Leveraging Graph Neural Networks to Boost Fine-Grained Image Classification

Duy M. Le, Bao Q. Bui, Anh Tran, Cong Tran, and Cuong Pham

***Abstract*—Fine-grained image classification, which is a challenging task in computer vision, requires precise differentiation among visually similar object categories. In this paper, we present a novel approach that utilizes Graph Neural Network (GNN) to effectively integrate visual feature vectors extracted by a deep neural network (DNN) encoder. The utilization of a graph-based approach in our model facilitates the acquisition of contextual information and the exploration of relationships among images within a given batch. Consequently, this approach allows the discernment of vital visual features associated with a class label, features that may remain elusive when examining a singular image representative of that particular class. In practice, our proposed method demonstrates significant improvements in the accuracy of different fine-grained classifiers, with an average increase of $(+2.78\%)$ and $(+3.83\%)$ on the CUB200-2011 and Stanford Dog datasets, respectively, while achieving a state-of-the-art result (95.79%) on the Stanford Dog dataset. Despite not achieving the same level of improvement as in fine-grained image classification, our method still demonstrates its prowess in leveraging general image classification by attaining a state-of-the-art result of (93.71%) on the Tiny-Imagenet dataset. Furthermore, our method serves as a plug-in refinement module and can be easily integrated into different networks .**

***Index Terms*—Computer vision, fine-grained image classification, deep learning**

I. Introduction

A. Background and Motivation

FINE-GRAINED classification is an important task in computer vision as it has a wide range of real-world applications, including image recognition, disease diagnosis [1]–[3], or biodiversity monitoring [4]–[6], where distinguishing between visually similar subcategories is crucial. With the rapid advancement of technology, we now have the capability to collect and store a large amount of image data from various sources. Nevertheless, the classification of objects in images characterized by a high degree of similarity, commonly referred to as the *fine-grained image classification* problem—examples include the categorization of bird species, types of leaves, or models of electronic products—presents a noteworthy and persistent challenge.

In addition to image classification in general, fine-grained image classification exposes more significantly challenging, including (i) substantial intra-class variation, with objects in the same category exhibiting significant pose and viewpoint differences; (ii) subtle inter-class distinctions, where objects from different categories may closely resemble each other with minor differences; (iii) constraints on training data, as labeling fine-grained categories often demands specialized expertise and a substantial amount of annotation effort [7]. For these reasons, fine-grained classification remains an open topic for research community.

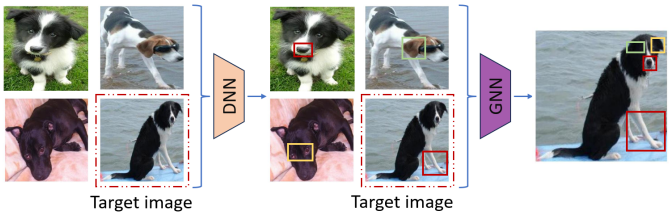


Fig. 1: Example of intra-batch feature fusion to enhance predictivity for target images.

Motivated by the fact that deep neural networks (DNNs) encounter challenges in effectively distinguishing intricate features and grappling with the inherent complexities of learning detailed patterns, our study centers on scrutinizing the mechanisms by which image feature extractors discern nuanced features. Additionally, our investigation delves into the process of *explicitly* amalgamating these discerned features across a batch of images to construct intricate feature maps, ultimately enhancing the accuracy of fine-grained image classification. Figure 1 illustrates an example of our idea, integrating subtle features extracted from several images to generate a sophisticated feature. In the figure, the delineated bounding boxes within each image delineate regions of interest identified by the model for focused attention. Boxes sharing the same color signify images belonging to identical classes, while distinct colors indicate diverse classification categories. The proposed GPH architecture discerns specific connections between the reference image and others, facilitating the amalgamation of subtle features derived from various images, even those belonging to different categories, to complement the features of the target image. Concretely, as exemplified in the provided figure, features corresponding to the snout, ear, and eye regions extracted from three distinct images are harmonized with the feet region. This integration results in a cohesive feature representation that concentrates on multiple salient positions of the dog within the target image. Intuitively, we have the option to employ either the Attention mechanism or GNNs for this purpose, and comprehensive discussions and experiments on these approaches are presented in the subsequent sections.

Duy M. Le is with Post & Telecommunication Institute of Technology in Hanoi, Vietnam. (e-mail: leminhduy131002@gmail.com).

Bao Q. Bui is with VinUniversity, Vinhomes Oceans Park, Gia Lam District, Hanoi 10000, Vietnam (e-mail: bao.bq@vinuni.edu.vn).

Cong Tran is with Post & Telecommunication Institute of Technology in Hanoi 10000, Vietnam. (e-mail: congtt@ptit.edu.vn).

Cuong Pham is with Post & Telecommunication Institute of Technology in Hanoi 10000, Vietnam. (e-mail: cuongpv@ptit.edu.vn).

B. Main Contributions

This paper presents a **G**NN **P**ost-**H**oc (GPH) plugin that leverages the power of graph neural networks (GNNs) to enhance existing fine-grained image classification methods. We propose a network architecture that integrates GNN blocks into a conventional DNN architecture, allowing for the extraction of fine-grained features while maintaining the robustness and generalization capabilities. Our approach aims to capture intricate inter-dependencies between feature vectors from multiple images within a batch, effectively utilizing this information to combine these features into final features and classify images. By doing so, we aim to improve the classification accuracy, particularly in scenarios where intra-class variations and inter-class distinctions are subtle.

In addition, we provide a comprehensive investigation into the effectiveness of our proposed model, benchmarking it against state-of-the-art methods on widely recognized fine-grained classification datasets. We demonstrate that the incorporation of GNN blocks leads to substantial performance gains, showcasing the potential of this hybrid approach for fine-grained image classification tasks. It can be worth to notice that our proposed GPH plugin is general and can be employed to leverage accuracy in any image classification problem. Its generalization ability is demonstrated by experiments on the TinyImageNet dataset, in which we attain the state-of-the-art accuracy of 93.71% (see in Section V-F) by incorporating GPH with the DNN network backbone. However, GPH provides a higher level of improvement in fine-grained image classification tasks due to their significant intra-class variations and subtle inter-class distinctions.

Our contributions can be summarized as follows:

- We propose a novel post-hoc module, termed GPH, that effectively exploits subtle features from multiple images and combines them to generate sophisticated features. By enhancing the concentration on pivotal features, GPH activates more gradient backpropagation for both CNN-based and Attention-based backbones, resulting in improved classification performance.
- The proposed architecture can be easily integrated into various fine-grained classifiers for enhancing performance, whistle the model's processing time remain manageable.
- Our extensive experiments on publicly available datasets demonstrate the model's capability to enhance feature clustering and accuracy, while also achieving state-of-the-art results on the CUB200-2011 and Stanford Dogs datasets.

The remainder of this paper is organized as follows: Section II provides an overview of related work in the field of fine-grained classification and graph neural networks. In Section III, we present our proposed model architecture in detail. Section IV and V describes the experimental setup and presents empirical results. Finally, in Section VI, we discuss the implications of our findings and outline avenues for future research.

II. Related work

This section investigates the previous works in fine-grained image classification and graph neural networks.

A. General image classification

In the field of general image classification, numerous studies have been conducted to develop effective algorithms and techniques. One prominent approach is the use of deep convolutional neural networks (CNNs), which have demonstrated remarkable performance in accurately categorizing diverse images. Models such as AlexNet [8], VGGNet [9], and ResNet [10] achieved significant advancements in image classification tasks by leveraging deep architectures and incorporating techniques like batch normalization and residual connections. Recently, Vision Transformer (ViT) [11] and Swin Transformer (SwinT) [12] emerged as a powerful architecture for general image classification, leveraging the transformer model's ability to capture long-range dependencies in images.

B. Fine-grained image classification

Recent deep learning research on fine-grained classification has primarily focused on two main directions: convolutional neural networks (CNN)-based and visual attention-based methods.

CNN-based Fine-Grained Image Classification is commonly seen in general classification tasks and specifically in fine-grained classification problems. Common backbone CNN architectures such as MobileNet [13], DenseNet [14], ConvNeXt [15], and others can also be applied to fine-grained classification tasks. Furthermore, there exist unique approaches, such as implicitly separating the class-relevant foreground from the class-irrelevant background [16], that aim to enhance the performance of the model. In a notable achievement, the CNN-based P2P-Net model [17] achieved top performance accuracy on the CUB-200-2011 datasets.

Visual attention-based approaches aim to mimic human visual attention by selectively focusing on informative regions or features within an image. One of the pioneering models utilizing this mechanism, [18], uses two-level attention to concentrate on both overall image context and fine-grained details. More recently, a reinforcement learning-based fully convolutional attention localization network [19] adaptively selects multiple task-driven visual attention regions. This model is renowned for being significantly more computationally effective in both the training and testing phases. Furthermore, the ViT-NeT [20] model augments the explicability of Vision Transformers [11] by integrating a neural tree decoder, enabling the generation of predictions with hierarchical structures that facilitate improved comprehension and examination of the model's decision-making process. In another context, HERBS [21] employs two innovative approaches, namely high-temperature refinement and background suppression, to address key challenges in fine-grained classification. Notably, the Attention-based TransFG model [22] demonstrated exceptional accuracy in fine-grained image classification datasets, attaining top performance. Currently, the ViT-NeT and HERB

models achieved the highest accuracies on the Stanford Dogs dataset [23], and the NABirds dataset [24], respectively.

C. Graph neural networks

Graph neural networks (GNN) can be categorized into four types, which encompass: convolutional graph neural networks (ConvGNNs), recurrent graph neural networks (RecGNNs), graph autoencoders (GAEs), and spatial-temporal graph neural networks (STGNNs). Inspired by the success of CNNs in computer vision, numerous methods have emerged to redefine convolution for graph data. These methods, collectively known as Convolutional Graph Neural Networks (ConvGNNs), can be categorized into two main streams: spectral-based and spatial-based approaches. Since the pioneering work on spectral-based ConvGNNs was presented by [25]; various advancements, extensions, and approximations have been made in spectral-based ConvGNNs including GCN [26], AGCN [27], and DualGCN [28]. On the other hand, Spatial-based ConvGNNs define graph convolutions based on a node's spatial relations (e.g., [29]–[31]). From a different perspective, spatial-based ConvGNNs share a similar concept of information propagation and message passing with RecGNNs. Furthermore, alongside RecGNNs and ConvGNNs, several other GNN variants have been devised, including Graph Autoencoders (GAEs) [32] and Spatial-Temporal Graph Neural Networks (STGNNs) [33]. Therefore, GNNs are employed in a wide range of applications, including Skeleton-Based Human Motion Prediction [34]. Although, these works often utilized GNN for exploiting the relational structure within a collection or set of elements, particularly in few-shot and semi-supervised learning scenarios (i.e. [35]), they do not significantly improve the performance of fine-grained and common image classification tasks.

III. Proposed Approach

A. Problem Definition

For the problem of fine-grained image classification, similar to the general image recognition, we are given a training dataset $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^{N}$ drawn from an unknown joint data distribution defined on $\mathcal{X} \times \mathcal{Y}$, with $\mathcal{X} \subset \mathbb{R}^{3\times H\times W}$ and $\mathcal{Y} \subset \{0,1\}^C$ denoting the input image space and the output label space (H, W denoted as height and width of an image in $\mathcal{X}$). In particular, the label space $\mathcal{Y}$ - which contains one-hot classification vectors, is the union space of all the C subspaces corresponding to the C subordinate categories of the same meta-category, i.e., $\mathcal{Y} = \mathcal{Y}_1 \cup \mathcal{Y}_2 \cup \cdots \cup \mathcal{Y}_c \cup \cdots \cup \mathcal{Y}_C$. Our goal is to learn a mapping function $f : \mathcal{X} \to \mathcal{Y}$ that correctly classifies images into one of the C categories.

B. GPH Architecture Design

In order to improve the model's understanding of complex image relationships and bolster its capability to distinguish subtle variations in fine-grained classification tasks, we propose a *simple yet effective* network architecture that utilizes a plug-in module based on GNNs. Figure 2 illustrates the workflow of our proposed design, in which the GNN encoder can be considered as a post-hoc plug-in. We first utilize a DNN-based encoder to generate feature vectors. These vectors are then constructed into a complete graph and input into a GNN model to obtain GNN embeddings, aiming to enhance the discriminative ability between feature clusters. The two features from the two encoders are then combined and fed into fully connected layers for classification. It is worth noting that the GNN plug-in can be integrated into any mainstream backbone network such as DenseNet, Swin Transformer, and ConvNeXt. In this section, we utilize comprehensive insights into our GNN Post-Hoc structure, consisting of two primary components: the deep neural network encoder and the graph neural network encoder, along with the inference process.

In our network architecture, function f consists of three components: (1) a deep neural network encoder $\Phi : \mathcal{X} \to \mathbb{R}^m$ that maps each input image x_i to a l_2-normalized feature embedding z_i; (2) a graph neural network encoder that constructs a fully connected graph $\mathcal{G}$ from the obtained feature vectors within a batch $\mathbf{z} = \{z_i\}_{i=1}^{b}$ and then maps them to l_2-normalized feature embeddings $\mathbf{g} = \{g_i\}_{i=1}^{b}$ with $g_i \in \mathbb{R}^m$; (3) a classifier $\Psi : \mathbb{R}^m \to \mathbb{R}^C$ that maps each feature in the combined m-dimensional embeddings of z and g to a classification vector, where a cross-entropy loss can be applied after using a sigmoid function.

DNN encoder. This encoder can be a typical encoder in any DNN-based image classification methods. Given a training batch $\{x_i, y_i\}_{i=1}^{b}$ with batch size b, the images are fed into the feature extractor network, yielding l_2-normalized embeddings $\{z_i\}_{i=1}^{b}$: $z_i = \Phi(x_i)$

GNN encoder. We enhance the capability of conventional classification networks for fine-grained classification tasks by incorporating a graph neural network module after their feature extraction module.

We denote a fully connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, where $\mathcal{V}$ represents the set of images in each batch, i.e., $|\mathcal{V}| = b$, $\mathcal{E} = \{e_{ij}\}_{i,j=\overline{1,b}}$ is the set of edges connecting images, and $\mathcal{F} = \{z_1, z_2, ..., z_b\}$ is the node features in the graph.

Our proposed GPH can employ various GNN architectures as the GNN encoder, such as GraphTransformer [36] and GraphSAGE [37] to learn the node embeddings, which are described by the feature matrix in $Z \in \mathbb{R}^{b\times m}$. Specifically, the initial node representation, which is the set of DNN embeddings $\{z_i\}_{i=1}^{b}$, are passed through multiple layers, with each layer encompassing two critical functions: AGGREGATE, responsible for gathering information from the neighbors of each node, and COMBINE, tasked with updating the node representations by combining the aggregated information from neighbors with the current node representations.

Mathematically, the general flow of the GNN encoder can be expressed as follows:

- Initialization: $Z^{(0)} = \mathcal{F}$.
- For each layer l-th of the GNN encoder ($l = \overline{1, L}$ with L is the number of layers), we update the embeddings of the graph to have $Z^{(l)} = \{z_i^{(l)}\}_{i=1}^{b}$, which is encoded through two general functions (here, $z_i^{(0)}$ refers to z_i):

$$a_i^{(l)} = \mathbf{AGGREGATE}^{(l)} \left\{ e_{ij}, z_j^{(l-1)} : j \in \mathcal{N}(i) \right\} \quad (1)$$

$$z_i^{(l)} = \mathbf{COMBINE}^{(l)} \left\{ z_i^{(l-1)}, a_i^{(l)} \right\} \quad (2)$$

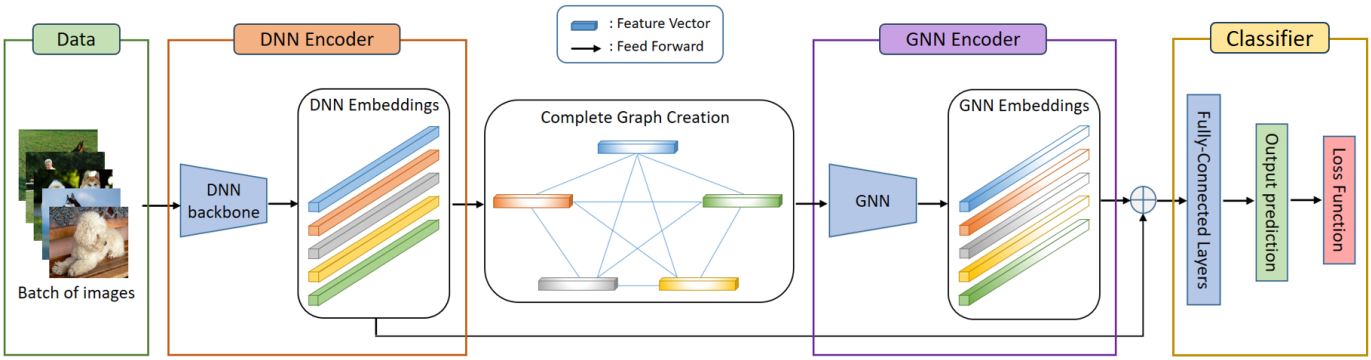


Fig. 2: GNN-based network post-hoc architecture

TABLE I: Dataset statistics. Imbalance is defined as the ratio of the number of images in the largest class to the number of images in the smallest class.

Dataset	# Train	# Test	Imbalance
CUB-200-2011	5,994	5,794	1.03
Stanford Dogs	12,000	8,580	1.00
NABirds	23,929	24,633	15.00

where $\mathcal{N}(i)$ is the set of neighbors for the i-th node. We note that the implementation of AGGREGATE and COMBINE functions can be varied depends on different GNNs.

Feature combination. The node representations $Z^{(L)}$ obtained at the last layer of the GNN encoder can be treated as the final node representations, and these features are subsequently merged with features from the DNN encoder $\{z_i\}_{i=1}^{b}$ as follows:

$$c_i = \textbf{COMBINE}\left\{z_i^{(L)}, z_i\right\}. \tag{3}$$

These final features $\{c_i\}_{i=1}^{b}$ are then passed through the classifier Ψ for classification.

IV. Datasets and Experimental Settings

A. Datasets

We perform experiments on three well-known fine-grained datasets: CUB-200-201 [38], Stanford Dogs [23], and NABirds [24]. First, the CUB-200-201 dataset, i.e., Caltech-UCSD Birds-200-201, comprises 11,788 labeled images of bird species. Originally, the dataset included 200 bird species, but the extended version incorporates extra images for each category, resulting in a total of 201 classes. This dataset also provides attribute labels and landmark annotations, which offer supplementary information for detailed analysis. Second, the Stanford Dogs dataset consists of 20,580 images featuring 120 distinct dog breeds, and it does not include meta-information similar to CUB-200-201. And finally, the NABirds dataset, short for "North American Birds Dataset," contains over 48,000 annotated images of 555 bird species found in North America. The division of training and testing data follows the predefined configurations in each dataset, with detailed quantities provided in Table I.

B. Implementation details

All experiments are conducted on an NVIDIA Tesla T4 GPU with 15GB of RAM. Initially, all input images are resized to 224x224 pixels. We employ simple data augmentation techniques such as RandomHorizontalFlip and RandomRotation during training. The DNN encoder is trained using pre-trained weights from the ImageNet1K dataset. For the GNN encoder, we integrate four blocks in total. The first block transforms the output features of the base encoder into embeddings with a size of 1024. The remaining three blocks further transform the features to ensure that the output features have a consistent dimension of 1024. The model is fine-tuned for 50 epochs using a batch size of 32 for all models. As the proposed GPH can be influenced by the batch size, we provide detailed experiments to evaluate the results corresponding to different batch size configurations in section V-B. We train the network using the Rectified Adam optimizer with a default epsilon value of e^{-8}. The dimension of the embedding of the encoder network is set to 1024. We evaluate the top-1 classification error on the shuffled validation set. Additionally, the initial learning rate is set to e^{-5}. In our experiment, we employ the sum function as COMBINE in Eq. 3 unless otherwise specified.[1]

V. Experimental Results

Our empirical studies in this subsection are designed to answer the following key research questions.

- **Q1.** To what extent does the GNN Post-Hoc model improve performance compared to regular classification networks and state-of-the-art fine-grained classification approaches?
- **Q2.** How do batch configurations affect the performance of the proposed model?
- **Q3.** How does integrating an additional GNN encoder with the DNN encoder impact the representation of feature vectors?
- **Q4.** How do different GNN encoders (GCN, GAT, GraphSAGE, and GraphTransformer) and the Attention mechanism affect the performance of the model across various benchmark datasets?

[1]The source code of the implementation is available online https://github.com/DuyMinhLe13/GNN_Post-hoc_method.

TABLE II: The impact of GPH on fine-grained classification outcomes when incorporated into various DNN techniques. The accuracy gain when applying GPH is provided in the brackets.

Method	Inference time	# params	Acc (%)		
			Stanford Dogs	CUB-200-2011	NABirds
MobilenetV3-S	0.013	1.6M	73.12	67.5	66.46
MobilenetV3-S-GPH	0.016	17.4M	77.01(+3.89)	69.86(+2.36)	69.1(+2.64)
MobilenetV3-L	0.035	4.4M	78.31	77.65	75.86
MobilenetV3-L-GPH	0.039	23.2M	82.72(+4.41)	80.77(+3.12)	79.82(+3.96)
Densenet201	0.28	18.3M	83.95	79.13	77.55
Densenet201-GPH	**0.29**	**73.7M**	87.72(+3.77)	84.48(+5.35)	83.81(+6.26)
Densenet161	**0.42**	**26.7M**	84.46	79.68	78.97
Densenet161-GPH	0.45	88.7M	88.47(+4.01)	84.79(+5.11)	84.75(+5.78)
SwinT-Small	0.51	49.1M	91.39	86.27	86.74
SwinT-Small-GPH	**0.52**	**61.7M**	92.79(+1.40)	87.35(+1.08)	87.97(+1.23)
SwinT-Big	**0.82**	**87.0M**	92.11	85.86	86.32
SwinT-Big-GPH	0.84	102.8M	93.06(+0.95)	87.9(+2.04)	88.03(+1.71)
ConvNeXtBase	0.59	88.7M	92.77	81.93	85.31
ConvNeXtBase-GPH	**0.61**	**103.4M**	94.56(+1.79)	87.52(+5.59)	87.86(+2.55)
ConvNeXtLarge	**1.22**	**197.9M**	93.71	81.74	85.53
ConvNeXtLarge-GPH	1.23	231.8M	95.79(+2.08)	87.8(+6.06)	88.11(+2.58)
HERB-SwinT	1.74	286.6M	88.62	89.9	90
HERB-SwinT-GPH	1.88	318.2M	88.9(+0.28)	90.37(+0.47)	90.61(+0.61)
TransFG	0.85	86.3M	89.18	90.19	89.9
TransFG-GPH	0.86	95.2M	89.76(+0.58)	90.66(+0.47)	90.45(+0.55)
P2P-Net	0.46	63.4M	83.14	86.07	85.12
P2P-Net-GPH	0.48	110.6M	89.3(+6.16)	91.07(+5.00)	90.56(+5.44)
Avg. Improvement			+2.51	+3.46	+3.04

- **Q5.** How does including edge attributes in GNN adjacency matrices affect the interpretability and accuracy of image connections within a batch?
- **Q6.** How do GPHs enhance accuracy in general image classification tasks?

A. Comparison with Existing Methods (Q1)

Baselines. To validate the effectiveness and generalization of our method, we investigate the performance of incorporating GPH on four different well-known DNNs and their variants, including DenseNet [14], MobileNet [13], ConvNeXt [15], SwinTransformer [12], and HERB [21]. It is important to highlight that our GPH is the only modification, while all other training configurations and hyperparameters remain unaltered from the original implementations. For consistency, we employ GraphTransformer as the GNN encoder for all experiments in this section. Even though we incorporate our proposed method across various techniques and assess it on diverse datasets, we maintain the consistent parameter configuration detailed earlier throughout all experiments.

Comparison results. Table II shows the impact of our GPH on fine-grained classification performance across different methods and datasets. Our interesting findings are summarized as follows:

- The table clearly illustrates that the incorporation of GPH consistently improves fine-grained classification results. Notably, we observe an average increase of $+2.78\%$, $+3.83\%$, and $+3.29\%$ on the Stanford Dogs, CUB-200-2011 datasets and NABirds, respectively.
- While GPH significantly enhances the performance of CNN-based models on both datasets, the improvement is more moderate for transformer-based models. We hypothesize that because of the inherent similarity between the attention mechanism of transformers and the nature of GNN, the accuracy improvement is not as substantial as with CNN-based models. For example, with models like DenseNet and MobileNet, accuracy increases by $3-6\%$ on both datasets, while with Swin Transformer, it ranges from $1-2\%$. Notably, ConvNeXt shows a slight performance boost on the Stanford Dogs dataset but a significant improvement of $5-6\%$ on CUB-200-2011.
- Improving existing fine-grained classification methods is a challenging endeavor. However, as shown in Table II, our proposed approach achieves new state-of-the-art results on the Stanford Dogs dataset[2]. It is important to highlight that the reported accuracies for the CUB-200-2011 and NABirds datasets slightly differ from the performance of state-of-the-art baselines as presented in their respective papers, i.e, HERB, TransFG, and P2P-Net. This discrepancy arises due to the fact that we train all models under the same optimizer and learning rate configurations in Section IV, which might not align with the settings used in the original papers.
- Additionally, we observe that for some models, when we add the GPH module to smaller variants, they achieve better accuracy than the larger variants without the module, while also being less time-consuming and complex. For instance, SwinT-Small-GPH (61.7M parameters) outperforms SwinT-Big (87M parameters), and ConvNeXtBase-GPH (103.4M parameters) surpasses ConvNeXtLarge (197.9M). This partly demonstrates the effectiveness of the proposed module when integrated into different backbones. Regarding neural network complexity, despite a significant increase in the number of parameters in the

[2]According to the comparison table in https://paperswithcode.com/sota/fine-grained-image-classification-on-stanford-1 on 26/09/2023.

TABLE III: Evaluation results on the three datasets employing two distinct data sampling techniques during validation, namely Sequential and Shuffle.

Method	Stanford Dogs		CUB-200-2011		NABirds	
	Sequential	Shuffle	Sequential	Shuffle	Sequential	Shuffle
Densenet161-GPH	88.47	88.27 ± 0.10	84.79	84.63 ± 0.10	84.75	84.74 ± 0.12
SwinT-Big-GPH	93.06	92.97 ± 0.15	87.90	87.86 ± 0.20	87.38	87.37 ± 0.16

proposed models compared to the base ones, the inference time varies only slightly between them.

- It is worth noting that our observations reveal that the inclusion of GPH in all smaller model versions outperforms their larger counterparts in terms of inference time. A specific example is the DenseNet161 model, which has a faster processing time (0.29 compared to 0.42) despite having nearly three times the number of parameters and simultaneously achieving a superior result by 3.26%. This comparison is fairer than solely considering the number of parameters, especially for compact models like MobileNet and DenseNet, which exhibit a remarkable speed improvement of up to twice as fast.

In summary, our proposed approach consistently demonstrates enhanced performance across various classifiers and fine-grained datasets. Moreover, our method can easily integrate with cutting-edge classifiers to yield further enhancements. Notably, the parameter configuration for our approach remains uncomplicated, delivering favorable outcomes with a single setup across diverse classifiers and datasets.

B. The Impact of Batch Configurations (Q2)

In both the training and inference phases of the proposed module, the feature learning process of the GNN encoder begins by constructing a complete graph based on the features of the DNN encoder within a batch. Therefore, batch configurations, including batch size and how images are selected, influence the model's performance to some extent. In this experiment, we examine the stability of GPH under different batch configurations.

1) Batch size: Figure 3 reveals that altering the batch size of the testing process has minimal impact on the accuracy of GPH. Note that in this experiment, we only compare the results of 4 out of the 9 GPH variants for ease of illustration, but other variants exhibit similar trends. The results plotted on both datasets demonstrate that larger models tend to exhibit higher stability, i.e., changes in batch size do not significantly affect performance. From the figure, one can observe that MobilenetV3-S-GPH exhibits the biggest variability. In contrast, the other 3 variants show differences of less than 1%. It is worth highlighting that despite reducing the batch size to 1, these models are able to maintain their accuracy due to their training methodology, which fully utilizes the available RAM resources.

2) Shuffling the validation dataset during evaluation: Since GPH refines image latent embeddings using a fully connected graph of all embeddings within a batch, its performance may depend on the variation of the samples in the batch. In this section, we examine the stability of GPH under different batch configs of the evaluation datasets. Table III displays the

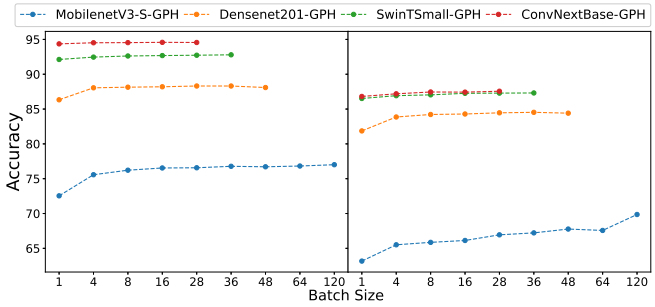


Fig. 3: Performance comparison for GPHs using various batch sizes on both the Stanford Dogs dataset (on the left) and the CUB-200-2011 dataset (on the right). Note that experiments with large batch sizes on Densenet201-GPH, SwinT-Small-GPH, and ConvNeXtBase-GPH are omitted due to the GPU's memory constraints.

comparison results of Densenet161-GPH and SwinT-Big-GPH models on the validation dataset with two different orders: sequential and shuffled-data sampling. In the sequential data sampling scenario, data is drawn from one class before moving on to the next class when filling the batches, making the variation of samples within each batch low. In contrast, in the common shuffled-data sampling, the variation within each batch is high since each sample is randomly picked from any class. As reported in Table III, sequential sampling provides slightly better accuracy, but the gap is small (maximum 0.3%). Therefore, we can confirm that GPH provides a pretty stable result, and the diversity of classes within the same batch has a minor impact on the model's classification performance. Note that we evaluate our model after training with shuffled data orders.

3) Feature selection within a batch during evaluation and prediction: The question at hand is whether, with pre-trained weights obtained during the training of the GPH model and a batch size of b, the model's input during testing or inference must necessarily be fixed with b images for the GNN encoder to process. To address this question, we employ a method of filling the batch embedding with vectors of all ones. Specifically, assuming we have $b_t < b$ images for testing, b_t images first pass through the DNN encoder to extract features $\{z_i\}_{i=1}^{b_1}$. Then, the $\{z_j\}_{j=b_1}^{b}$ are initialized as vectors of ones, and the entire set of b features is subsequently input into the GNN encoder for processing, as described in Section III-B. Table IV presents the evaluation results on the validation set using this method with $b_1 = 1$, corresponding to different values of b. The results demonstrate the stability of the batch size-specific filling method, even with MobileNetV3-S-GPH, where this method achieves better accuracy than the conventional approach of taking the entire batch of images. From

TABLE IV: The performance of models using various batch sizes after filling batch feature embeddings with ones tensors on the Stanford Dogs dataset.

	1	4	8	16	28	36	48	64	120
MobilenetV3-S-GPH	72.54	75.57	76.22	76.54	76.57	76.78	76.71	76.82	77.01
MobilenetV3-S-GPH-filled	76.20	76.33	76.52	76.64	76.93	76.98	76.94	76.92	77.01
Densenet201-GPH	86.33	88.05	88.14	88.20	88.31	88.31	88.09	_	_
Densenet201-GPH-filled	87.25	87.47	87.68	88.00	88.27	88.27	88.09	_	_

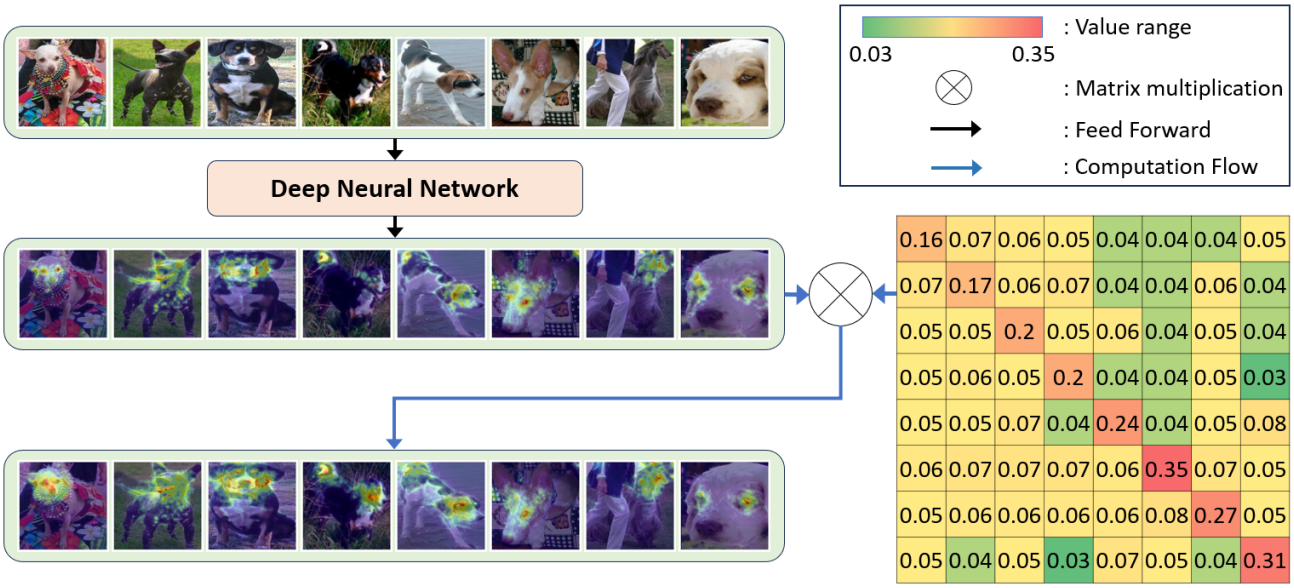


Fig. 4: The flow chart illustrates the GradCAM visualizations of features extracted by ConvNeXt-Large-GPH within a batch containing 8 images.

these results, it is evident that the filling method effectively addresses the posed question.

TABLE V: The maximum batch size for each model.

Full RAM	
MODEL	BATCH SIZE
MobilenetV3	120
Densenet	48
SwinTransformer	36
ConvNext	28
HERB-SwinT	5

4) The accuracies of GPHs on various batch sizes and datasets: In Table VIII, detailed results are presented for GPH models across three datasets: Stanford Dogs, CUB-200-2011, and NABirds, considering various batch sizes. The experiments encompass batch sizes ranging from 1 to 120, and it's noteworthy that the maximum batch size for each model is explicitly outlined in Table V, providing insights into the configurations concerning an NVIDIA Tesla T4 GPU with 15GB of VRAM. These results further build upon the observations from Section V-B, specifically addressing the impact of batch configurations on model performance. Notable observations are listed as follows:

- It becomes apparent that as the batch size decreases, there is a consistent trend of accuracy reduction across all three datasets. Notably, MobilenetV3-S-GPH shows the most significant performance gap when the batch size is reduced to 1. The observed gap amounts to 4.47%, 6.69%, and 9.38% in Stanford Dogs, CUB-200-2011, and NABirds datasets, respectively. This suggests that MobilenetV3-S-GPH is more sensitive to smaller batch sizes compared to other models.
- Additionally, in the Stanford Dogs dataset, ConvNextBase-GPH outperforms SwinT-Small-GPH, demonstrating superiority in accuracy. However, when examining the CUB-200-2011 and NABirds datasets, both models exhibit similar levels of accuracy, indicating a dataset-specific variation in performance.
- The performance tables also reveal interesting trends in the behavior of models across different datasets and batch sizes. For instance, SwinT-Small-GPH consistently maintains high accuracy across various batch sizes in all three datasets, showcasing its robustness in handling different batch configurations.

In summary, the analysis of model performance across different batch sizes provides valuable insights into the sensitivity of GPH models to batch variations. The observed trends contribute to our understanding of the practical implications of batch size selection in real-world scenarios, offering guidance on optimizing model performance for specific datasets and computational constraints.

C. Visual Analysis (Q3)

1) Visualization of Feature Extracted by Conventional DNNs and GPHs: Recall that, in our proposed GPH architecture, we train the DNN and GNN components simultaneously

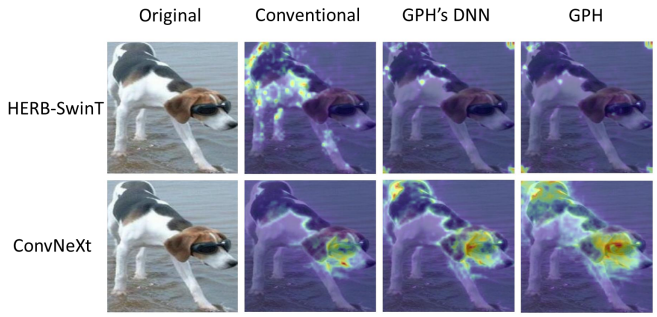


Fig. 5: Comparison between features extracted by ConvNeXt-Large, ConvNeXt-Large-GPH, HERB-SwinT and HERB-SwinT-GPH on Stanford Dogs dataset, illustrated by GradCam

within an end-to-end framework, enabling direct influence of GNN backpropagation on DNN parameters during training. As a result, features generated by a conventional DNN backbone exhibit distinct characteristics compared to those produced by the DNN component in the GPH method. Specifically, the GNN tends to propagate gradients backward, allowing the DNN to discern subtler features for combinatorial purposes and create intricate, rich feature maps. Conversely, traditional DNN parameters are propagated to emphasize the most critical feature, assisting in the classification of images into distinct classes.

GradCAM [39] visualizations in Figure 5 highlight aspects of the aforementioned argument. Here, we employ ConvNeXt-Large, HERB-SwinT as the DNN backbone and GraphTransformer for the GNN. The top row displays the GradCAM visualizations of HERB-SwinT, while the bottom row showcases the visualizations of ConvNeXt. The columns, from left to right, represent the feature from a conventional method, the feature extracted by GPH's DNN, and the combined feature of both DNN and GNN, respectively. On one hand, in ConvNeXt, the feature map extracted by the traditional method primarily highlights the dog's head, with relatively less emphasis on the colored regions of the dog's body. On the other hand, employing the GPH method's DNN yields a feature map that assigns significance to multiple subtle positions, encompassing the head, colored regions on the dog's body, and even the tail. In contrast, when considering Attention-based models such as HERB-SwinT, the GPH method deviates from the conventional approach of focusing on large image regions. Instead, it aims to eliminate redundant features and concentrate solely on the specific subtle features that significantly contribute to fine-grained image classification tasks. Figure 8 showcases a comprehensive comparison of features visualized by GradCAM, encompassing both conventional models and the GPH one. The comparison includes various datasets, such as Stanford Dogs and CUB-200-2011, as well as different model backbones like HERB-SwinT, P2P-Net, and ConvNeXt-Large.

For a more comprehensive understanding of the features extracted by DNN backbone and the combined feature of both DNN and GNN when following the GPH architecture, Figure 4 presents a flow chart showcasing the GradCAM visualizations of features extracted by ConvNeXt-Large-GPH. In the initial step, raw images are passed through the DNN backbone of the ConvNeXt-Large-GPH model, resulting in the extraction of features. These extracted features are then obtained as the output of GPH's DNN. Subsequently, the extracted features from GPH's DNN are organized into a fully connected graph using the GNN module. During training, the GNN module learns the edges of this graph. In the figure, the visualization of this graph is represented as the GNN adjacent matrix. Within this matrix, each cell represents the connection between two images corresponding to a specific row and column. The row order aligns from top to bottom, mirroring the image order in the figure. Similarly, the column order matches the row order but progresses from left to right. Following the computation of the adjacent matrix, the GNN combines the features extracted by GPH's DNN using a matrix multiplication operation. This operation involves multiplying the transpose of the GPH's DNN features matrix with the GNN adjacent matrix. The final outputs consist of the transpose of the GPH combined features matrix, which possesses the same shape as the transpose of the GPH's DNN features matrix.

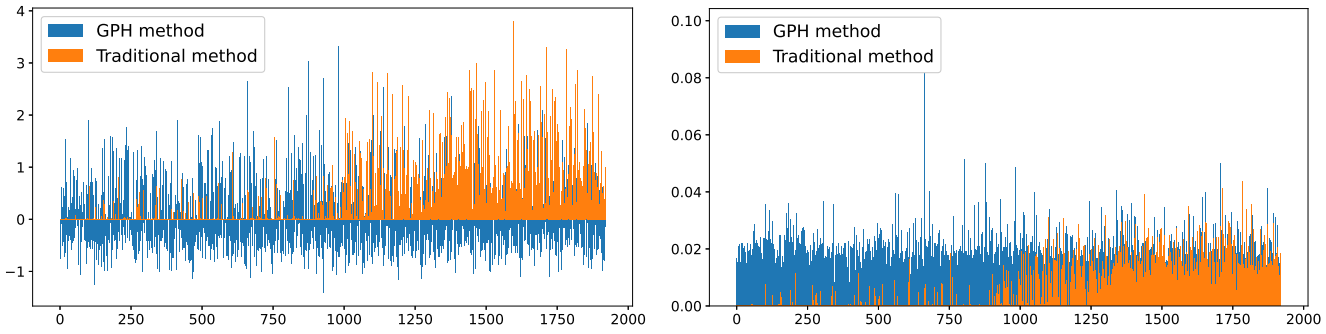
Furthermore, our investigation delves deeper into the analysis of numerical values within feature vectors when images within the same batch share a common label. The exploration involves an examination of the x-axis positions in the 1920-dimensional feature vector, where Figure 6a and 6b illustrate these positions, with the corresponding values represented on the y-axis.

In this particular scenario, the features under consideration are the mean and standard deviation of the feature vector within a single-label batch. Notably, the conventional method predominantly emphasizes the right side of the feature vector, indicating a concentration of attention on specific positions. In contrast, when employing GPH, attention is distributed across scattered positions along the feature vectors. This nuanced exploration and integration of diverse features are indicative of the GNN capacity to incorporate information in a more holistic and interconnected manner.

This analysis further underscores the value of GPH in capturing and utilizing a broader range of features, potentially contributing to the observed performance improvements in image classification tasks. By attending to scattered positions in the feature vectors, GPH showcases its ability to leverage relational information and contextual dependencies, allowing for a more robust and nuanced understanding of the underlying data. This not only strengthens the theoretical foundation of GPH but also provides practical implications for its application in improving the interpretability and performance of deep neural networks in image processing tasks.

2) Visualization of gradient backpropagation between Attention-based models and CNN-based GPH models: In this subsection, we aim to uncover the underlying reasons behind the Attention-based GPH model's tendency to focus on fewer subtle features, while the CNN-based GPH model tends to focus on multiple subtle features. To this end, we conduct experiments by visualizing the heatmap of the gradient backpropagation during the backward step.

Figure 7 depicts the heatmap visualization of gradient backpropagation between conventional and GPH models uti-



(a) (b)

Fig. 6: Feature vector visualization for 50 images in a single-label batch. a) Mean feature vector, b) Standard deviation feature vector

lizing HERB-SwinT and ConvNeXt backbones. The rows are organized in the following sequence from top to bottom: raw image, gradient heatmap visualization of conventional models, and finally, gradient heatmap visualization of GPH models. The left column displays the gradient heatmap visualizations of HERB-SwinT, while the right column showcases the visualizations of ConvNeXt. Our interesting findings are listed as follows:

- It is evident that the gradient backpropagation of the traditional ConvNeXt model focuses on a larger region of the image, whereas the gradient backpropagation of the conventional HERB-SwinT model only concentrates on a few pixels of the image. This phenomenon can be attributed to the structural differences between Attention-based and CNN-based backbones. Attention-based models flatten the image to pixels and apply Attention mechanisms to process them, whereas CNN-based models utilize spatial kernels to extract spatial features from the image. As a reminder, in our proposed GPH architecture, the GNN solely transfers its influence to the DNN through gradient backpropagation during training. Hence, the gradient backpropagation of the ConNeXt-GPH model is adept at uncovering a greater number of subtle features and integrating them into complex features.
- Conversely, the gradient backpropagation of the HERB-SwinT-GPH model primarily focuses on pixel-level features and identifies fewer features compared to ConNeXt-GPH, although it still surpasses the gradient backpropagation of the conventional HERB-SwinT-GPH model in terms of the number of features. This is one of the reasons why Attention-based GPH models tend to enhance their focus on subtle features, while CNN-based models improve their ability to discover more subtle features. Consequently, CNN-based models exhibit greater accuracy improvements than Attention-based models, as mentioned in Section V-A.

D. Investigating the Impact of Different GNN Encoders and Attention Mechanism on Model Performance (Q4)

1) Different GNN Encoders: To investigate the effect of employing different GNN models as the GNN encoder, we perform an experiment using four popular GNN methods

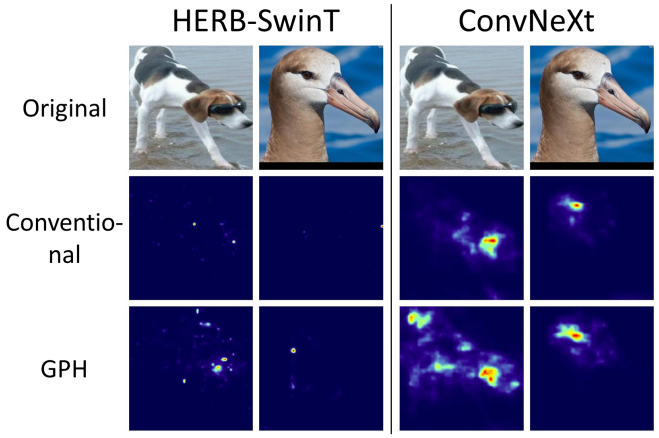


Fig. 7: The heatmap visualization of gradient backpropagation between conventional and GPH models utilizing HERB-SwinT and ConvNeXt backbones

and Attention [40] mechanism using the mean aggregation function, including GCN [26], GAT [41], GraphSAGE [37], and GraphTransformer [36], by assessing their performance on the three benchmark datasets while using Densenet201 as the underlying DNN backbone.

Table VI demonstrates that models equipped with these additional modules consistently enhance accuracy compared to the standard Densenet201. Notably, our five post hoc models show more significant improvements, particularly in the realm of fine-grained classification across these three datasets. However, the enhancements of the four models with the GNN plugin still surpass those with the Attention plugin. Specifically, on the CUB and NAbirds datasets, the improvements with Attention are marginal, only 0.32% and 1.04% respectively. Meanwhile, the other methods exhibit improvements averaging 5.35% and 6.26%. This also partly indicates the superior capabilities of GNNs over attention mechanisms.

Moreover, when comparing the four GNN plugin models with each other, there is no specific post hoc model that outperforms the others on all three datasets. Specifically, the GraphTransformer architecture demonstrates superior performance on the StanfordDogs dataset, whereas the GCN architecture excels on the NAbirds dataset. Nonetheless, in pragmatic applications, there may be a preference for adopting

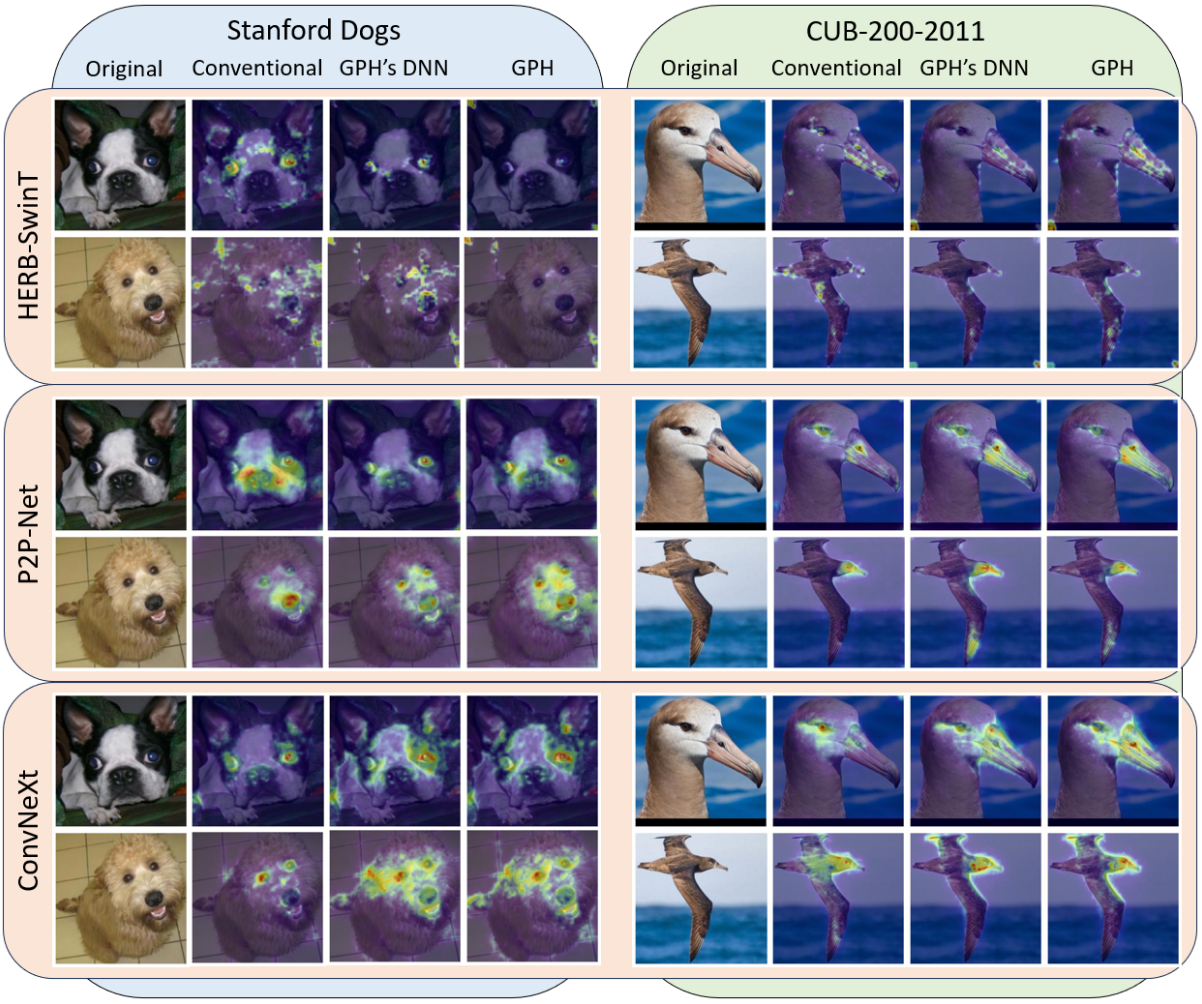


Fig. 8: Feature visualization comparison using GradCAM: Conventional models and the GPH model, diverse datasets (Stanford Dogs, CUB-200-2011), and various backbone architectures (HERB-SwinT, P2P-Net, ConvNeXt-Large).

the GraphTransformer as the default architecture, given its consistent and satisfactory results observed across all three datasets.

TABLE VI: Model accuracy according to different GNN encoders.

Model	Acc (%)		
	Stanford Dogs	CUB-200-2011	NAbirds
Densenet201	83.95	79.13	77.55
Densenet201-Attention-Mean-Aggr	85.28	79.45	78.59
Densenet201-GCN	87.6	84.40	**84.14**
Densenet201-GAT	87.82	**84.61**	83.94
Densenet201-SAGE	87.39	84.43	83.54
Densenet201-GraphTransformer	**88.09**	84.48	83.62

2) Attention Mechanism and GNNs: In this subsection, we investigate the difference between employing Attention mechanism and GNNs. Attention can be considered as a specific instance of GNN, making it a viable replacement for GNN using equations 1 and 2. In these formulas, the aggregate and combined functions involve matrix multiplication between attention weights and batch feature vectors.

Table VII encompasses the analysis of four models, corresponding to each variant to GNN component in the GPH: GNN, Attention (regular Attention layer, utilizing the sum aggregation function), Attention-MeanAggr (employing the mean aggregation function). We inspect the value ranges for both features derived from the DNN and Attention/GNN components in the GPH model. It is evident that using sum aggregation leads to an excessively large value range for the Attention feature, introducing noise into the DNN backbone feature when applying Equation 3. Employing mean aggregation, on the other hand, narrows the value range and yields superior accuracy in the case of a full RAM batch. However, both methods experience a significant drop in accuracy as the batch size decreases. With the proposed enhancement, the GNN demonstrates a more favorable value range for both feature components, resulting in a marginally superior accuracy.

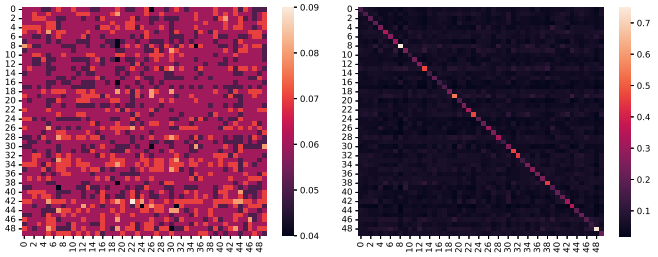
E. GNN Adjacent matrix (Q5)

A noteworthy advantage of representing features in a graph format lies in the ability to establish connections among images within a batch. In this context, we explore two scenarios: GNN with and without edge attributes in Equation 1. More specifically, the computation of the edge weight e_{ij} between two images involves calculating the L1-norm distance between their respective pixel values. The experimental result is visu-

TABLE VII: Feature value range and accuracies of models trained with full RAM batch size and batch size of 1 with Densenet201 backbone in Stanford Dogs dataset.

Model	Feature value range		Acc(%)	Acc (%) (batchsize=1)
	DNN	Attention/GNN		
Attention	[-3.147, 2.877]	[-1.756, 2.020]	24.46	31.42
Attention-MeanAggr	[-3.147, 2.877]	[-0.032, 0.041]	85.47	31.42
GNN	[-1.423, 1.190]	[-0.420, 0.405]	88.09	86.33

alized in Figure 9, where the x-axis and y-axis enumerate the image indices within a batch, and each cell corresponding to positions on the x and y axes signifies the connection between the two images. The color intensity mirrors the strength of relationships between images, with brighter hues denoting stronger connections and darker shades indicating weaker ones. The matrix observed from Figure 9a that, without edge attributes, lacks intuitiveness, whereas the matrix generated by GNN with edge attributes in Figure 9b is more easily comprehensible. This is evident from the prominent values along the diagonal and the symmetrical appearance of the matrix across the main diagonal. There are two benefits to having diagonally symmetrical elements and larger diagonal elements in the adjacency matrix. Firstly, it enhances the understanding and intuitiveness of how graphs synthesize images in batches. Secondly, it leads to a reduced loss of original image information, resulting in a slight improvement in accuracy.



(a) (b)

Fig. 9: The heatmap of the adjacent matrix representing the relationship between images in a batch generated by Multihead GraphTransformer. (a) without edge attribute, (b) with edge attribute.

F. The accuracies of GPHs on general image classification (Q6).

The outcomes depicted in Table IX underscore the improvements introduced by GPHs when integrated into conventional models for general image classification tasks. Notably, the results are drawn from well-established datasets such as CIFAR-100 [42] and Tiny-Imagenet [43], widely recognized benchmarks in the field.

One notable observation is the consistent performance improvement across various model architectures, such as MobilenetV3, Densenet, SwinT, ConvNextBase, and ConvNext-Large when equipped with GPHs. This uniform enhancement suggests that the effectiveness of GPHs extends beyond specific network designs and problem domains and serves as a versatile augmentation for diverse model backbones.

Typically, for conventional CNN models in classification tasks, increasing the number of parameters in either the width or depth of the network may yield marginal improvements. Adding GPH modules post hoc to existing backbones results in a new model with more parameters compared to the corresponding DNN classifier. The results in the table show a consistent level of improvement for the proposed models, although these improvements are not as significant as observed in fine-grained image classification tasks. In particular, our proposed approach achieves new state-of-the-art results on the Tiny-Imagenet[3] dataset. This indicates that the enhancement is not solely due to an increase in the number of parameters but also reflects the effective functioning of the model architecture.

VI. Conclusion

We have represented a novel network architecture that appears deceptively straightforward yet has remained unexplored in prior studies. Rigorous experimentation conducted on

[3]https://paperswithcode.com/sota/image-classification-on-tiny-imagenet-1

TABLE VIII: The performance of GPHs on various batch sizes across the Stanford Dogs, CUB-200-2011, and NABirds datasets.

Dataset	Method	1	4	8	16	28	36	48	64	120
Stanford Dogs	MobilenetV3-S-GPH	72.54	75.57	76.22	76.54	76.57	76.78	76.71	76.82	77.01
	Densenet201-GPH	86.33	88.05	88.14	88.2	88.31	88.31	88.09	_	_
	SwinT-Small-GPH	92.12	92.46	92.62	92.68	92.73	92.79	_	_	_
	ConvNextBase-GPH	94.36	94.52	94.54	94.58	94.56	_	_	_	_
CUB-200-2011	MobilenetV3-S-GPH	63.17	65.51	65.86	66.13	66.94	67.22	67.77	67.57	69.86
	Densenet201-GPH	81.86	83.86	84.22	84.29	84.46	84.53	84.41	_	_
	SwinT-Small-GPH	86.52	86.92	87.04	87.26	87.29	87.31	_	_	_
	ConvNextBase-GPH	86.81	87.19	87.45	87.43	87.56	_	_	_	_
NABirds	MobilenetV3-S-GPH	59.72	66.05	67.27	68.05	68.4	68.64	68.77	68.83	69.1
	Densenet201-GPH	73.96	80.23	82.14	83.09	83.47	83.56	83.62	_	_
	SwinT-Small-GPH	85.13	86.43	87.07	87.64	87.88	87.97	_	_	_
	ConvNextBase-GPH	85.72	87.33	87.57	87.78	87.86	_	_	_	_

TABLE IX: The impact of GPHs on conventional classification outcomes when incorporated into various DNN techniques

Method	Acc (%)	
	CIFAR-100	Tiny-Imagenet
MobilenetV3-S	77.97	71.66
MobilenetV3-S-RBI	79.44	75.84
MobilenetV3-L	79.88	76.84
MobilenetV3-L-RBI	80.01	77.87
Densenet201	81.12	78.51
Densenet201-RBI	83.78	84.22
Densenet161	82.43	78.85
Densenet161-RBI	84.58	84.56
SwinT-Small	88.37	88.2
SwinT-Small-RBI	89.2	89.65
SwinT-Big	88.75	89.68
SwinT-Big-RBI	89.3	90.12
ConvNextBase	88.06	92.21
ConvNextBase-RBI	89.74	92.72
ConvNextLarge	88.12	93.18
ConvNextLarge-RBI	**89.98**	**93.71**

benchmark datasets underscores the efficacy of our proposed approach, showcasing its seamless integration with a variety of fine-grained classifiers. These synergistic interactions yielded appreciable improvements in accuracy, establishing a new benchmark for performance in the field. Additionally, our architectural innovation fostered a reduction in both model parameters and inference latency when compared to conventional DNN methodologies.

Our research opens up several promising avenues for future exploration. First, further investigation can delve into optimizing the architecture and hyperparameters of the integrated GNN-DNN model for different fine-grained classification tasks. Additionally, exploring different graph construction strategies and graph neural network architectures may yield insights into improving model performance. Moreover, the application of this integrated approach to other computer vision tasks and datasets warrants exploration, as it has the potential to enhance various aspects of visual recognition.

REFERENCES

[1] F. Lu, W. Li, C. Li, S. Liu, D. Wu, M. Fang, X. Zou, M. Li, R. Zheng, Y. Ren, X. Liao, H. Jin, and A. Y. Zomaya, "Fine-grained lesion classification framework for early auxiliary diagnosis," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pp. 1–13, mar 2023.

[2] C. Zhang, J. He, and L. Shang, "An x-ray image classification method with fine-grained features for explainable diagnosis of pneumoconiosis," *Personal and Ubiquitous Computing*, pp. 1–13, 06 2023.

[3] S. Wen, Y. Chen, S. Guo, Y. Ma, Y. Gu, and P. Chan, "Discriminative domain adaptation network for fine-grained disease severity classification," in *2023 International Joint Conference on Neural Networks (IJCNN)*, 2023, pp. 1–8.

[4] G. V. Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie, "The inaturalist species classification and detection dataset,," in *CVPR*, 2017, p. 8769–8778.

[5] G. V. Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. I. andS. Belongie, and P. Perona, "Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2015, pp. 595–604. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298658

[6] G. V. Horn, E. Cole, S. Beery, K. Wilber, S. Belongie, and O. M. Aodha, "Benchmarking representation learning for natural world image collections,," in *CVPR*, 2015.

[7] X. He and Y. Peng, "Fine-grained image classification via combining vision and language," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jul. 2017. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2017.775

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[12] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9992–10 002.

[13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017. [Online]. Available: http://arxiv.org/abs/1704.04861

[14] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261–2269.

[15] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[16] Y. Pan, Y. Xia, and D. Shen, "Foreground fisher vector: Encoding class-relevant foreground to improve image classification," *IEEE Transactions on Image Processing*, vol. 28, no. 10, pp. 4716–4729, 2019.

[17] X. Yang, Y. Wang, K. Chen, Y. Xu, and Y. Tian, "Fine-grained object classification via self-supervised pose alignment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7399–7408.

[18] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," 2014.

[19] X. Liu, T. Xia, J. Wang, Y. Yang, F. Zhou, and Y. Lin, "Fully convolutional attention networks for fine-grained recognition," 2017.

[20] S. Kim, J. Nam, and B. C. Ko, "ViT-NeT: Interpretable vision transformers with neural tree decoder," in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 11 162–11 172. [Online]. Available: https://proceedings.mlr.press/v162/kim22g.html

[21] P.-Y. Chou, Y.-Y. Kao, and C.-H. Lin, "Fine-grained visual classification with high-temperature refinement and background suppression," 2023.

[22] J. He, J.-N. Chen, S. Liu, A. Kortylewski, C. Yang, Y. Bai, and C. Wang, "Transfg: A transformer architecture for fine-grained recognition," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, no. 1, 2022, pp. 852–860.

[23] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei, "Novel dataset for fine-grained image categorization," in *First Workshop on Fine-Grained Visual Categorization (FGVC)*, 2011.

[24] G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie, "Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 595–604.

[25] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2014.

[26] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2017.

[27] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks,," in *in Proc. of AAAI*, 2018, p. 3546–3553.

[28] C. Zhuang and Q. Ma, "Dual graph convolutional networks for graph-based semi-supervised classification," in *Proceedings of the 2018 World Wide Web Conference*, ser. WWW '18. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2018, p. 499–508. [Online]. Available: https://doi.org/10.1145/3178876.3186116

[29] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *International Conference*

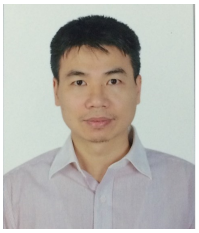
on Learning Representations, 2019. [Online]. Available: https://openreview.net/forum?id=rklz9iAcKQ
[30] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.
[31] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 257–266. [Online]. Available: https://doi.org/10.1145/3292500.3330925
[32] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016.
[33] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, jul 2018. [Online]. Available: https://doi.org/10.24963%2Fijcai.2018%2F505
[34] X. Wang, W. Zhang, C. Wang, Y. Gao, and M. Liu, "Dynamic dense graph convolutional network for skeleton-based human motion prediction," *IEEE Transactions on Image Processing*, vol. 33, pp. 1–15, 2024.
[35] V. Garcia and J. Bruna, "Few-shot learning with graph neural networks," *arXiv preprint arXiv:1711.04043*, 2017.
[36] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/9d63484abb477c97640154d40595a3bb-Paper.pdf
[37] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 1024–1034. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html
[38] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie., "The caltech-ucsd birds-200-2011 dataset, computation & neural systems, technical report, cns-tr, california institute of technology, usa." 2011.
[39] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, oct 2019. [Online]. Available: https://doi.org/10.1007%2Fs11263-019-01228-7
[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
[41] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," *6th International Conference on Learning Representations*, 2017.
[42] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.
[43] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," *CS 231N*, vol. 7, no. 7, p. 3, 2015.

Duy Minh Le is an undergraduate student in the Faculty of Information Technology at the Post & Telecommunication Institute of Technology in Hanoi, Vietnam. His research interests include machine learning, Neuro-Symbolic AI and reinforcement learning.

Bao Bui-Quoc was born in Thai Binh, Viet Nam in 1998. He received a B.E. degree in applied mathematics and informatics from Hanoi University of Science and Technology, Ha Noi, in 2021. He is currently a Research Assistant at the College of Engineering and Computer Science, VinUniversity. His research interests include computer vision, image processing, and machine learning/reinforcement learning.

Anh Tran is a senior research scientist at VinAI Research. Before that, he worked at Amazon AWS Rekognition for 1.5 years. He earned a Ph.D. degree from the Computer Science Department of the University of Southern California in 2017 under the supervision of Professor Gerard Medioni. His research interest is Computer Vision, and his research topics include Generative AI, Facial Image Processing, and AI Security. He has had around 30 papers accepted to top conferences in Computer Vision and Machine Learning. He has served as an Area Chair at nearly ten top-tier AI conferences and organized two workshops at NeurIPS23 and CVPR24.

Cong Tran received his doctoral degree in computer science from Dankook University, Yongin, Republic of Korea, in 2021. He previously received his M.Sc. in computer science in 2014 and his B.Sc. in network and communication in 2009 from Vietnam National University, Hanoi, Vietnam. Since September 2021, he has been with the Faculty of Information Technology, Posts & Telecommunication Institute of Technology, Hanoi, Vietnam, as a lecturer. His research interests include social network analysis, data mining, and machine learning.



Cuong Pham received a PhD in Computer Science at Newcastle University in 2012. He is an Associate Professor of Computer Science and the Director of NAVER AI center at Posts and Telecommunications Institute of Technology. He is also a Visiting Research Scientist at VinAI Research. His research interests include machine learning/deep learning, ubiquitous computing, wearable computing, computer vision, human activity recognition, human computer interaction, and pervasive healthcare.