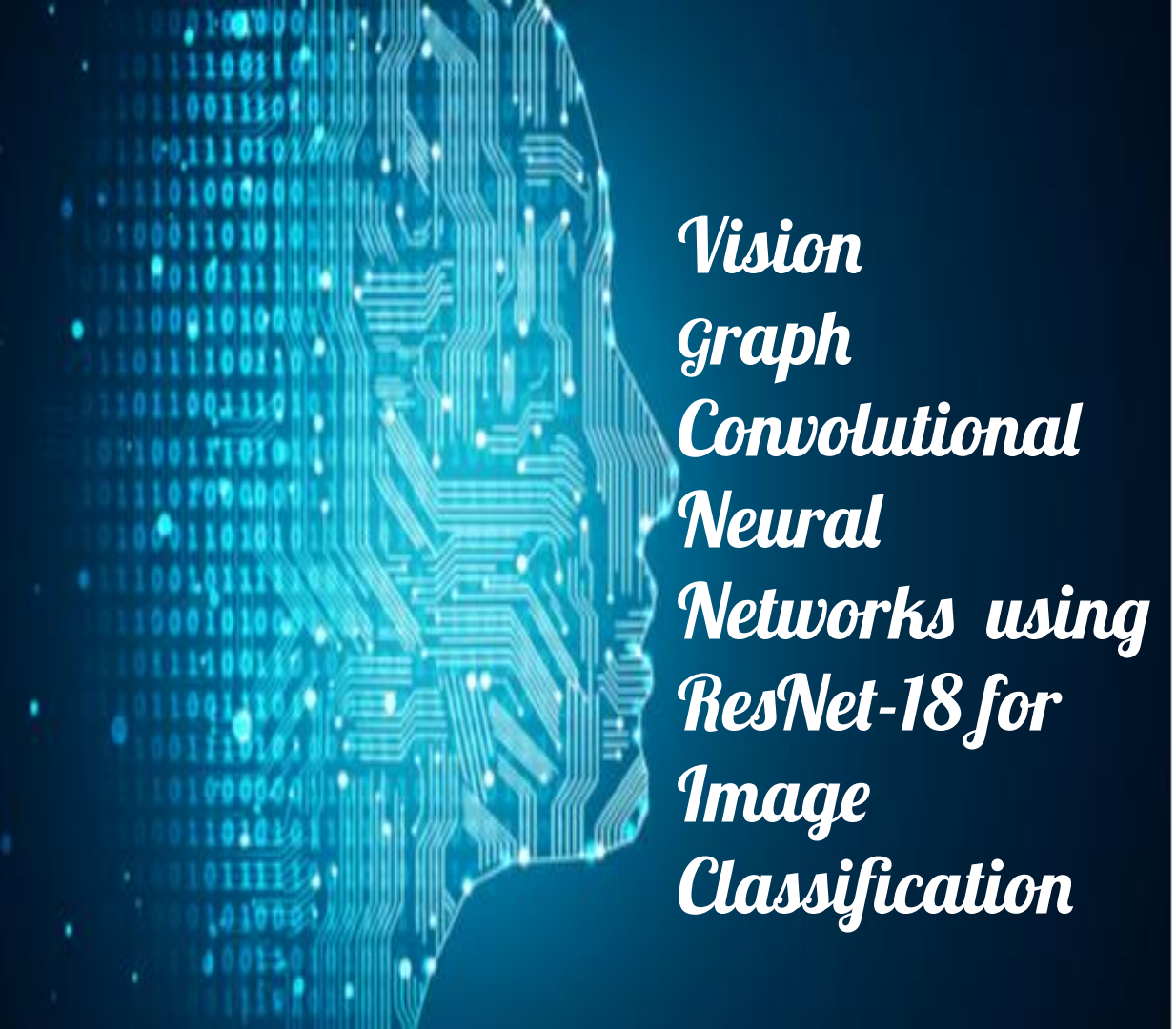

Vision Graph Convolutional Neural Networks using ResNet-18 for Image Classification

Department of Computer Science and Engineering
United International University

Pattern Recognition Lab
CSI(416)

Submitted To:
Dr. Md. Saddam Hossain Mukta
Associate Professor
United International University

Submitted By:
Faria Haque Marvi
ID- 011 181 100

May 14th, 2023

Table of contents

Introduction

The crucial task of image classification in computer vision involves categorizing an image into predefined categories. Different models have been developed to increase the accuracy and efficiency of picture classification. Deep learning models have demonstrated amazing performance in image classification tasks.
In this report, I use a combination of ResNet18(Residual Neural Network) and Vision Graph Neural Network (VGNN) models for image classification. ResNet18 is a widely used convolutional neural network architecture that effectively learns and extracts meaningful features from images. VGNN is a recent model that uses graph neural networks to capture the relationships between objects in an image and is particularly suitable for tasks that require understanding the semantic structure of an image.

1.1 Project Overview

The goal of this project is to investigate the effectiveness of using a combination of ResNet18 and Vision Graph Neural Network (VGNN) models for image classification. I implemented a VGNN model with a pre-trained ResNet-18 backbone and graph convolutional layers to capture the relationships between the extracted features from ResNet-18.
The project involved the following steps:
1. Data collection: I collected a standard image classification benchmark dataset for evaluating the performance of the implemented model.

2. Data preprocessing: Preprocessed the data by resizing and normalizing the images to prepare them for training and testing.
3. Model implementation: Implemented a VGNN model with a pre-trained ResNet-18 backbone and graph convolutional layers to classify the images into predefined classes or categories.
4. Model training and evaluation: Trained the implemented model on the dataset and evaluated its performance in terms of accuracy and computational efficiency.
5. Result analysis: Analyzed the results to determine the strengths and weaknesses of the implemented model and identify any potential improvements.

1.2 Motivation

This report aims to explore the potential benefits of combining different neural network architectures for image classification tasks. While deep convolutional neural networks such as ResNet18 have achieved impressive results in image classification, there is still room for improvement in terms of accuracy, speed, and robustness. By combining ResNet18 with a Vision Graph Neural Network (VGNN), we can potentially leverage the strengths of both architectures and develop a more effective and efficient image classification solution. The motivation for this report is to investigate whether this approach can lead to improved classification performance and to provide insights and recommendations for future research in this area. Ultimately, the goal is to contribute to the development of more accurate and reliable image classification systems that can benefit a wide range of applications in various fields, from healthcare and autonomous driving to security and entertainment.

1.3 Objectives

This report aims to explore the use of a combination of ResNet18 and Vision Graph Neural Network (VGNN) models for image classification tasks. The primary goals of this project are to implement a VGNN model with a pre-trained ResNet-18 backbone and graph convolutional layers, train and evaluate the model on a standard image classification benchmark dataset, analyze the results to identify the strengths and weaknesses of the implemented model, and provide insights into the effectiveness of this approach for image classification tasks.

By achieving these objectives, this project aims to contribute to the advancement of research in computer vision and image classification and provide practical implications for practitioners in this field.

1.4 Methodology

The methodology involved in this project includes three main steps: (1) Preprocessing of the image dataset, including resizing and data augmentation, (2) Implementation of the ResNet18 and VGNN models for feature extraction and classification, and (3) Training and testing the models using the stochastic gradient descent optimizer and cross-entropy loss function. The models were trained and tested on a large dataset of images from 3 different classes. The performance of the models was evaluated based on accuracy. The results show that the combination of ResNet18 and VGNN models improves the accuracy of image classification compared to using each model individually.

2. Preliminaries

2.1 Dataset: A collection of images with associated labels that can be used for training and testing the classification model.

2.2 Data pre-processing: The images in the dataset must be pre-processed to ensure consistency in image size, color space, and pixel values and mean and standard deviation. Additionally, the data may need to be split into training, validation, and testing sets

2.2.1 Augmentation

2.2.1.1 ColorJitter: This transformation randomly changes the brightness, contrast, saturation, and hue of the input image to create variations of the same image.

2.2.1.2 RandomHorizontalFlip: This transformation randomly flips the input image horizontally with a probability of 0.5.

2.2.1.3 RandomRotation: This transformation randomly rotates the input image by a certain angle (specified in degrees) within a range of -10 to +10 degrees.

2.2.1.4 GaussianBlur:

This transformation applies a Gaussian blur filter to the input image with a kernel size of 3 and a random sigma value between 0.1 and 2.0. This helps to smooth out the image and reduce noise, which can improve the robustness of the model to small variations in the input data.

2.3 **Models**

2.3.1 ResNet18 model(Residual Neural Network): ResNet-18 is a deep convolutional neural network architecture that utilizes residual connections to enable training of very deep networks. This model must be pre-trained on a large dataset of images to extract meaningful features from images.

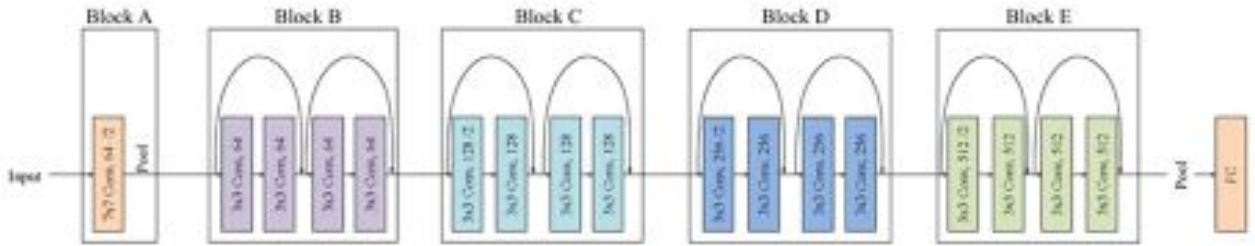

Figure1: Structure of Resnet-18

Advantages: using a pre-trained ResNet-18 model can be beneficial in the context of the VGNN model. Pre-trained models like ResNet-18 are trained on large-scale image datasets and have learned to extract high-level features from images that are relevant to many image recognition tasks. By using a pre-trained ResNet-18 model as a feature extractor, the VGNN model can leverage these learned features to construct a graph of pairwise similarities between the extracted features, which can then be used to make accurate predictions. This can lead to faster and more accurate training and better generalization to new images.

2.3.2 VGNN model (Vision Graph Neural Network): A Visual Graph Neural Network (VGNN) is a type of graph convolutional neural network that is designed to process visual data. It works by constructing a graph of pairwise similarities between features extracted from a pre-trained visual model, such as a convolutional neural network (CNN) like ResNet. The graph convolutional layers then operate on the adjacency matrix of the graph to extract global features, which are then fed into a fully connected layer to make predictions. The VGNN can be trained end-to-end on a visual dataset and can achieve state-of-the-art results on tasks such as image classification, object detection, and segmentation.

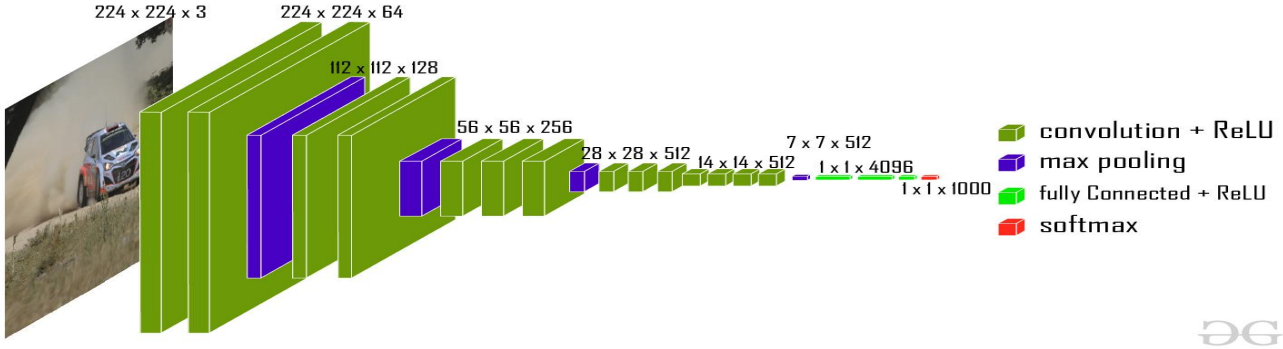

Figure 2: VGNN Model

The VGNN model must be designed with graph convolutional layers to capture the relationships between the extracted features from ResNet18

2.4. Training procedure: The model must be trained using the training set of images and their corresponding labels. The training process involves forward and backward propagation to adjust the model's parameters to minimize the loss function.

2.5. Evaluation: The trained model must be evaluated on a separate validation set of images to ensure that it generalizes well to unseen data. Additionally, the model's performance can be measured using metrics such as accuracy.

2.6. Testing: The final step is to test the model on a separate test set of images to assess its performance on unseen data. The results can be compared to state-of-the-art image classification models to determine the effectiveness of using a combination of ResNet18 and VGNN for image classification.

3. Project Design:

Functional requirements: Functional requirements are the specific features and capabilities that the image classification system must perform. These requirements describe the expected behavior of the system and are used to define the scope of the project. Some examples of functional requirements for an image classification system might include:

- The system ought to be able to categorize photos into several classifications.
- Both RGB and grayscale photos should be supported by the tool.
- Large databases of images should be manageable for the application.
- The architecture of the system is supposed to be able to handle images with different aspect ratios and sizes.

Non-functional requirements: Define the aspects of the system, however, that are crucial to its performance and usability but are not immediately related to its functionality. Non-functional requirements for an image categorization system could be, for instance:

- The program should be simple to use and have an intuitive interface.
- The software must be scalable and ready to manage growing data volumes.
- The program should be simple to use and have an intuitive interface.
- Also should have low latency and be able to process images quickly
- Should be secure and protect user data and privacy

For a successful development of an effective image categorization system that satisfies the expectations of its users, functional and non-functional requirements are equally crucial. These specifications aid in directing the design and development process as well as testing and performance evaluation of the system.

4. Implementation and Results

4.1 Environment Setup

To set up the environment, first of all Python needs to be installed.Used Google colab.Then pytorch, Also checks whether a GPU is available or not by using function. If a GPU is available, it sets the device to 'cuda', otherwise it sets it to 'cpu'.

4.2 Testing and Evaluation

The Visual Graph Neural Network (VGNN) model was trained using 13 nodes and 50 epochs on a dataset of 300 images, with the categories being bedroom, drawing room, and officeroom.

We preprocessed our dataset of 300 images by splitting it into a training set of 282 images and a test set of 18 images. I loaded the data using PyTorch's DataLoader module, which enabled us to load the data in 32 batches and apply data augmentation techniques.The VGNN model is a graph convolutional neural network that processes visual data by constructing a graph of pairwise similarities between features extracted from a pre-trained ResNet-18 model. The graph convolutional layers then operate on the adjacency matrix of the graph to extract global features, which are then fed into a fully connected layer to make predictions.
Including training and evaluation functions for the VGNN model using a specified number of epochs and a specified loss function (CrossEntropyLoss) and optimizer (Stochastic Gradient Descent with a learning rate of 0.01, momentum of 0.9, and weight decay of 0.003. After training, the model was evaluated on the test set and achieved an accuracy of 33%.

Saving the best model:
modifying the pre-trained Visual Graph Neural Network (VGNN) model by replacing its final fully connected layer with a new one that has 3 output classes to match the categories of the dataset (bedroom, drawingroom, and officeroom). Then, it loads the checkpoint weights of the VGNN model with the highest accuracy on the test dataset during training. Finally, it saves the modified VGNN model with the updated fully connected layer as a PyTorch model checkpoint (.pth file) to a specified location in Google Drive.It is useful for reusing the trained VGNN model in future experiments or for deployment in production environments. By saving the model checkpoint, we can avoid having to retrain the model from scratch every time we want to use it.

Conclusion

The accuracy of ResNet-18 and VGNN in combination for image classification is unfavorable,However, it provides a promising direction for future research in combining these two architectures for image classification tasks. One possible reason for the low accuracy could be the small size of the dataset used for training. In future work, using a larger dataset, fine-tuning the pre-trained ResNet-18 model, and adjusting the hyperparameters of the VGNN model could potentially improve the performance of the model. Overall, this project provides a starting point for investigating the effectiveness of combining ResNet-18 and VGNN for image classification tasks.