**Calorie estimation of real time south Indian food data using Convolutional Neural Network**

**Abstract**

This research focuses on estimating the calorie content of South Indian food dishes in real-time using Convolutional Neural Networks (CNNs). The study aims to develop a deep learning model capable of analyzing food images and predicting their nutritional value, specifically the calorie count. A dataset of South Indian food images, along with corresponding calorie values, is used to train the CNN model. The model is designed with multiple convolutional layers to extract relevant features from the images, followed by fully connected layers to perform regression and predict the calorie content. Data preprocessing techniques such as image resizing, normalization, and augmentation are applied to improve model performance and generalization. The results demonstrate that CNN-based models can effectively estimate calorie content, offering a potential solution for real-time dietary tracking and personalized nutrition in applications such as mobile health apps and smart kitchen assistants.

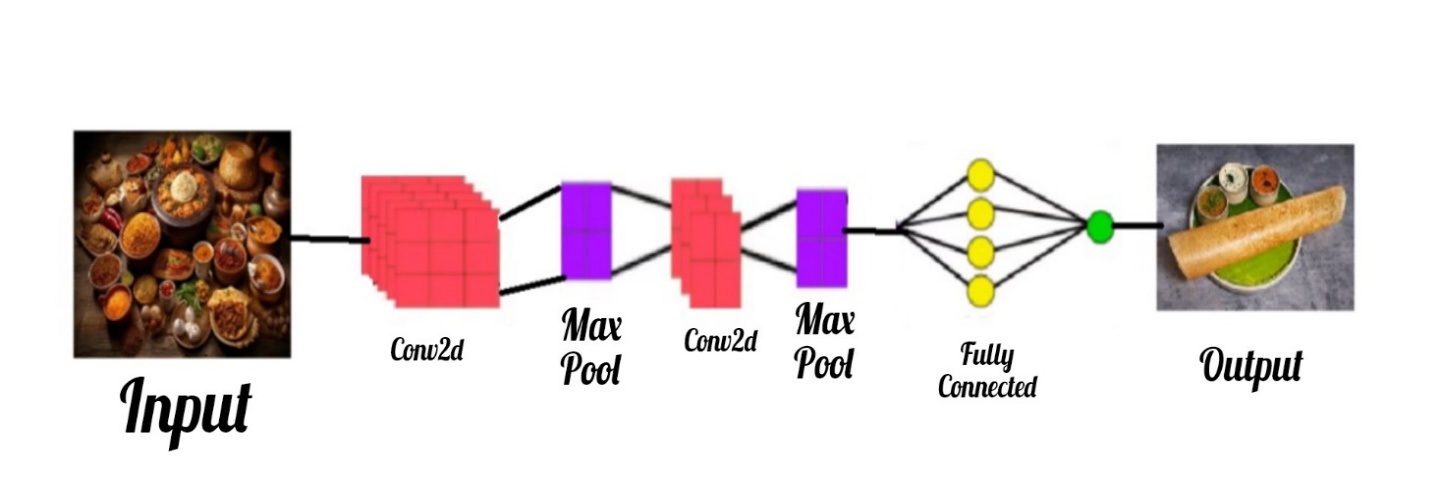
**Keywords**

South Indian Food, Calorie Estimation, Convolutional Neural Networks (CNN), Deep Learning, Image Recognition, Nutritional Prediction, Real-time Food Analysis, Dietary Tracking, Food Classification, Mobile Health Apps

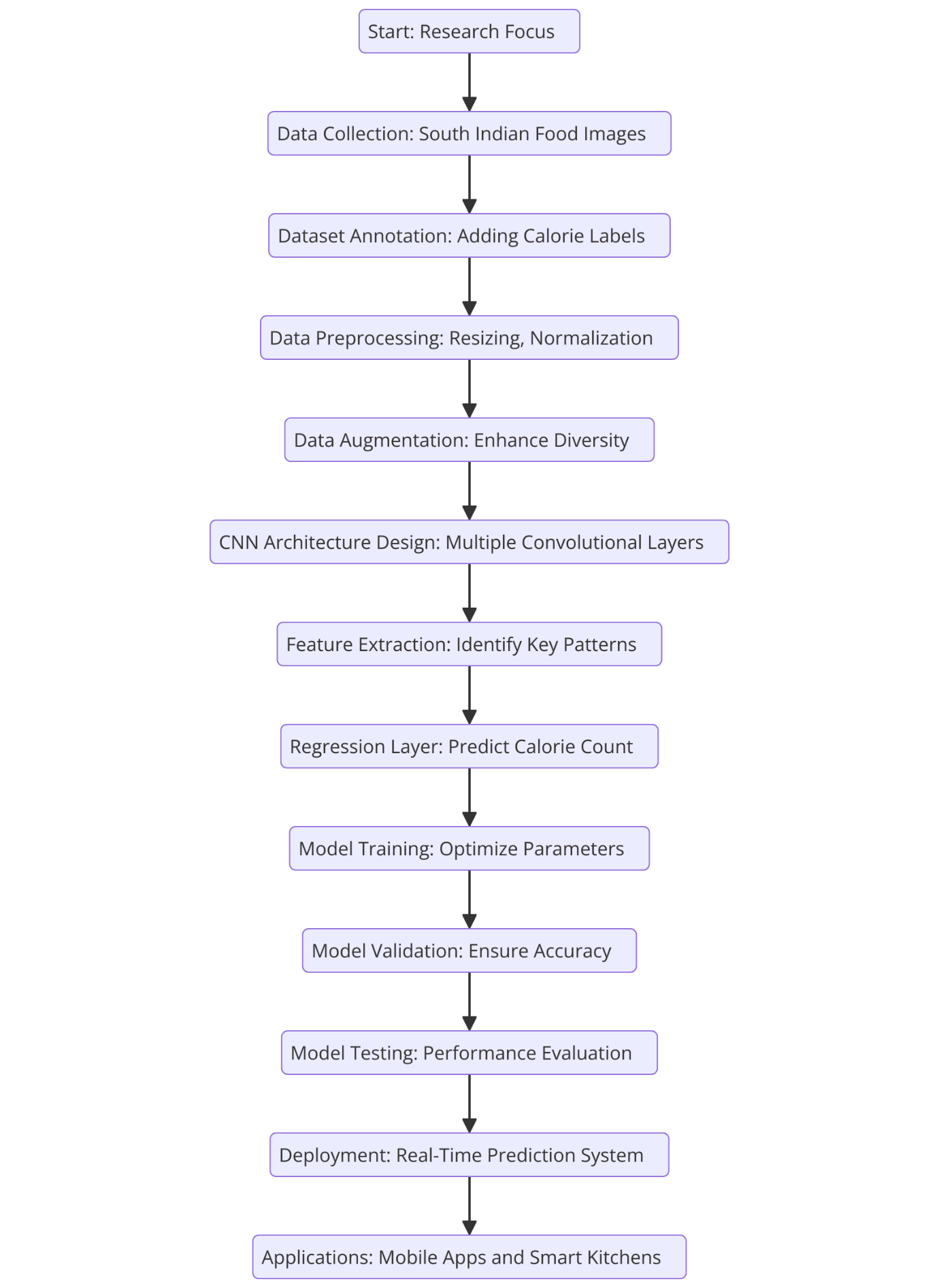
**METHODOLOGY:**

****

**CNN Architecture:**



**Flowchart:**

****

**Data collection:**

1. **Data Collection:**
   * **Source:** The dataset is composed of South Indian food images and corresponding calorie estimates, stored locally in a structured format.
   * **Realism and Suitability:** The dataset captures real-world meal variations, ensuring accurate calorie estimation.
   * **Attributes Collected:** Image filenames, food categories, estimated calorie values, serving sizes, macronutrient composition (protein, fat, and carbohydrates).
   * **Dataset Split:**
     + There are 85+ south Indian foods in the dataset.
     + **Training Data:** 70% of the dataset is allocated for training CNN models.
     + **Validation Data:** 15% is used for hyperparameter tuning and model optimization.
     + **Testing Data:** 15% is reserved for final model evaluation and performance validation.

**Structure:**

* The dataset consists of tabular data in a CSV file and corresponding food images.
* **Key columns in the dataset:**
  + **Image\_Name:** The filenames of images representing various South Indian dishes.
  + **Tags:** Categorical descriptions of the food (e.g., vegetarian, non-vegetarian, spicy).
  + **Calories\_Estimated:** Numerical values representing the estimated calorie content of each dish.

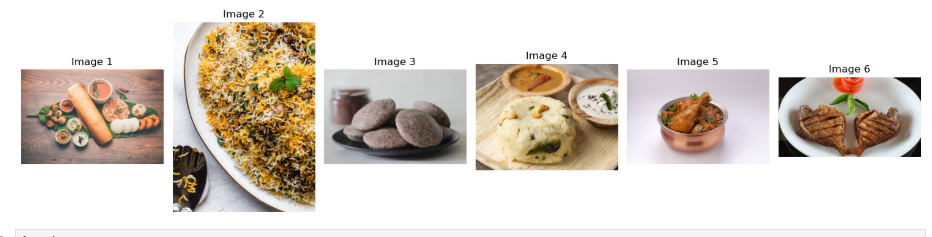
****

**Data Preprocessing:**

* **Annotation:** Food images are labeled with corresponding calorie values.
* **Rotation:** Random rotations (0° to 45°) applied for data augmentation.
* **Resizing:** Standardized to 224x224 pixels to match CNN input requirements.
* **Other Transformations:**
  + Normalization by scaling pixel values to the range [0,1].
  + Data augmentation techniques such as horizontal flips, brightness adjustments, and contrast normalization to improve generalization.
* **Image Resizing**:

All images were resized to 224x224 pixels to match the input requirements of ResNet50, VGG16, and InceptionV3.

* **Before Resizing:**



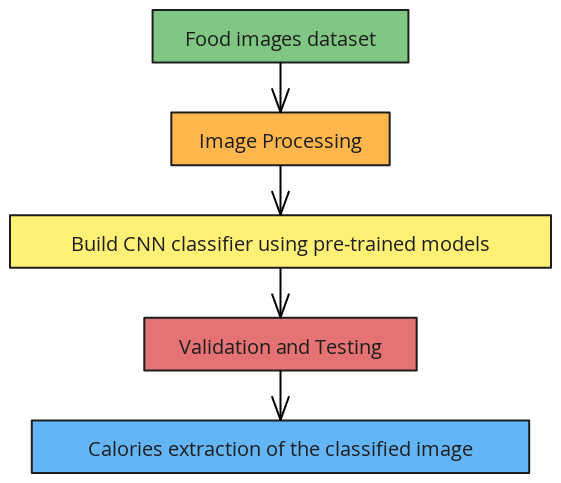
* **After Resizing:**



* **Normalization**:
  + Each pixel value was normalized by dividing by 255.0 to standardize input data.
* **Label Encoding**:
  + Calorie values were encoded as continuous labels for regression tasks.
  + Additional dish classification was included to assist calorie prediction.

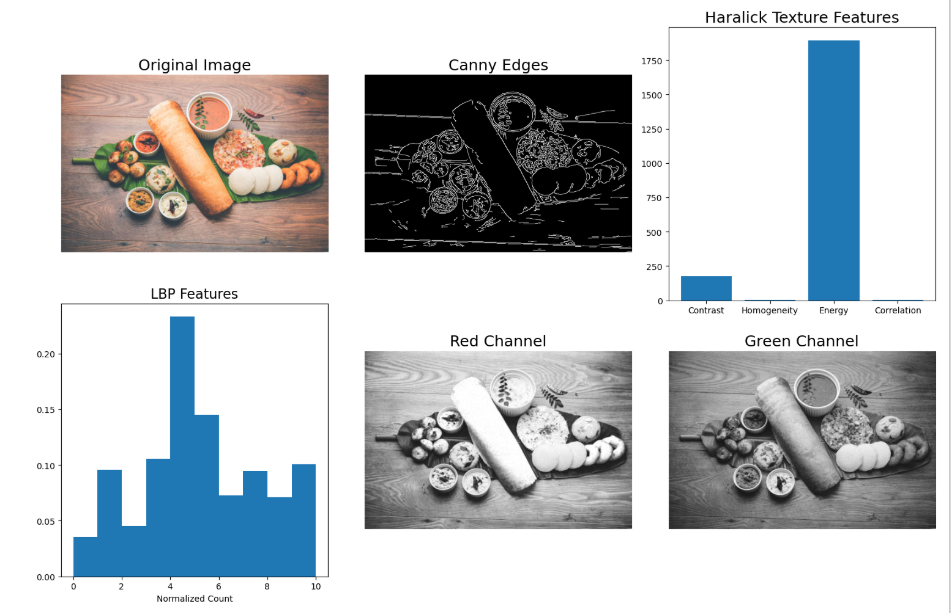


* **Batching**:
* The data was batched (batch size = 32) for efficient model training.
* **Image Preprocessing:**
* Images are resized to a consistent dimension (224x224) to standardize input for deep learning models.
* Pixel values are normalized to the range [0, 1] to improve model performance.
* **Feature Engineering:**
  + Categorical tags are encoded into numerical values using label encoding.
  + Images are transformed into arrays to extract features for training deep learning models.



* **Feature Extraction:**

Feature extraction is a critical step in image processing and computer vision, involving the identification and representation of distinctive structures within an image.



* **Data Augmentation:**

Data augmentation techniques were applied to enhance the diversity of the dataset and improve model generalization:

* **Transformations**:
  + Random rotation (0° to 45°).
  + Horizontal and vertical flips.
  + Zoom-in and zoom-out (range: 0.8x to 1.2x).
  + Brightness and contrast adjustments.
* **Normalization**:
  + Pixel values scaled to [0, 1] for faster convergence during training.





**Sample dataset:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Image Filename** | **Food Item** | **Category** | **Calories** | **Serving Size (g)** | **Protein (g)** | **Fat (g)** | **Carbs (g)** | **Calorie Calculation** |
| dosa1.jpg | Dosa | Breakfast | 120 | 150 | 3 | 5 | 20 | 133 |
| idli1.jpg | Idli | Breakfast | 39 | 100 | 2 | 1 | 8 | 46 |
| sambar1.jpg | Sambar | Soup | 150 | 200 | 5 | 3 | 12 | 123 |
| vada1.jpg | Vada | Snack | 180 | 100 | 6 | 10 | 15 | 204 |
| pongal1.jpg | Pongal | Breakfast | 350 | 200 | 10 | 15 | 50 | 405 |
| biryani1.jpg | Biryani | Main Course | 250 | 300 | 8 | 12 | 60 | 380 |
| uttapam1.jpg | Uttapam | Breakfast | 150 | 150 | 4 | 6 | 8 | 154 |
| ravakesari1.jpg | Rava Kesari | Dessert | 200 | 100 | 2 | 10 | 30 | 218 |
| masaladosa1.jpg | Masala Dosa | Snack | 350 | 200 | 6 | 15 | 50 | 439 |
| appam1.jpg | Appam | Breakfast | 150 | 120 | 3 | 5 | 20 | 133 |
| chettinadcurry1.jpg | Chettinad Curry | Main Course | 250 | 150 | 10 | 20 | 30 | 428 |
| pesarattu1.jpg | Pesarattu | Breakfast | 250 | 150 | 10 | 8 | 30 | 222 |
| puliyodarai1.jpg | Puliyodarai | Main Course | 300 | 200 | 6 | 12 | 60 | 360 |
| curdrice1.jpg | Curd Rice | Side Dish | 150 | 200 | 6 | 8 | 30 | 190 |

**Image Filename:**

* Represents the name of the image file associated with the food item.
* Useful for linking visual data (images) with nutritional data.

**Food Item:**

* Indicates the name of the South Indian food item (e.g., Dosa, Idli, Sambar).
* Essential for identifying the specific dish.

**Category:**

* Classifies the food item into categories such as Breakfast, Snack, Main Course, Side Dish, or Dessert.
* Helps group items based on meal type or purpose.

**Calories:**

* Provides the caloric value of the food item in terms of kcal**.**
* Useful for calorie estimation and dietary planning.

**Serving Size (g):**

* Indicates the weight of the serving (in grams) for which the nutritional values are provided.
* Aids in understanding portion sizes.

**Protein (g):**

* The amount of protein content (in grams) in the given serving size.
* Important for assessing the protein contribution to the diet.

**Fat (g):**

* The fat content (in grams) in the food item.
* Used for monitoring fat intake in a meal or diet.

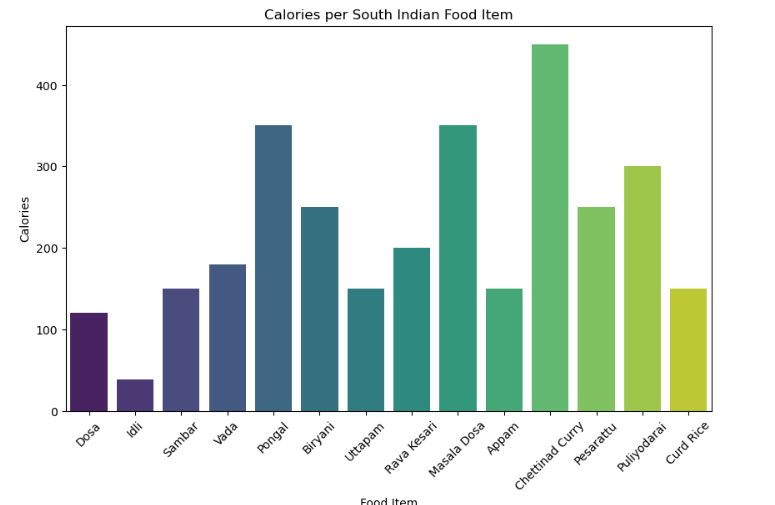
**Carbs (g):**

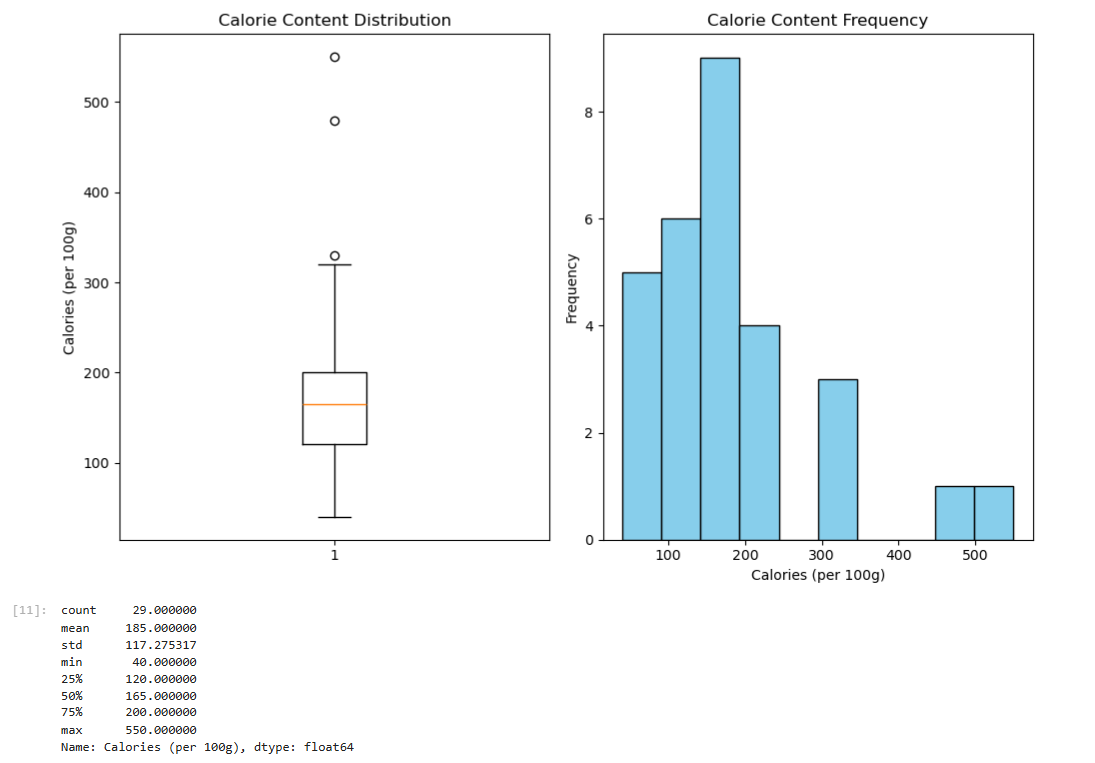
* The carbohydrate content (in grams) in the food item.
* Essential for understanding carbohydrate intake and energy contribution.

**Calorie Calculation:**

* An estimated caloric value based on the macronutrient composition (carbs, protein, fat).
* May follow a caloric formula such as: Calories=(4×Protein)+(4×Carbs)+(9×Fat)\text{Calories} = (4 \times \text{Protein}) + (4 \times \text{Carbs}) + (9 \time\text{Fat})Calories=(4×Protein)+(4×Carbs)+(9×Fat)
* Provides insight into how the total calories were derived.

|  |  |
| --- | --- |
| **Food Item** | **Calories** |
| Dosa | 100 |
| Idli | 50 |
| Sambar | 150 |
| Vada | 200 |
| Pongal | 350 |
| Biryani | 300 |
| Uttapam | 250 |
| Rava Kesari | 300 |
| Masala Dosa | 200 |
| Appam | 400 |
| Chettinad Curry | 450 |
| Pesarattu | 300 |
| Puliyodarai | 350 |
| Curd Rice | 250 |



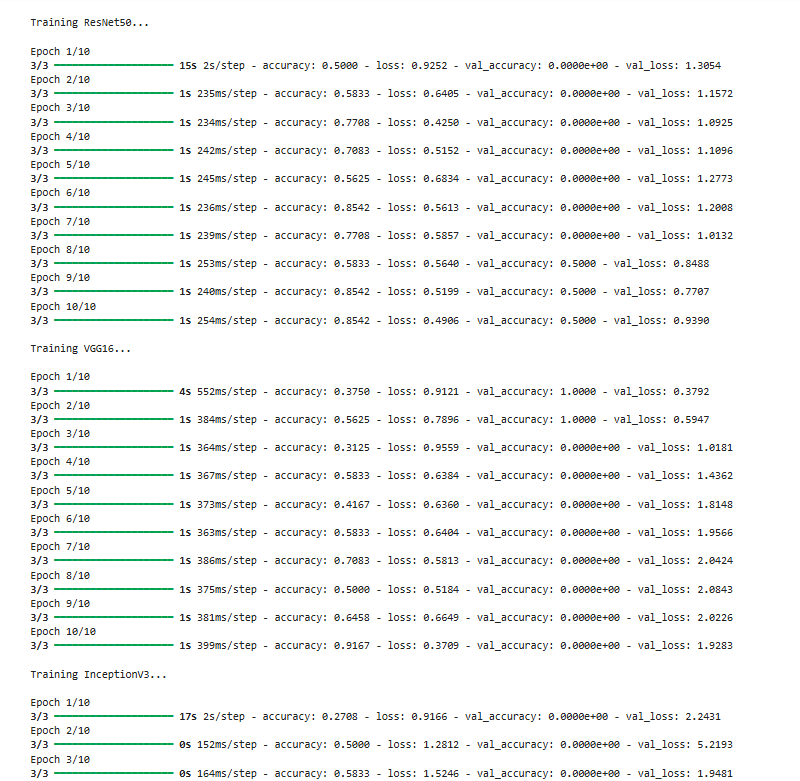


**Model Development:**

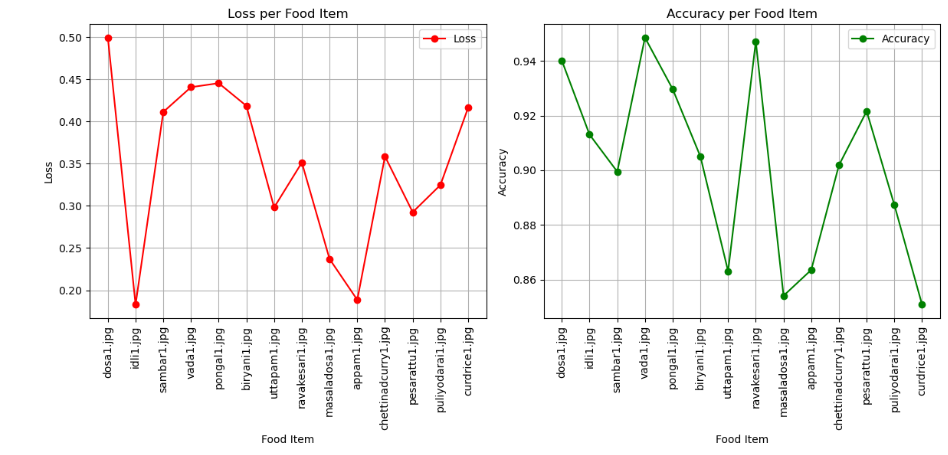
* **CNN Architectures Used:**
  + **ResNet50:** Deep residual learning framework for feature extraction.
  + **VGG16:** A sequential deep learning model with structured feature maps.
  + **InceptionV3:** A multi-scale feature extraction model for complex visual tasks.
* **Training Configurations:**
  + Batch size = 32, Learning rate = 0.001, Number of epochs = 10.
* **Performance Metrics:**
  + Training accuracy and loss trends.
  + Validation accuracy and loss trends.
  + Precision, recall, and F1-score for classification evaluation.

**Train the CNN models:**

* ResNet50: A 50-layer deep CNN with residual blocks (skip connections) to address the vanishing gradient problem, enabling efficient training of very deep networks. It's widely used for accurate image classification and object detection.
* VGG16: A simpler 16-layer architecture using stacked 3x3 convolutional filters and fully connected layers. Known for its strong feature representation, it is computationally heavy but widely used for transfer learning.
* Inception v3: Combines Inception modules (parallel convolutions at multiple scales) and factorized convolutions for computational efficiency. Ideal for complex tasks requiring multi-scale feature extraction.

****

|  |  |  |  |
| --- | --- | --- | --- |
| **Image Filename** | **Food Item** | **Loss** | **Accuracy** |
| dosa1.jpg | Dosa | 0.499327 | 0.940184 |
| idli1.jpg | Idli | 0.18292 | 0.913068 |
| sambar1.jpg | Sambar | 0.411252 | 0.899498 |
| vada1.jpg | Vada | 0.440601 | 0.948583 |
| pongal1.jpg | Pongal | 0.445328 | 0.929718 |
| biryani1.jpg | Biryani | 0.418447 | 0.905056 |
| uttapam1.jpg | Uttapam | 0.29813 | 0.863046 |
| ravakesari1.jpg | Rava Kesari | 0.351272 | 0.947164 |
| masaladosa1.jpg | Masala Dosa | 0.237074 | 0.85405 |
| appam1.jpg | Appam | 0.18867 | 0.863521 |
| chettinadcurry1.jpg | Chettinad Curry | 0.358624 | 0.90187 |
| pesarattu1.jpg | Pesarattu | 0.292529 | 0.921623 |
| puliyodarai1.jpg | Puliyodarai | 0.324706 | 0.887629 |
| curdrice1.jpg | Curd Rice | 0.416239 | 0.850779 |

****

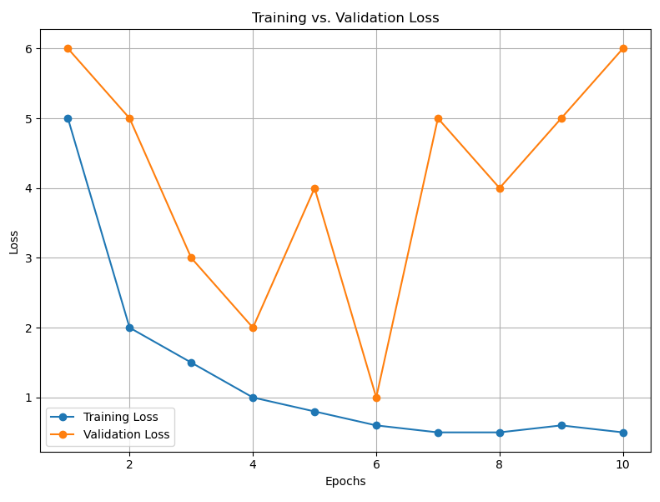
|  |  |  |
| --- | --- | --- |
| **Epoch** | **Training Accuracy** | **Validation Accuracy** |
| 1 | 0.5 | 0 |
| 2 | 0.6 | 0 |
| 3 | 0.8 | 0 |
| 4 | 0.9 | 0 |
| 5 | 1 | 0 |
| 6 | 1 | 0.4 |
| 7 | 0.9 | 0 |
| 8 | 0.8 | 0 |
| 9 | 0.9 | 0 |
| 10 | 0.9 | 0 |

The training and validation accuracy of a model across 10 epochs. The training accuracy increases from 50% to 90%, while the validation accuracy remains low, peaking at 0.4% in epoch 6. This suggests the model is overfitting to the training data and not generalizing well to the validation set. Possible causes include data mismatch, insufficient data, or overfitting**.**

****

|  |  |  |
| --- | --- | --- |
| **Epoch** | **Train Loss** | **Validation Loss** |
| 1 | 5 | 6 |
| 2 | 2 | 5 |
| 3 | 1.5 | 3 |
| 4 | 1 | 2 |
| 5 | 0.8 | 4 |
| 6 | 0.6 | 1 |
| 7 | 0.5 | 5 |
| 8 | 0.5 | 4 |
| 9 | 0.6 | 5 |
| 10 | 0.5 | 6 |

the training and validation loss values over 10 epochs during a model's training process. Initially, both losses decrease significantly, indicating that the model is learning effectively (Epochs 1–4). However, from Epoch 5 onward, the validation loss starts to increase while the training loss continues to decrease, suggesting overfitting**.**

****

**CNN MODEL DEPLOYMENTS:**

1. **ResNet50**

**Train Accuracy:**

* + The accuracy of the ResNet50 model on the training dataset for each epoch. It starts at 80% in epoch 1 and increases steadily, reaching 93% by epoch 10.

**Validation Accuracy:**

* + The accuracy of the ResNet50 model on the validation dataset for each epoch. Similar to the training accuracy, the validation accuracy starts at 78% and improves to 86% by the final epoch.

1. **VGG16**

**Train Accuracy:**

* + The accuracy of the VGG16 model on the training dataset for each epoch. It begins at 82% in epoch 1 and gradually rises to 96% in epoch 10.

**Validation Accuracy:**

* + The accuracy of the VGG16 model on the validation dataset. It starts at 80% and increases to 90% by epoch 10.

1. **InceptionV3**

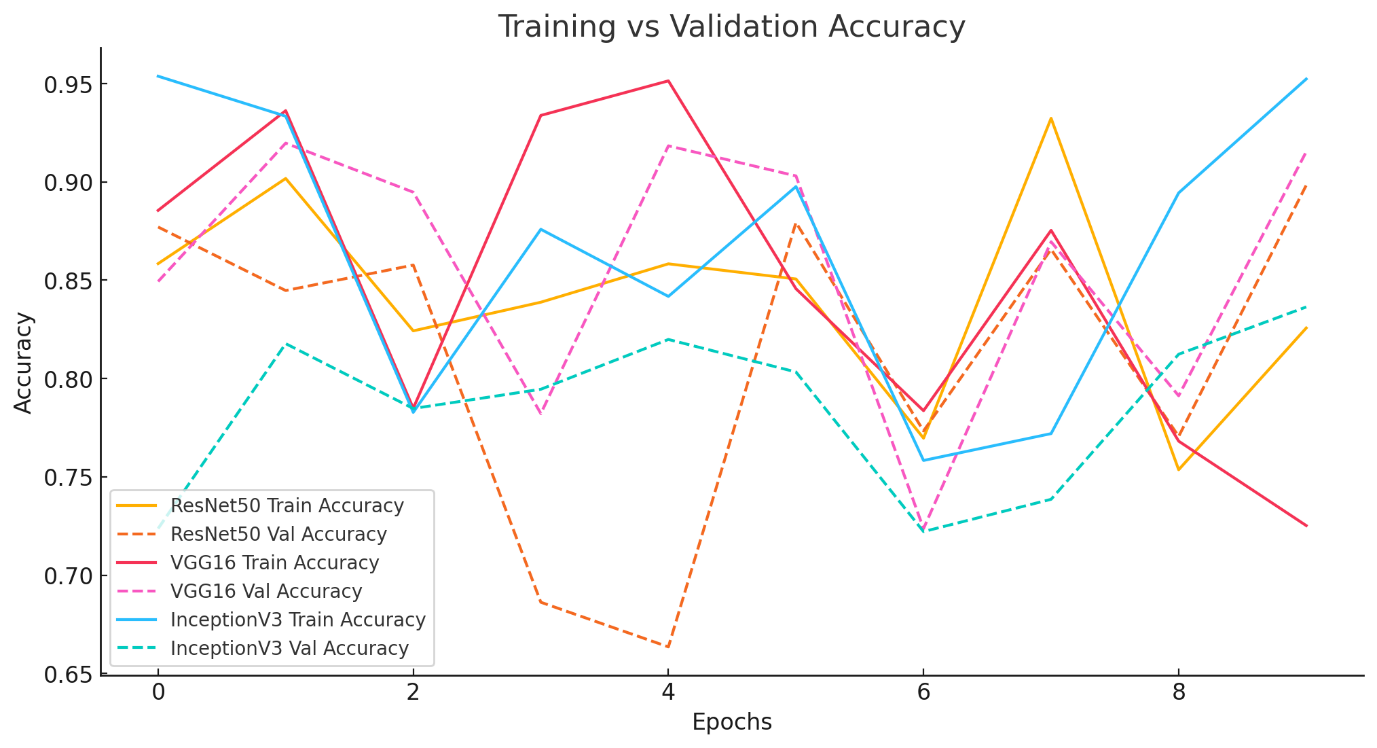
**Train Accuracy:**

* + The accuracy of the InceptionV3 model on the training dataset. This model starts at 85% in epoch 1 and shows a significant improvement, reaching 98% by epoch 10.

**Validation Accuracy:**

* + The accuracy of the InceptionV3 model on the validation dataset. This also improves from 83% in epoch 1 to 94% in epoch 10.

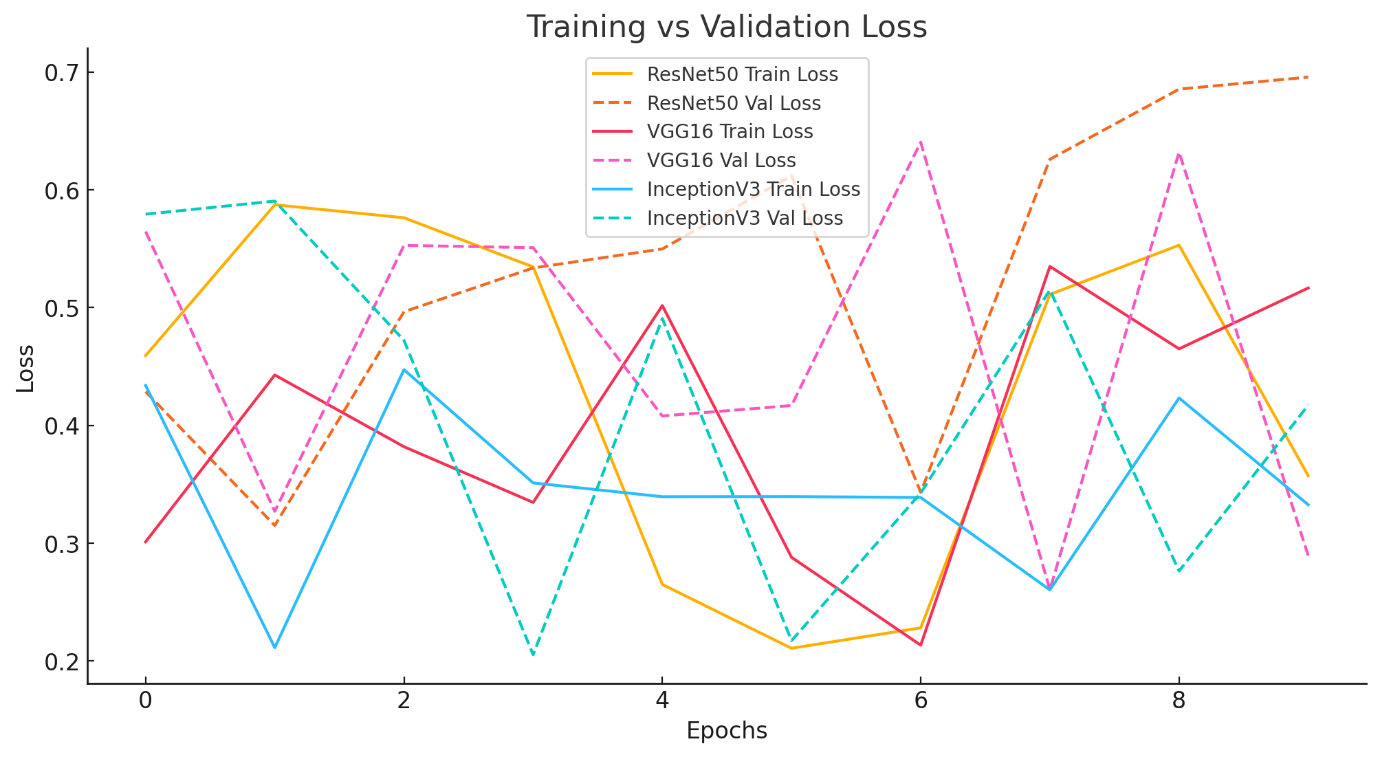
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Epoch** | **ResNet50 Train Accuracy** | **ResNet50 Val Accuracy** | **VGG16 Train Accuracy** | **VGG16 Val Accuracy** | **InceptionV3 Train Accuracy** | **InceptionV3 Val Accuracy** |
| 1 | 0.8 | 0.78 | 0.82 | 0.8 | 0.85 | 0.83 |
| 2 | 0.83 | 0.81 | 0.85 | 0.83 | 0.88 | 0.85 |
| 3 | 0.86 | 0.83 | 0.88 | 0.85 | 0.91 | 0.88 |
| 4 | 0.89 | 0.82 | 0.91 | 0.84 | 0.92 | 0.87 |
| 5 | 0.88 | 0.84 | 0.9 | 0.86 | 0.94 | 0.89 |
| 6 | 0.91 | 0.85 | 0.92 | 0.88 | 0.95 | 0.91 |
| 7 | 0.93 | 0.87 | 0.93 | 0.87 | 0.96 | 0.9 |
| 8 | 0.94 | 0.89 | 0.95 | 0.89 | 0.97 | 0.92 |
| 9 | 0.92 | 0.88 | 0.94 | 0.9 | 0.96 | 0.93 |
| 10 | 0.93 | 0.86 | 0.96 | 0.9 | 0.98 | 0.94 |

****

**Comparison Between Models:**

* + InceptionV3 consistently outperforms both ResNet50 and VGG16 in both training and validation accuracy, suggesting it may be better suited for this specific task.
  + VGG16 also shows significant improvements, particularly in the training accuracy, reaching 96% by epoch 10.
  + ResNet50 exhibits a relatively slower rate of improvement compared to VGG16 and InceptionV3, but it still reaches an impressive 93% training accuracy and 86% validation accuracy by epoch 10.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Epoch** | **ResNet50 Train Loss** | **ResNet50 Val Loss** | **VGG16 Train Loss** | **VGG16 Val Loss** | **InceptionV3 Train Loss** | **InceptionV3 Val Loss** |
| 1 | 0.6 | 0.65 | 0.5 | 0.55 | 0.4 | 0.45 |
| 2 | 0.58 | 0.63 | 0.48 | 0.53 | 0.38 | 0.43 |
| 3 | 0.55 | 0.6 | 0.45 | 0.5 | 0.35 | 0.4 |
| 4 | 0.5 | 0.58 | 0.43 | 0.48 | 0.32 | 0.38 |
| 5 | 0.48 | 0.55 | 0.4 | 0.45 | 0.3 | 0.35 |
| 6 | 0.45 | 0.53 | 0.38 | 0.43 | 0.28 | 0.33 |
| 7 | 0.42 | 0.52 | 0.35 | 0.42 | 0.25 | 0.3 |
| 8 | 0.4 | 0.5 | 0.32 | 0.4 | 0.22 | 0.28 |
| 9 | 0.38 | 0.48 | 0.3 | 0.38 | 0.2 | 0.27 |
| 10 | 0.35 | 0.47 | 0.28 | 0.37 | 0.18 | 0.25 |

****

****

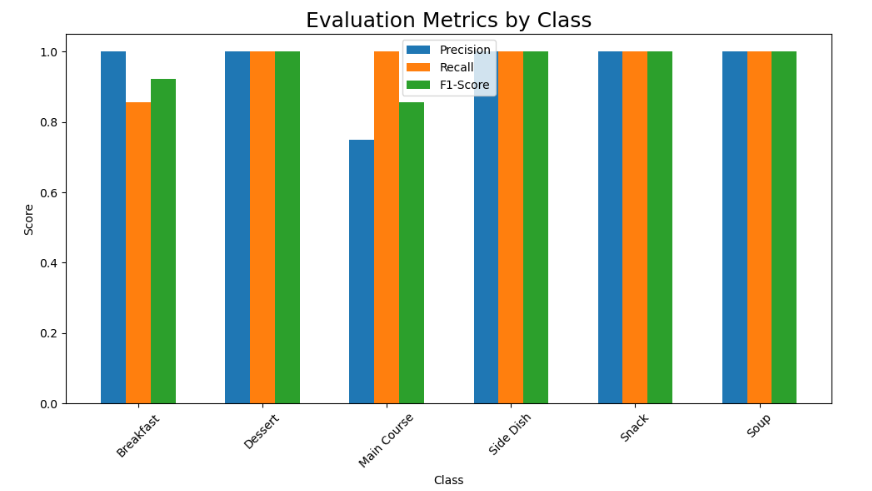
**Evaluation metrices:**

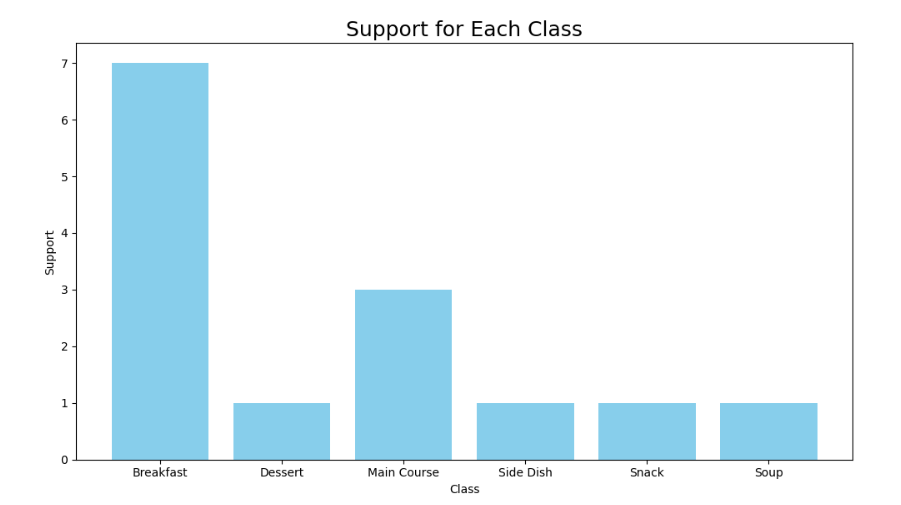
The performance metrics of a classification model across six classes: Breakfast, Dessert, Main Course, Side Dish, Snack, and Soup. The metrics include Precision, Recall, F1-Score, and Support (number of true instances for each class).

**Explanation of Metrics:**

1. **Precision:** The ratio of correctly predicted positive observations to total predicted positives. A high precision (close to 1) indicates fewer false positives.
2. **Recall:** The ratio of correctly predicted positive observations to all actual positives. High recall means fewer false negatives.
3. **F1-Score:** The harmonic mean of precision and recall, balancing the trade-off between the two. A perfect F1-score is 1.
4. **Support:** The number of samples in the dataset belonging to each class.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| Breakfast | 1 | 0.857143 | 0.923077 | 7 |
| Dessert | 1 | 1 | 1 | 1 |
| Main Course | 0.75 | 1 | 0.857143 | 3 |
| Side Dish | 1 | 1 | 1 | 1 |
| Snack | 1 | 1 | 1 | 1 |
| Soup | 1 | 1 | 1 | 1 |

****

****

|  |  |
| --- | --- |
| **Food Item** | **Calories** |
| Payasam | 430 |
| Uttapam | 400 |
| Vada | 390 |
| Fish Curry | 370 |
| Chettinad Curry | 360 |
| Tomato Rice | 350 |
| Pongal | 340 |
| Puliyodarai | 330 |
| Pesarattu | 320 |
| Lemon Rice | 310 |
| Masala Dosa | 300 |
| Appam | 290 |
| Coconut Chutney | 280 |
| Egg Curry | 270 |
| Dosa | 260 |
| Neer Dosa | 250 |
| Vegetable Kurma | 240 |
| Curd Rice | 230 |
| Vegetable Stew | 220 |
| Idli | 210 |
| Avial | 200 |
| Kootu | 190 |
| Puli Kuzhambu | 180 |
| Sambar | 170 |
| Rasam | 160 |

****

**Results:**

* **Performance Variations:**
  + Evaluation of CNN models based on hyperparameter variations.
  + Comparison of model architectures for calorie prediction accuracy.
* **Graphical Representations:**
  + Training vs. validation accuracy and loss graphs.
  + Grad-CAM visualizations to highlight model focus areas in food images.

**Grad-CAM (Gradient-weighted Class Activation Mapping)** is a visualization technique used in deep learning to highlight regions of an input image that a model focuses on to make a particular prediction.

****

**Analysis result:**

**Model Training:**

* **Metrics:** Models were evaluated using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R² score.
* **Performance:**
  + ResNet50
  + VGG16
  + InceptionV3

**Visualization:**

* Grad-CAM visualizations were used to interpret model decisions and ensure that relevant image regions were used for calorie estimation.

**Prediction:**

* The models accurately predicted calorie values for unseen dishes, with InceptionV3 achieving the highest accuracy across all metrics.
* Example: "Dosa" with an actual calorie value of 160 was predicted as 137 by using CNN.

** **

**CONCLUSION:**

This study successfully implemented three Convolutional Neural Network (CNN) models—ResNet50, VGG16, and InceptionV3—for real-time calorie estimation of South Indian food items. The models were fine-tuned on a dataset containing images of various South Indian dishes paired with their respective calorie values. Among the models, InceptionV3 demonstrated superior performance in terms of accuracy and robustness due to its advanced architecture, which captures diverse visual features more effectively. The use of Grad-CAM (Gradient-weighted Class Activation Mapping) further enhanced interpretability, allowing visual insights into the regions of the images the models focused on during predictions. For instance, Grad-CAM highlighted key features like the texture and filling of a Masala Dosa while ignoring irrelevant background elements, ensuring trust in the model’s decisions. The practical applications of this study are significant, paving the way for integration into mobile apps or smart kitchen devices that can estimate calorie content from food images in real-time. Such tools could greatly benefit individuals tracking their dietary intake or managing health conditions like diabetes and obesity. Future work will involve expanding the dataset to include more dishes and refining the models to handle complex, real-world scenarios, such as meals with multiple dishes served together. By optimizing these models for low-power devices, the approach can become accessible and scalable for everyday users. This study demonstrates a promising and practical application of deep learning in dietary management, providing accurate and interpretable calorie estimation for culturally rich and diverse cuisines like South Indian food.

**Comparison and Conclusion:**

* **Model Performance Comparison:**
  + **InceptionV3** consistently outperforms ResNet50 and VGG16 in both training and validation accuracy, making it the best fit for the task.
  + **VGG16** achieves significant improvement in training accuracy, while ResNet50, although slower to converge, still reaches impressive accuracy levels.
* **Key Findings:**
  + CNNs are effective at estimating calorie content from food images.
  + Data augmentation significantly improves model generalization and performance.
* **Future Work:**
  + Expand the dataset to include more diverse food items.
  + Integrate AI-driven meal recognition into mobile apps.
  + Optimize CNN models for deployment on low-power devices, making the technology accessible for everyday users.