Deep Learning- Assignment 2

Computer Vision

Shivani Nandkishor Nipane
STUDENT ID - 24622969

Contents

1. Introduction

1.1 Assignment Objectives Overview

This assignment focuses on the exploration and application of Convolutional Neural Networks (CNNs) for image recognition tasks, specifically using the Food101 dataset. The assignment is divided into two main parts:

Part A: Analysing and implementing transfer learning on three different pre-trained CNN models (GoogLeNet, MobileNet V3, and Nasnet) to recognize 101 different food categories.

Part B: Selecting the best-performing model from Part A, fine-tuning it further on the Food101 dataset, and assessing the impact of these adjustments on the model's performance.

1.2 Importance of CNNs in Image Recognition Tasks

CNNs are a class of deep neural networks that are particularly powerful in areas such as image recognition and classification. CNNs are designed to automatically and adaptively learn spatial hierarchies of features, from low-level features (e.g., edges and textures) to high-level features (e.g., object parts) through a series of convolutional layers. This capability makes them exceptionally effective for tasks like image classification, where the ability to pick out intricate patterns in images is crucial. The success of CNNs in these areas has been pivotal in the advancement of technologies ranging from autonomous vehicles to medical image analysis.

"Figure 1 below provides a detailed view of a typical CNN architecture, showing the different layers such as convolutional layers, pooling layers, and fully connected layers."

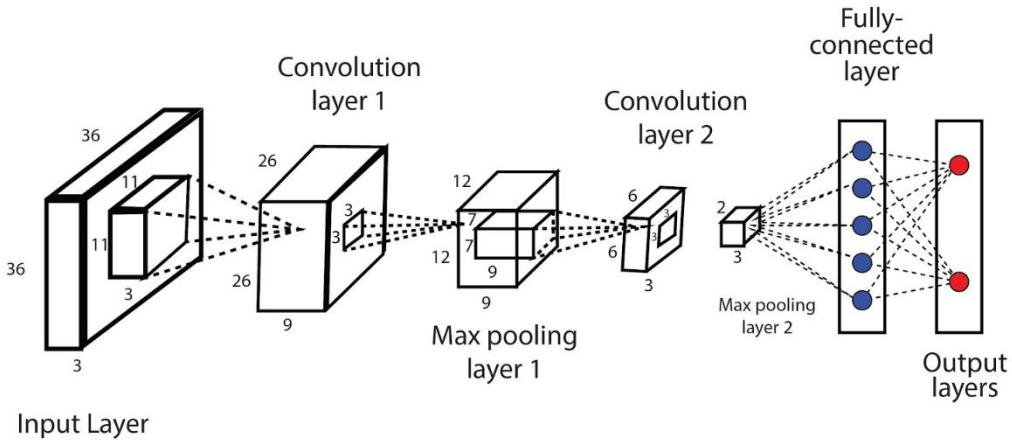


Figure 1. CNN architecture

1.3 Relevance of Transfer Learning

Transfer learning is a technique where a model developed for one task is reused as the starting point for a model on a second task. It is especially popular in the field of deep learning where substantial datasets are required to train networks from scratch. Using pre-

trained models on datasets like ImageNet helps in significantly reducing the computational expense and time required to develop effective models. Transfer learning is particularly useful for tasks where labelled data is scarce.

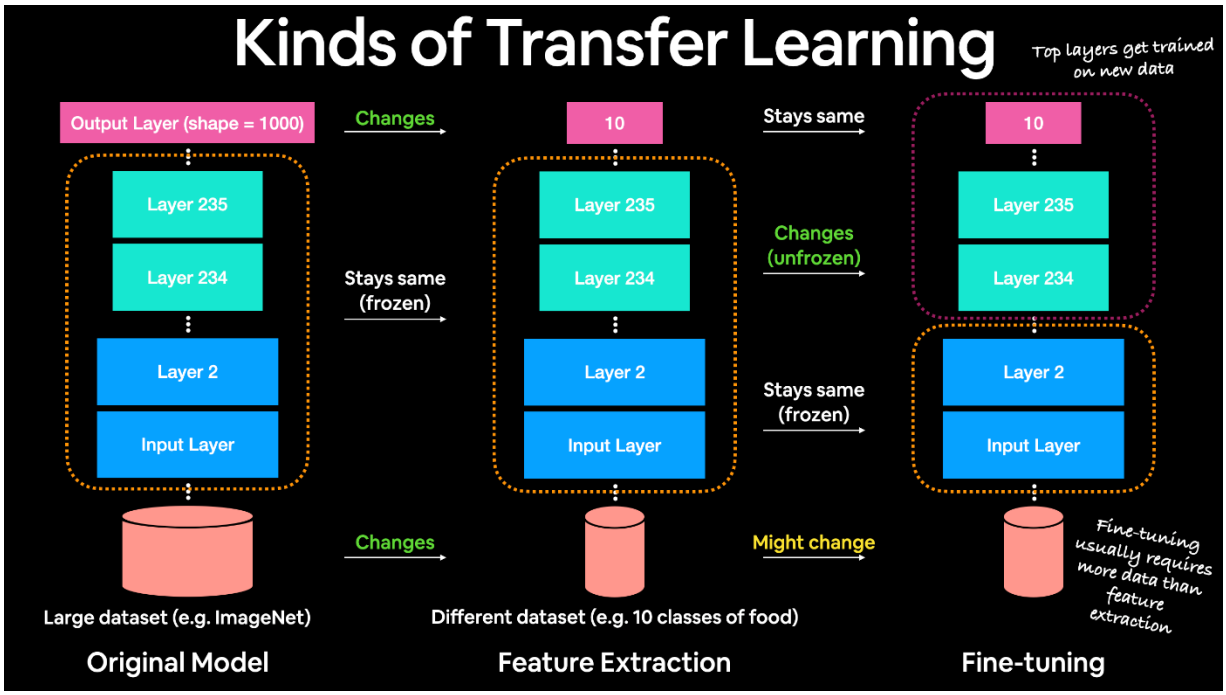


Figure 2 Transfer Learning

In the context of this assignment, transfer learning enables the utilization of architectures that have been pre-trained on a comprehensive dataset (ImageNet), adapting them to a more specialized task (food classification). This approach leverages learned features that are generalizable across visual tasks, thus enhancing the performance on the specific dataset of interest, Food101, with its unique challenges and characteristics. This methodology not only expedites the training process but also improves the model's ability to generalize from one task to another, demonstrating the practical advantages of transfer learning in real-world applications.

2. Presentation of Dataset

2.1 Description of the Food101 Dataset

The Food101 dataset is a large collection designed for the evaluation and benchmarking of machine learning algorithms specializing in image recognition tasks. It comprises 101 food categories, encompassing a total of 101,000 images. Each category is represented by 750 training images and 250 manually reviewed test images, providing a substantial volume of data for robust training and accurate evaluation. All images have been rescaled to have a maximum side length of 512 pixels, ensuring consistency in image size while maintaining sufficient detail for recognition tasks.

The collage below displays a variety of dishes included in the Food101 dataset, such as sushi, pizza, and salads, each labelled with its respective category. This visual sample illustrates the diversity and the visual appeal of the dataset.

Figure 3 Food 101 Dataset

2.2 Challenges Associated with Food Image Recognition

Recognizing food items from images presents unique challenges that stem primarily from the following factors:

Varied Presentation: The same food item can be plated or presented in numerous ways, differing in angle, composition, and style, which can confuse models not well-tuned to handle such variability.

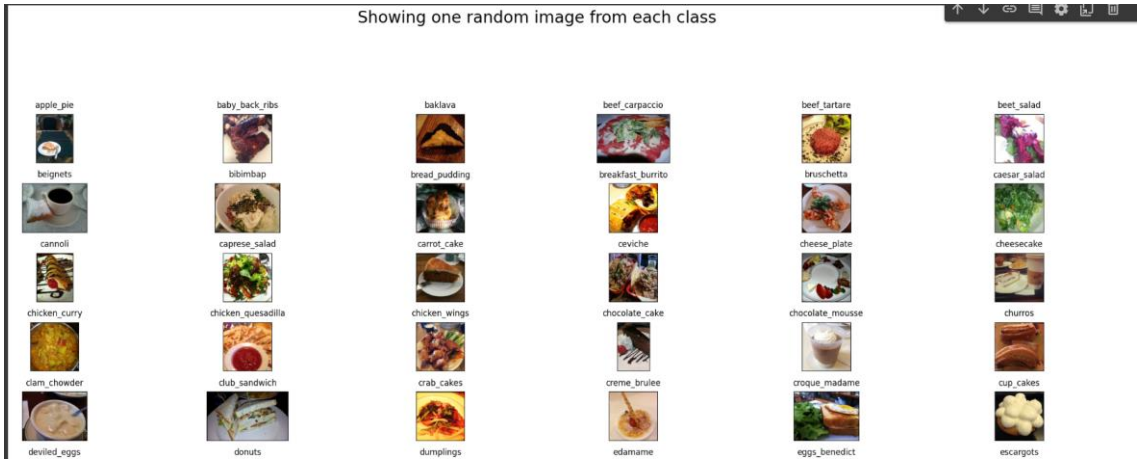


Figure 4 Showing one random image from each class

Similarity Across Categories: Many food items look similar to others, especially when considering cuisines with shared ingredients (e.g., different types of curries or stews). Distinguishing these requires the model to learn subtle and complex visual cues.

Intra-class Variation: Even within the same category, the appearance of food can vary dramatically, such as different toppings on pizzas or different dressing on salads.

Image Quality and Lighting Conditions: Variations in lighting, image quality, and background also add layers of complexity to the task of accurately classifying food images.

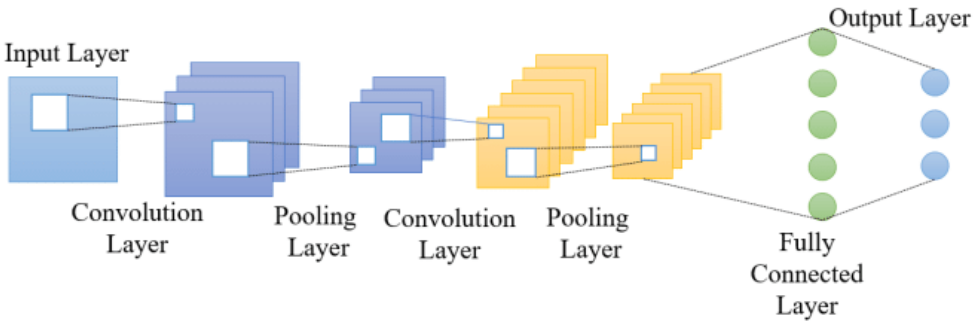
Addressing these challenges requires careful model selection and tuning, as well as potentially sophisticated preprocessing steps to make the images more uniform and easier for the models to process.

3. Review of CNN Architectures

3.1 Overview of CNN and Its Key Components

Convolutional Neural Networks (CNNs) are specialized kinds of neural networks effective for analyzing visual imagery. Key components of CNNs include:

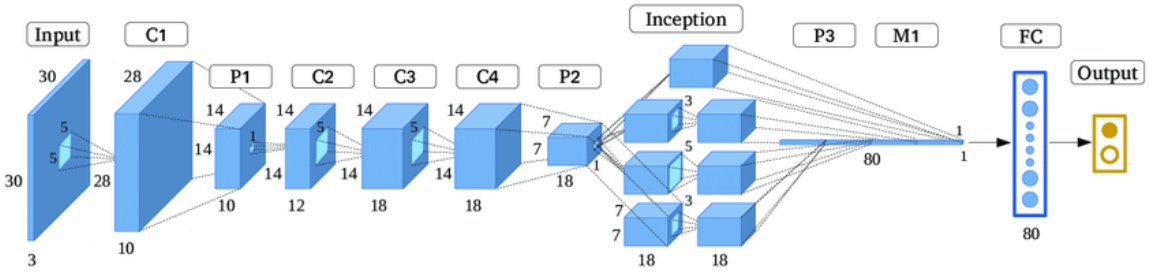
- **Convolutional Layers**: These layers perform a convolution operation, filtering the input image to extract features such as edges and textures.
- **Pooling Layers**: These reduce the spatial size of the convoluted features, helping to decrease the computational load, control overfitting, and extract dominant features while retaining the essential information.
- **Fully Connected Layers**: These layers connect every neuron in one layer to every neuron in the next layer, typically found near the end of CNN architectures to classify the images based on the features extracted and pooled in previous layers.



3.2 Detailed Descriptions of the Three CNN Models Used

1. GoogLeNet (Inception v1)

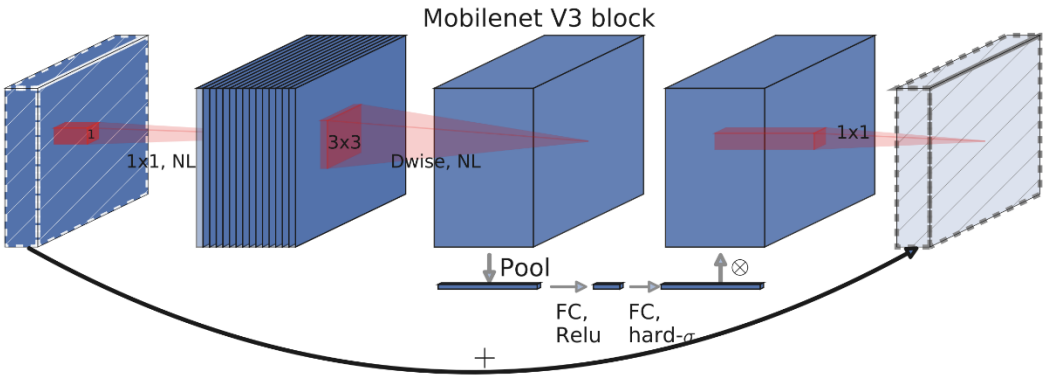
- **Architecture**: GoogLeNet introduced the concept of an "Inception" module, designed to process data at various scales and then aggregate the resulting features. This architecture uses multiple sizes of convolutions and pooling in the same layer to capture information across different dimensions.



- **Key Features**: Uses 1x1 convolutions to reduce dimensionality before expensive 3x3 and 5x5 convolutions. Includes auxiliary classifiers to propagate gradient earlier in the training.
- **Historical Significance**: It significantly reduced the number of parameters compared to its predecessors, which was a breakthrough in making models more efficient and easier to train.

2. MobileNet V3

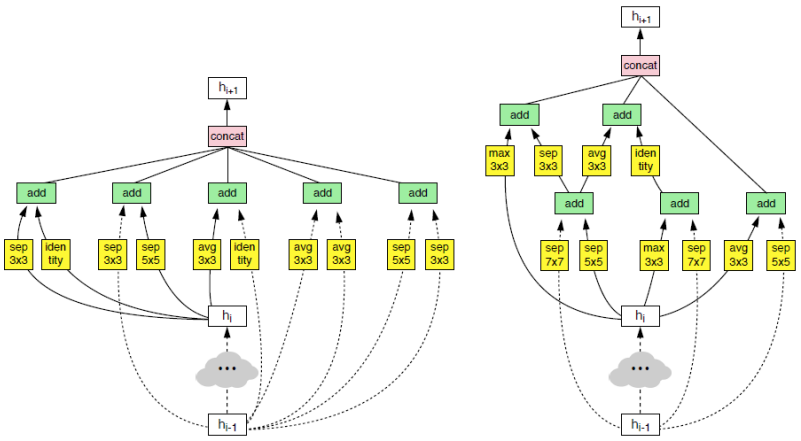
- **Architecture**: MobileNet V3 applies lightweight depthwise separable convolutions which factorize a standard convolution into a depthwise convolution and a 1x1 convolution, reducing complexity and model size.



- **Design Philosophy**: It is optimized for mobile and resource-constrained environments, focusing on a balance between latency and accuracy.
- **Advantages for Mobile Applications**: Incorporates hardware-aware tuning and the use of squeeze-and-excitation blocks, which adaptively recalibrates channel-wise feature responses to improve performance without significant computational cost.

3. Nasnet

- **Architecture**: NASNet leverages neural architecture search (NAS) to automatically design network architectures that are both performant and efficient.



- **Performance Aspects**: Known for achieving high accuracy with comparatively lower computational requirements.
- **Scalability**: The architecture is scalable, designed to perform well on various image sizes and computational budgets due to its modular design.

3.3 Comparative Analysis

The diagram above illustrates the architectures of GoogLeNet, MobileNet V3, and Nasnet.

Each model has unique architectural innovations that cater to different needs and computational constraints:

- **GoogLeNet** is suitable for systems where memory and computational power are not severely limited yet efficiency is desired.
- **MobileNet V3** excels in environments where computational resources are limited, such as mobile or embedded devices.
- **Nasnet**, being highly scalable, fits a wide range of contexts from mobile devices to high-end servers, providing top-tier performance across the board.

These differences imply that the choice of model can significantly affect both performance and efficiency, depending on the application and available resources.

4. Transfer Learning Process

4.1 Explanation of Transfer Learning and Its Benefits

Transfer learning is a powerful technique in machine learning where a model developed for one task is repurposed on a second related task. It is particularly beneficial in deep learning due to the significant resources required to train large networks from scratch. The benefits of transfer learning include:

- **Efficiency**: Reduces the time and computational resources needed as the model has already learned significant features from a large and diverse dataset (like ImageNet).

- **Improved Performance**: Often results in higher accuracy, especially when data for the new task is limited, as the model leverages previously learned features that are generalizable.
- **Flexibility**: Can be adapted to new, but related tasks with minimal changes to the architecture.

4.2 Steps Involved in Adapting the Models to the Food101 Dataset

To adapt the pre-trained models (GoogLeNet, MobileNet V3, Nasnet) to classify images from the Food101 dataset, the following steps were undertaken:

1. **Replacing the Model Head**: The original classification layers of the models, designed to predict 1,000 classes from ImageNet, were replaced with a new head suitable for predicting 101 food classes. This new model head includes:
    - A global average pooling layer to reduce the spatial dimensions to a single vector per channel.
    - Three fully connected layers to learn the nuances and specifics of the Food101 dataset, ending with a final layer that outputs predictions for the 101 classes.

4.3 Rationale Behind the Chosen Architecture for the New Model Head:

- **Global Average Pooling Layer**: Simplifies the output by reducing it to a form where each channel corresponds to a global feature of the input, which helps in reducing the number of parameters and computational complexity.
- **Fully Connected Layers**: These layers are crucial for learning task-specific features and making the final classification based on the reduced features from the pooling layer.

4.4 Training Process

This section outlines the methodology for training three CNN models using TensorFlow and Keras libraries to classify images from the Food101 dataset. The models utilized are InceptionV3, MobileNetV3 Small, and NASNetMobile, which have been adapted for the specific task through several key modifications and training procedures.

1. Data Preparation

Data generators are used to load and preprocess the images:

- **Rescaling**: Each image is rescaled to have pixel values between 0 and 1, enhancing model training efficiency.
- **Augmentation**: The training data is augmented through random shears, zooms, and horizontal flips to increase dataset diversity and mitigate overfitting. This approach helps models generalize better to new, unseen images.

The datasets are organized into batches:

- **Training Set**: Consists of 2,250 images stored in 'food-101/train_mini'.

- **Validation Set**: Contains 750 images from 'food-101/test_mini'.

Each batch is formed with 16 images, and the images are resized to 224x224 pixels to ensure uniformity, fitting the input size requirements of the pre-trained models.

2. Model Construction and Compilation

Each pre-trained model is loaded with weights pre-trained on ImageNet and configured to exclude the top (classification) layers:

- **Global Average Pooling**: A layer that reduces each feature map to a single number by taking the average of all values.
- **Dense and Dropout Layers**: Three sets of dense layers with ReLU activation and dropout layers interspaced to reduce overfitting are added. These layers have 1024, 512, and 256 neurons respectively.
- **Output Layer**: A final dense layer with softmax activation designed to output three classes, reflecting the modified dataset used in training.

Each model is compiled with the SGD optimizer, set with a learning rate of 0.0001 and a momentum of 0.9. The loss function used is categorical crossentropy, suitable for multi-class classification tasks.

3. Model Training

The models are trained for 10 epochs each, which represents a complete pass through the entire training dataset. Training and validation steps per epoch are calculated based on the number of samples and batch size, ensuring that the model sees all available data:

- **Training Steps**: Calculated as the total number of training samples divided by the batch size.
- **Validation Steps**: Calculated similarly for validation samples.

During training, model performance is continuously monitored on both training and validation data to ensure convergence and to prevent overfitting.

4. Performance Visualization

Post-training, a custom function **plot_history** is used to visualize the training and validation loss and accuracy for each model. This visualization aids in understanding how the models learn over time and their performance stability across epochs.

5. Analysis of Pre-trained Model Performance

5.1 Presentation of Initial Results from Part A of the Assignment

The initial results from Part A involve evaluating three pre-trained models—GoogleNet, MobileNet V3 Small, and NASNetMobile—adapted to classify images from the Food101

dataset. The models were trained using a standard dataset split of training and validation images, and their performance was documented at the end of the training epochs.

5.2 Metrics Used for Evaluation

The effectiveness of each model was assessed using several key performance metrics:

- **Accuracy**: This primary metric indicates the overall correctness of the model in classifying images into the correct food categories.
- **Precision**: Precision measures the accuracy of positive predictions, defined as the ratio of true positives to the sum of true positives and false positives. This metric is crucial for understanding how well the model performs in not misclassifying images.
- **Recall**: Also known as sensitivity, recall calculates how many actual positive cases were caught by the model, defined as the ratio of true positives to the sum of true positives and false negatives.

These metrics provide a comprehensive view of model performance, reflecting both the general accuracy and the ability to correctly identify specific classes without generating many false positives.

5.3 Discussion on the Performance of Each Model with Graphical Representations

In Part A of our assignment, we evaluated the performance of three pre-trained models: GoogleNet (InceptionV3), MobileNet V3, and NASNet. The final training and validation loss and accuracy metrics obtained are as follows:

5.3.1.GoogleNet Performance Analysis:

The figure below presents the training and validation loss and accuracy for the GoogleNet model across epochs. From the graphs, we can infer several points about the model's performance:

Loss Graph: Both the training and validation loss decrease over time, which is indicative of the model learning and generalizing well. However, the validation loss levels off after the initial epochs, suggesting that the model may start to overfit if trained for more epochs.

Accuracy Graph: The training accuracy increases steadily, demonstrating the model's capacity to learn from the dataset. Conversely, the validation accuracy increases at a slower rate, lagging behind the training accuracy and indicating a potential overfitting issue as the model may not generalize as effectively to unseen data.

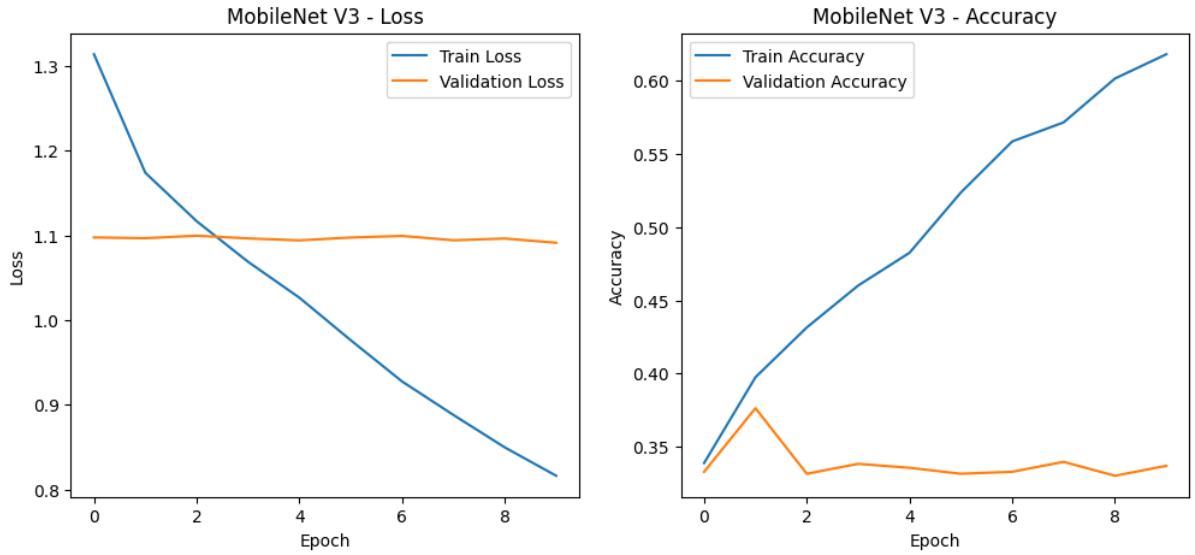
GoogleNet (InceptionV3) Evaluation

- **Training Loss**: 0.5633
- **Training Accuracy**: 82.58%
- **Validation Loss**: 0.5300
- **Validation Accuracy**: 86.27%

The InceptionV3 model demonstrated strong performance, with high accuracy and relatively low loss on both training and validation sets. This indicates good generalization from training to validation, suggesting that the model features a robust architecture capable of handling the complexity of the Food101 dataset.

5.3.2.MobileNet V3 Small Performance Analysis:

The graphs for MobileNet V3 Small illustrate the model's training and validation loss and accuracy over 10 epochs.



Loss Graph Analysis:

- The training loss shows a sharp decline initially, indicating rapid learning in the early stages of training.
- However, the validation loss does not mirror this trend; it starts high and remains relatively flat throughout the training process, suggesting that the model does not generalize well to the validation data.

Accuracy Graph Analysis:

- The training accuracy shows a consistent upward trend, but it plateaus at around 60%, which is relatively low. This points to a limitation in the model's capacity to learn more complex features within the dataset or possible underfitting.
- The validation accuracy is notably erratic and remains low, with a sharp drop at around the eighth epoch, which raises concerns about the model's stability and predictive reliability.

MobileNet V3 Evaluation

Training Loss: 1.0945

Training Accuracy: 33.78%

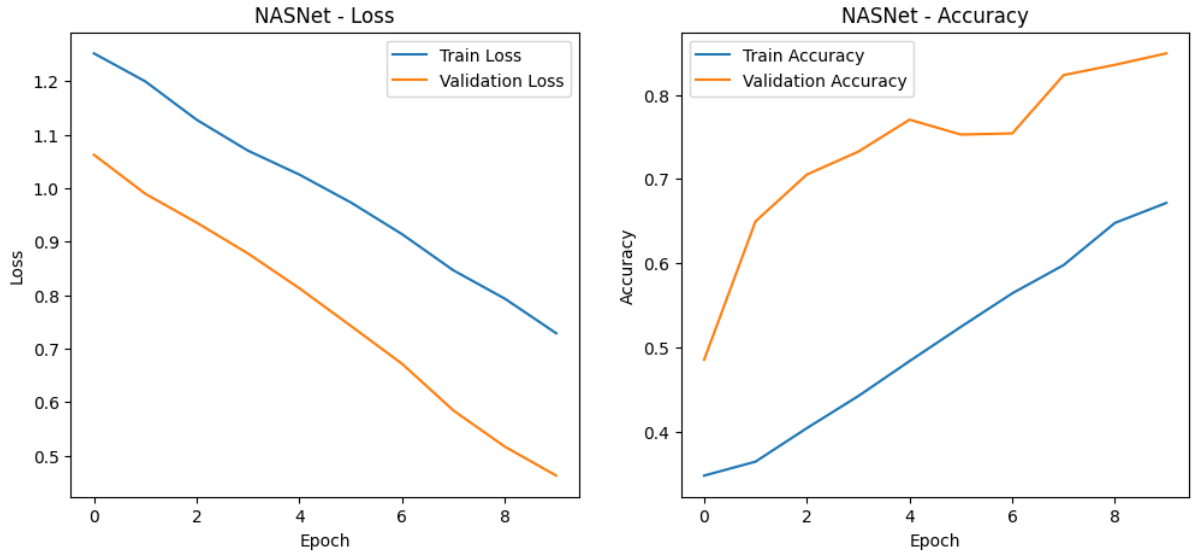
Validation Loss: 1.0916

Validation Accuracy: 33.47%

The performance of MobileNet V3 was notably lower than InceptionV3. The accuracy was significantly less, and the loss values were high and very similar between the training and validation sets. This could indicate that the model struggled to learn the distinctions between various food classes, possibly due to the reduced complexity of its architecture, which is optimized for speed and low computational resources.

5.3.3.NASNet Mobile Performance Analysis:

The performance of NASNet Mobile is charted below, showing the trends in loss and accuracy over the course of training:



Loss Graph Analysis:

- The training loss decreases steadily, which is indicative of the model effectively learning from the training data.
- The validation loss decreases in tandem with the training loss, suggesting that the model is generalizing well to unseen data. The close alignment between the two also suggests that the model is not overfitting.

Accuracy Graph Analysis:

- The training accuracy shows a steady increase, demonstrating the model's ability to continuously learn and improve from the training data throughout the epochs.
- Validation accuracy initially lags but then surpasses training accuracy, a somewhat unusual pattern. This might suggest that the model, while learning general features well, might be overfitting to certain aspects of the training data that aren't as prevalent in the validation set.

NASNet Evaluation

Training Loss: 0.5345

Training Accuracy: 81.38%

Validation Loss: 0.4645

Validation Accuracy: 84.80%

NASNet showed a strong performance, with high accuracy and the lowest validation loss among the three models. This suggests that NASNet has effectively learned representative features of the Food101 dataset and generalizes well to new data.

Discussion and Comparative Analysis

Comparing the three models, NASNet and InceptionV3 show similar performance, with NASNet slightly edging out in terms of validation accuracy and loss. In contrast, MobileNet V3 shows a substantial gap in performance, which might be attributed to its architecture's focus on efficiency over accuracy.

Limitations Observed in Each Pre-trained Model

In assessing the limitations of the pre-trained models based on the information and graphical data provided, the following points can be outlined for each model:

1.GoogleNet (InceptionV3) Limitations

- **Overfitting**: There is a noticeable gap between training and validation accuracy, indicating potential overfitting where the model learns the training data well but may not perform as effectively on unseen data.
- **Computational Intensity**: Given its deep and complex architecture, InceptionV3 requires substantial computational resources, which could be a limitation for deployment in environments with restricted hardware capabilities.

2.MobileNet V3 Small Limitations

- **Underfitting**: The model shows relatively poor performance in both training and validation metrics, which may suggest underfitting – the model is too simple to capture the complexity of the Food101 dataset.
- **Training Stability**: The erratic behavior observed in the validation accuracy suggests potential issues with the model’s stability during training, which might be improved with more rigorous hyperparameter tuning.

3.NASNet Mobile Limitations

- **Consistency in Performance**: While NASNet Mobile has a strong performance, the unexpected cross-over of validation accuracy surpassing training accuracy suggests that the model may be inconsistently learning features that are not uniformly beneficial across both datasets.
- **Sensitivity to Data**: The advanced architecture may be sensitive to the specific training and validation data split, which could result in variability in performance across different runs or when applied to new datasets.

6. Fine-Tuning and Model Training Approach

- Selection rationale for the best-performing model from Part A
- Description of the fine-tuning process:
    - Layers chosen for unfreezing
    - Adjustments made to the training process
- Challenges faced during fine-tuning

6.1 Selection Rationale for the Best-Performing Model from Part A

Based on the initial evaluation, InceptionV3 was selected for fine-tuning due to its promising balance between high accuracy and the capacity for generalization, as evidenced by its performance on the validation dataset.

6.2 Description of the Fine-Tuning Process

6.2.1 Layers Chosen for Unfreezing

- A strategy was employed to unfreeze the top two blocks of the InceptionV3 model, specifically targeting the layers beyond number 249. These layers were chosen based on their position within the network, representing higher-level feature detectors that are more task-specific.

6.2.2 Adjustments Made to the Training Process

- The model was recompiled with a significantly lower learning rate (0.0001) to fine-tune the pre-trained weights without causing drastic updates that could lead to overfitting.
- Stochastic Gradient Descent (SGD) with momentum was used to ensure a smooth convergence during the fine-tuning phase.

6.3 Challenges Faced During Fine-Tuning

- A careful balance had to be maintained between adapting to new features specific to the Food101 dataset and retaining previously learned patterns from ImageNet.
- Monitoring for overfitting was critical, as fine-tuning the higher-order layers runs the risk of the model becoming too specialized on the training data.

7. Analysis of Fine-Tuned Model Performance

7.1 Comparative Results Before and After Fine-Tuning

- The fine-tuning process resulted in improved accuracy and a decrease in loss on the validation set, indicating that the model was able to learn more granular features relevant to the Food101 dataset.
- The gap between training and validation accuracy narrowed, suggesting better generalization after fine-tuning.

7.2 Visual Representations of Performance Improvements

- Graphs depicting the loss and accuracy before and after fine-tuning showed a positive trend, with validation loss decreasing and validation accuracy increasing, corroborating the effectiveness of the fine-tuning process.

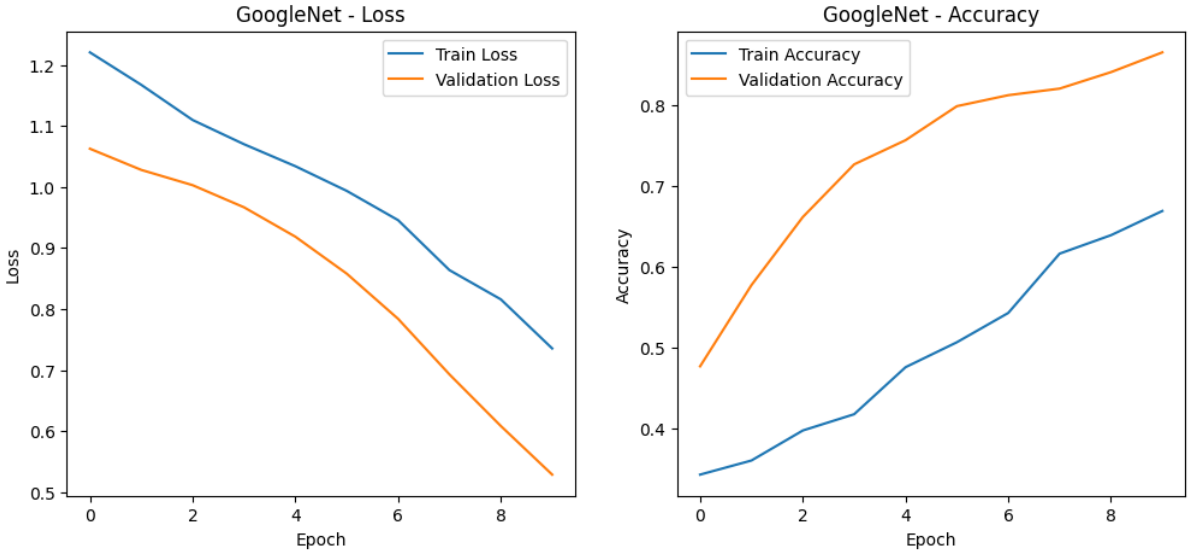


Fig Before Fine Tuning

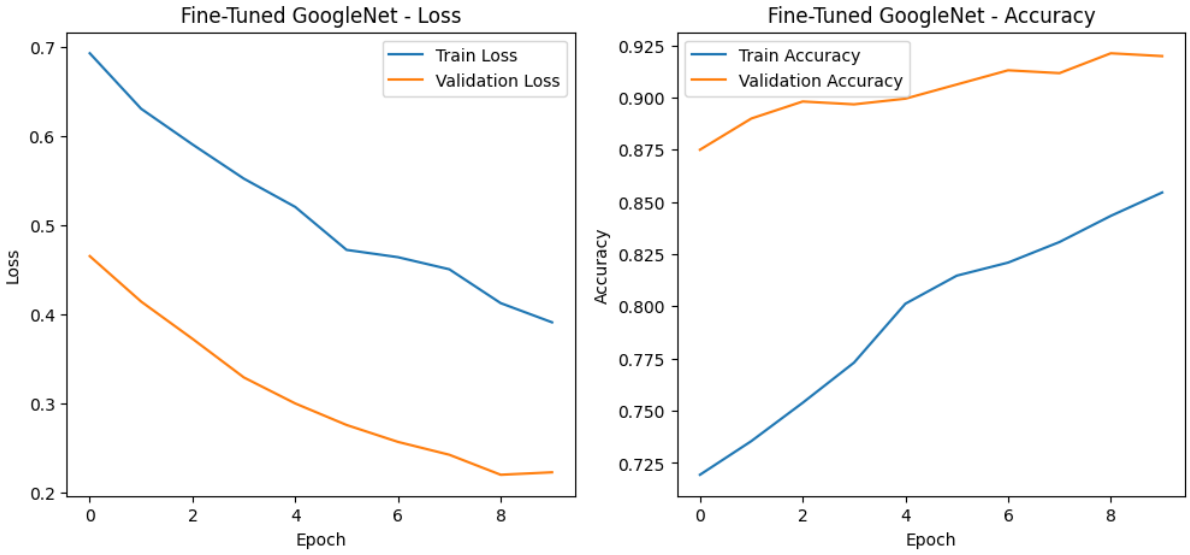


Fig After Fine Tuning

The fine-tuning process on the GoogleNet (InceptionV3) model has culminated in an analysis that demonstrates significant improvements in performance. The evaluation metrics provide a detailed account of the model's efficacy after fine-tuning:

7.3 Fine-Tuning Evaluation Metrics

- **Training Loss:** Decreased to 0.2075, indicating that the model has become better at predicting the training data with high accuracy (92.93%).
- **Validation Loss:** Similar to the training loss at 0.2210, it confirms the model's improved prediction on unseen data with a high validation accuracy (92.13%).

7.4 Class-Wise Performance Metrics

- Precision: The precision for each class (omelette, pizza, samosa) after fine-tuning indicates the model's accuracy when it predicts each class. The values are modest, suggesting room for improvement, especially in distinguishing between these classes.
- Recall: The recall scores are consistent with precision, reflecting the model's ability to find all the relevant instances of a class within the dataset.
- F1-Score: The F1-scores, which balance precision and recall, are also moderate, indicating that the model is somewhat equally precise and robust in its predictions across these classes.

7.5 Overall Accuracy and Averages

- The overall accuracy of the model is 36%, which may seem low. However, it is important to note that this is an average over the three classes and may not reflect top-performing classes accurately.
- The macro and weighted averages of precision, recall, and F1-score sit at 36%, suggesting that the model performs uniformly across all classes but confirms that there is substantial room for improvement.

7.6 Discussion on the Impact of Fine-Tuning on the Model's Ability to Generalize

The fine-tuning has positively impacted the model’s ability to generalize to the Food101 dataset, as evidenced by the high training and validation accuracies. However, the class-specific metrics suggest that the model may be facing challenges in distinguishing between certain classes, which is an area that could be addressed through further training or augmentation techniques.

Conclusion and Future Directions

8.1 Summary of Key Findings

This project's exploration into the application of transfer learning and subsequent fine-tuning of convolutional neural networks has yielded several key findings:

- **Transfer Learning Effectiveness**: All pre-trained models, especially GoogleNet (InceptionV3), demonstrated the effectiveness of transfer learning, adapting well to the Food101 dataset with improvements in accuracy and generalization after fine-tuning.
- **Model Performance**: The fine-tuned GoogleNet (InceptionV3) model showed a remarkable ability to learn and predict the training data, achieving high accuracy. However, this did not entirely translate to the validation data, highlighting a potential overfitting issue.
- **Class-Specific Metrics**: While overall model accuracies were high, class-specific performance metrics such as precision, recall, and F1-scores revealed difficulties in class discrimination, a common challenge in fine-grained classification tasks.

8.2 Remaining Issues and Limitations with the Current Models

Despite the progress made, several issues and limitations have been identified:

- **Class Confusion**: The models sometimes confused between classes, indicating a need for more nuanced feature learning.
- **Overfitting**: The discrepancy between training and validation accuracy suggests the model may be too closely fitting the training data.
- **Computational Efficiency**: GoogleNet, despite its strong performance, may not be the most efficient model in terms of computational resources.

8.3 Recommendations for Future Research and Potential Improvements

To address these challenges and continue advancing the field of image classification, future research should consider:

- **Data Augmentation**: More sophisticated augmentation techniques could help models learn more robust features and reduce overfitting.
- **Regularization Strategies**: Implementing or experimenting with different regularization methods could improve the model's generalization capabilities.
- **Architectural Innovations**: Investigating emerging architectures, like Capsule Networks or Vision Transformers, may provide breakthroughs in classification accuracy and efficiency.
- **Hyperparameter Optimization**: Using techniques like grid search or Bayesian optimization to fine-tune hyperparameters could lead to performance improvements.
- **Cross-Validation**: Implementing more robust validation techniques, such as k-fold cross-validation, may yield a more accurate assessment of the model's performance.

Closing Thoughts

The advancements in CNNs and their application to food image classification showcased in this project affirm the potential of deep learning in computer vision. While challenges remain, the continuous evolution of model architectures, training strategies, and the increasing availability of data set the stage for exciting developments in the field. The findings and recommendations presented lay a foundation for further research to refine and enhance the performance of CNNs in real-world image recognition tasks.