IMAGE CLASSIFICATION FEED FORWARDING NEURAL NETWORK

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI (Rajasthan)

(September 30th 2024)

By:

Samraddh Saxena: 2021A7PS1455P

Anshul Maheshwari: 2021B5A70900P

Nimish Sharma: 2021B5A71144P

TASK 1

Table of Contents

INTRODUCTION

The objective of this project is to classify images of flowers into one of 60 distinct categories using a feed-forward neural network. Image classification is a fundamental task in computer vision, where models are trained to recognize patterns in images and assign them to specific categories. It has broad applications, from medical imaging and security systems to automatic plant species identification.

In this case, the dataset consists of images of 60 different flower types. Each category includes images that vary significantly in terms of color, shape, and size, making this a challenging classification problem. The ability to distinguish between visually similar flowers based on their subtle differences is crucial for successful classification.

Although convolutional neural networks (CNNs) are often the standard choice for image classification tasks due to their ability to capture spatial relationships in images, this project posed a unique challenge: we were restricted from using CNNs. To work around this, we focused on traditional feature extraction techniques, ultimately selecting **Histogram of Oriented Gradients (HOG)** as the primary method for extracting key features from the images. HOG is effective at capturing the structure and gradient orientation of objects in an image, making it well-suited for this task.

We also experimented with **Scale-Invariant Feature Transform (SIFT)**, a popular method for detecting and describing local features in images. However, SIFT did not yield the desired performance, and HOG was more efficient for our purposes.

This report outlines the steps taken to preprocess the dataset, design the neural network, train the model using optimization techniques, and evaluate the performance on a validation set. Our approach emphasizes the use of HOG for feature extraction and a fully connected neural network for classification, along with techniques like batch normalization and dropout to improve model generalization.

DATASET OVERVIEW

For our flower classification project, we worked with a vibrant dataset that features 60 different types of flowers. Each type has 50 images in the training set, which gives us a decent amount of data to help the model learn the unique characteristics of each flower. Additionally, we included a validation set with 10 images per type to evaluate how well the model performs on new, unseen examples.

What makes this dataset particularly interesting is the variety of colors and shapes among the different flower types. This diversity not only makes the task more enjoyable but also adds complexity to the classification challenge. Each flower has its own distinctive traits,

and our goal is to create a model that can accurately recognize and differentiate between them.

However, we faced some challenges due to the relatively small size of the dataset. A limited amount of data can often lead to overfitting, where the model performs well on training data but struggles with new data. To combat this, we employed various image augmentation techniques to artificially expand our dataset, providing the model with more diverse examples to learn from.

PRE-PROCESSING

Data Augmentation

To enhance the robustness of our model and combat overfitting, we implemented several image augmentation techniques. Given that our dataset was relatively small, with only 50 training images per flower type, we needed to artificially expand our dataset to improve generalization. We employed the following five techniques:

- **Horizontal Flip**: This method involves flipping images from left to right. It helps the model learn to recognize flowers regardless of their horizontal orientation.
- **Vertical Flip**: By flipping the images upside down, we allow the model to identify flowers in both upright and inverted positions, increasing the variety of training examples.
- **Color Jitter**: This technique introduces slight variations in brightness, contrast, saturation, and hue. It ensures that the model isn't overly sensitive to lighting conditions, allowing it to focus more on the structural features of the flowers.
- **Random Crop**: This method randomly selects portions of the images to use as input. By simulating different perspectives and framing, it enriches the dataset with various scales and viewpoints.
- **Rotation**: Randomly rotating the images at different angles enables the model to learn from flowers that might appear at various orientations. This is particularly useful in real-world applications, where flowers may not always be perfectly aligned.

By employing these augmentation techniques, we significantly increased the effective size of our training dataset, providing the model with a richer variety of examples to learn

from. This strategy is crucial for enhancing performance and ensuring the model generalizes well to new, unseen data.

Image Processing

In addition to the augmentation techniques, we needed to ensure our images were properly processed before feeding them into the model. Here's what we did:

- **Converting to Grayscale**: We converted all images to grayscale to simplify the input data. This step helps the model focus on the essential shapes and structures within the flowers without being distracted by color variations.
- **Applying HOG (Histogram of Oriented Gradients)**: For feature extraction, we used HOG. This method captures the distribution of gradient orientations in the images, highlighting edges and contours that are crucial for distinguishing between different flower types. HOG proved to be a reliable choice for our classification task.
- **Attempt with SIFT (Scale-Invariant Feature Transform)**: We initially considered using SIFT for feature extraction as well. However, after testing it with our model, we found that it didn't yield the desired results. While SIFT is known for detecting local features effectively, it just didn't mesh well with our approach. As a result, we decided to stick with HOG for its superior performance in this context.

These preprocessing steps were vital in preparing our dataset, allowing the model to learn effectively from the images provided.

Dimensionality Reduction

To make our model more efficient and manageable, we turned to Principal Component Analysis (PCA) for dimensionality reduction. This technique is great for transforming high-dimensional data into a lower-dimensional form while still keeping the majority of the variance intact.

For our project, we set a goal to maintain at least 95% of the variance in the dataset. After running PCA, we discovered that we could effectively reduce the number of features to around 2997 components. This was a significant reduction, allowing us to keep the most relevant information while filtering out noise and less important details.

We also created a graph to visualize the explained variance, which showed how much variance each principal component accounted for. This graph was helpful in confirming our decision to stick with the 95% threshold. It reassured us that we were capturing the essential features necessary for classifying the flowers, without burdening the model with too many dimensions.

Overall, using PCA not only streamlined our model but also improved the speed and performance during training. This approach made the classification task much more efficient, allowing us to focus on what really matters in distinguishing the different types of flowers.

Model Architecture

For our project, we developed a feed-forward neural network specifically designed to classify flower images. The architecture of the model consists of multiple layers that allow it to learn complex patterns from the input data effectively.

The first layer in our network contains 2048 neurons and utilizes the ReLU (Rectified Linear Unit) activation function. ReLU is known for its ability to introduce non-linearity into the model, enabling it to capture intricate relationships in the data. To combat overfitting, we applied a dropout rate of 50%. This means that during training, half of the neurons in this layer are randomly deactivated, which encourages the network to learn more robust features.

Subsequent layers include two additional dense layers, each with 1024 neurons and followed by batch normalization. Batch normalization normalizes the inputs to these layers, which helps stabilize the learning process and allows the network to train more effectively by reducing internal covariate shift. This technique also facilitates the use of higher learning rates, leading to faster convergence.

Following these layers, we added another dense layer with 512 neurons, followed by a layer containing 256 neurons. Finally, the output layer consists of 60 neurons, each representing one of the flower classes. The softmax activation function is applied in this layer to generate probabilities for each class, allowing the model to make predictions about which flower type is present in the input image.

Normalization Techniques

To enhance the model's training process, we incorporated several techniques that are vital for improving performance:

- **Dropout**: By randomly dropping a portion of the neurons during training, we help prevent overfitting. This technique forces the network to be less reliant on any specific set of features, which ultimately results in a more generalized model.
- **Batch Normalization**: This method was used after certain layers in our network. By normalizing the inputs to these layers, we stabilize the learning process and improve the overall efficiency of training. Batch normalization reduces the risk of issues like vanishing gradients, which can hinder deep networks.

Training Process

For training, we used Stochastic Gradient Descent (SGD) as our optimizer. We set the learning rate to 0.01 and utilized a momentum value of 0.9. The loss function employed is categorical cross-entropy, which is particularly suited for multi-class classification tasks. During the training phase, we trained the model for 10 epochs with a batch size of 32, allowing it to learn from the training data while validating its performance against a separate validation set.

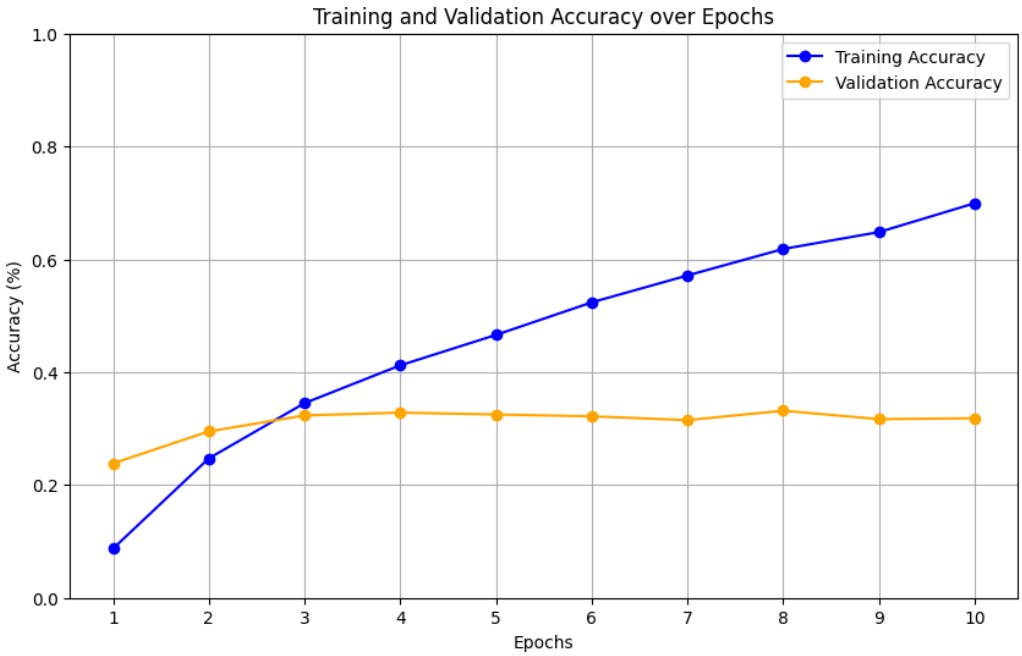
Through careful architecture selection and the application of normalization techniques, we aimed to create a robust model capable of accurately classifying images of flowers.

Results and Evaluation

After training our feed-forward neural network for 10 epochs, we evaluated the model's performance using accuracy as our primary metric. The results revealed that the highest accuracy on the validation set reached around 33%, which indicates that the model has started to learn to classify the flower images but still has considerable room for improvement.

Throughout the training process, we observed a gradual increase in training accuracy, reflecting the model's ability to learn from the data. However, the validation accuracy demonstrated fluctuations, suggesting that the model was facing challenges in generalizing its learned features to unseen data. This is often indicative of overfitting, where the model performs well on the training set but struggles with validation data.

The accompanying graph illustrates the training and validation accuracy over the epochs, providing a visual representation of the model's learning trajectory.



Conclusion

In this project, we developed a feed-forward neural network to classify images of flowers, leveraging a dataset that comprised 60 different types. Despite the initial challenges posed by the relatively small dataset, we implemented various strategies such as image augmentation and dimensionality reduction through PCA, which helped streamline our model.

By the end of our training process, the model achieved a validation accuracy of approximately 33%. While this indicates that the model has learned to some extent, it also highlights that there is substantial room for improvement. The journey has provided valuable insights into the complexities of image classification and the importance of fine-tuning models to achieve better performance.

Scope for Further Improvement

To enhance the performance of our flower classification model, a primary area for exploration is the adoption of convolutional neural networks (CNNs). Given their proven effectiveness in image-related tasks, CNNs could significantly improve our model's ability to extract and learn spatial hierarchies of features. Transitioning to a CNN architecture would enable the model to automatically capture intricate patterns and structures within the images, likely leading to improved accuracy.

In addition to adopting CNNs, we could implement advanced data augmentation techniques. These methods would further enrich our dataset by simulating variations in flower images, helping the model generalize better to unseen examples. Furthermore, hyperparameter tuning remains crucial; optimizing parameters such as learning rates, batch sizes, and the number of epochs could lead to more effective training and convergence.