

We stack additional layers for information propagation in order to leverage higher-order connectivity between nodes and further enhance the representation. Through GNN cells, we get single modality information of each node:

$$h_{u,m_i}^{(l)} = \text{GNNCell}(h_{u,m_i}^{(l-1)}). \quad (31)$$

Through fusion cell, we get multimodal fusion information of each node:

$$f_u^{(l)} = \text{FusionCell}(h_{m_1}^{(l-1)}, h_{m_2}^{(l-1)}). \quad (32)$$

These steps enable us to enrich the representation of node information by leveraging both within-modality and cross-modality connectivity between nodes in the graph.

B.4 Prediction Layer

After updating the representations of nodes in a particular modality m, we combine the representations of different modality into a new representation. This process can be expressed mathematically as follows:

$$h_u = h_{u,v}^{(L)}||h_{u,t}^{(L)}||h_f^{(L)}. \quad (33)$$

For node classification task, we let the concatenation of the output of the final GNN layers $h_{u,v}^{(L)}, h_{u,t}^{(L)}$ and fusion layer $h_f^{(L)}$ to be the input of the classification layer, and use a two-layer MLP to predict the label of each node u:

$$\hat{y}_u = softmax(tanh(W_2(tanh(W_1 h_u + b_1)) + b_2), \quad (34)$$

where W_1, W_2 are the trainable weight matrices and b_1, b_2 are the bias vectors. The softmax function is applied to obtain the final prediction score $\hat{y}_u$.

For graph classification task, we use a pooling layer such as max pooling and average pooling before the classification layer to aggregate the node information in the last layers and get the representation of the graph G:

$$h_G = \text{POOL}(\{h_j^{(L)} : j \in \mathcal{U}_G\}), \quad (35)$$

$$\hat{y}_G = softmax(tanh(W_2(tanh(W_1 h_G + b_1)) + b_2). \quad (36)$$

C Details of Training OMG-NAS

C.1 For Each Batch of Graph Classification Tasks.

In Section 3.3 we propose a global sample weight estimator across architectures. However, the weight learning process for graph classification tasks processes another challenge that in SGD optimization, only a batch of samples can be accessed at a time. To address this issue, we adopt the global-local graph weight estimator introduced in (Zhang et al. 2021; Li et al. 2022a). Let the uni-modal representation and the weights of batch L to be Z_L^m and w_L^m. After we sample an architecture, the global information (Z^m, w^m) are initialized. Given the global information (Z^m, w^m), we learn the weight of batch L while concatenating the global and local information:

$$Z_O^m = [Z^m \| Z_L^m], w_O^m = [w^m \| w_L^m]. \quad (37)$$

After that, we fuse the global uni-modal information and local uni-modal information by:

$$\begin{aligned} Z_L^m &\leftarrow \alpha_i Z^m + (1-\alpha_i) Z_L^m, \\ w_L^m &\leftarrow \alpha_i w^m + (1-\alpha_i) w_L^m, \end{aligned} \quad (38)$$

where α_i is a hyper-parameter that controls the weight of long-term global information in the training process. By utilizing this parameter, the global weights can be gradually updated, ensuring the consistency of the entire graph dataset.

C.2 Details of OGM-GE in Section 3.2

When one of the modalities displays a higher likelihood of being influenced by spurious features, MGNN tends to learn the spurious features of this modality over the other modality. To ensure fully learn from the two GNN encoders of MGNN, we want to balance the learning of the two modalities to prevent over-reliance on the spurious feature of a single modality. Consequently, balanced multimodal graph stable learning enables the fully learned embedding of all modalities, which facilitates the decorrelation of multimodal features.

Similar with Peng et al. (Peng et al. 2022), we design the discrepancy ratio ρ^t and ρ^v for textual and visual modality, which monitor the contribution discrepancy between textual and visual modalities.

$$\begin{cases} \rho^t = \dfrac{\sum_x (1_{y_i=0} \cdot \text{softmax}(f^t(x))_0 + 1_{y_i=1} \cdot \text{softmax}(f^t(x))_1)}{\sum_x (1_{y_i=0} \cdot \text{softmax}(f^v(x))_0 + 1_{y_i=1} \cdot \text{softmax}(f^v(x))_1)} \\ \rho^v = \dfrac{1}{\rho^t} \end{cases} \quad (39)$$

where $\text{softmax}(f^t(x))_k = \text{softmax}(W^t \cdot \phi^t(\theta^t, u^t) + b^t)_k$ represent the prediction probability made by MGNN that x belongs to label $k (k = 0, 1)$ under textual modality. Denominator of ρ^t is the approximated prediction of modality t to estimate uni-modal performance of MGNN and the numerator is that of modality v.

We integrate the coefficient ρ^t and ρ^v into widely used SGD optimization method, and θ^t, θ^v in iteration i is up-

Action	Operator	Value
Agg	sum	$\sum_{j\in N_u} h_u$
	mean	$1/\lvert N_u\rvert \sum_{j\in N_u} h_u$
	max	$\max_{j\in N_u} h_u$
	mlp	$\mathrm{MLP}((1+\epsilon)h_u + \sum_{v\in N(u)} h_v)$
Act	/	tanh, relu, identity, softplus, leaky_relu, relu6, elu
Att	const	$e_{uv}^{con} = 1$
	gcn	$e_{uv}^{gcn} = 1/\sqrt{d_u d_v}$
	gat	$e_{uv}^{gat} = \mathrm{LeakyReLU}(W_l * h_u + W_r * h_v)$
	sym-gat	$e_{uv}^{sgat} = e_{uv}^{gat} + e_{vu}^{gat}$
	mgat	$e_{uv}^{mgat} = e_{uv}^{gat} * \sigma(h_u * h_v/\sqrt{d_u d_v})$
	cos	$e_{uv}^{cos} = \langle W_l * h_u, W_r * h_u\rangle$
	linear	$e_{uv}^{lin} = tanh(sum(W_r * h_v))$
VisualConv	max-relative	$W^{(l)}[h_u^{(l-1)}, max(\{h_v - h_u \lvert v \in \mathcal{N}\})]$
	edgeconv	$\Theta^{(l)}[h_u^{(l-1)}, h_v - h_u] \lvert v \in \mathcal{N}]$
	graphsage	$[W^{(l)} max(h_v^{(l-1)} \lvert v \in \mathcal{N}), B_k h_u^{(l-1)}]$
	gin	$\mathrm{MLP}^{(l)}((1+\epsilon^{(l)})h_u^{(l-1)} + \sum_{v\in\mathcal{N}} h_v^{(l-1)})$

Table 6: Operators of search space $\mathcal{M}_{GNN}$

Operator	Value
cross-attention	Equation (29-33)
co-attention	Equation (34-35)
linearglu	$xW_1 \odot Sigmoid(yW_2)$
concatfc	$ReLU(Concat(x,y)W + b)$
sum	$Sum(x,y) = x + y$

Table 7: Operators of search space $\mathcal{M}_{Fusion}$. x and y represent input from different modalities, respectively.

dated as follows:

$$\begin{cases}\theta_{i+1}^t = \theta_i^t - (1 - 1_{\rho^t>1}\cdot \tanh(\alpha\cdot\rho^t))\cdot\eta\nabla_{\theta^t}L(\theta_i^t)\\ \theta_{i+1}^v = \theta_i^v - (1 - 1_{\rho^v>1}\cdot \tanh(\alpha\cdot\rho^v))\cdot\eta\nabla_{\theta^v}L(\theta_i^v)\end{cases} \tag{40}$$

where α is a hyper-parameter to control the degree of modulation, η is the learning rate, $\nabla_{\theta^t}L(\theta_i^t)$ is the gradient of modality t. By introducing ρ^t, ρ^v the optimization of modality with better performance on validation dataset is mitigated, while the other modality is not affected.

D Details of Dataset

After investigating existing multimodal graph datasets, we construct our own multimodal graph dataset under OOD settings leveraging existing datasets because there was no multimodal graph dataset specifically designed for OOD settings. In this section we will introduce the details of Tencent dataset, Amazon review dataset and Recipe dataset, including the dataset statistic in Table 8, Table 9 and Table 10.

Tencent dataset contains articles from two different domains which allow us to analyze the multimodal graph OOD ability. 1) Junk domain includes articles of poor quality and false contents, often with malicious intent or for promotional purposes. 2) Vulgar domain contains sexual activities, explicit sexual descriptions and images. 3) These two domains have different point densities and transmission mode as shown in Table 8.

	Domain	Nodes	Edges	Avg. Edges
Multi-OOD	Junk	2979	188332	63.22
	Vulgar	604	5094	8.43
	Total	**3583**	**-**	**-**

Table 8: Statistics of the Tencent article dataset.

	Domain	Nodes	Edges	Avg. Edges
Multi-OOD	Dress	1240	13392	10.8
	Trousers	787	5908	8.6
	Shoes	790	7176	10.4
	Baby	680	4964	7.3
	Total	**3497**	**-**	**-**
Single-OOD	Dress-Black	369	2731	7.4
	Dress-Red	331	1854	5.6
	Dress-Blue	318	1018	3.2
	Total	**1018**	**-**	**-**

Table 9: Statistics of the Amazon Review dataset.

	Domain	Graphs	Avg. Words
Multi-OOD	Chocolate Beverage	89	38.7
	Yogurt	53	35.3
	Drink	235	29.9
	Coffee	121	32.4
	Chocolate Cake	241	41.3
	White Cake	153	43.9
	Fruit Cake	149	46
	Others	46	34.2
	Total	**1087**	**-**
Single-OOD	Cake-Brown	238	41.2
	Cake-White	161	43.7
	Beverage-Brown	207	34.5
	Beverage-White	204	34.1
	Total	**810**	**-**

Table 10: Statistics of the Recipe dataset.

E Details of Parameter Settings

Hyper-parameters of the controller. We use a one-layer LSTM with 100 hidden units trained with the ADAM op-

timizer and a learning rate of 0.00035. After OMG-NAS searches S=50 architectures, we collect the best architecture and best global weights that achieve the best validation accuracy.

Hyper-parameters of MGNNs. After the controller has sampled an architecture, a child MGNN model will be built and trained for certain epochs. For Recipe dataset, the number of epochs is set to be 50, and the batch size is selected from {8,16,64}. The learning rate is chosen from {0.001, 0.005, 0.01}. The dimension of the representations and hidden layers are 200 for text modality and 128 for visual modality. For Amazon dataset, the number of epochs is set to be 100. The learning rate is chosen from {0.001, 0.005, 0.01}. The dimension of the representations and hidden layers are 128 for both text modality and visual modality. For Tencent dataset, the number of epochs is set to be 200. The learning rate is 0.001. The dimension of the representations and hidden layers are 768 for both text modality and visual modality.

Hyper-parameters for global weight learning. In the training phase, OMG-NAS learns a set of sample weights for each MGNN architecture. The parameters of the controller and the sample weights are optimized iteratively. To optimize the global weights, we set the number of epochs of learning sample weights to be 30. The learning rate is set to be 0.01 for Amazon rewiew dataset under Multi-OOD settings and 0.001 for other settings.