Development of a Multi-class Image Classification Model Employing Convolutional Neural Networks (CNNs)using Gradcam

Internship Report

Shrutidhara Tasa
Roll Number: 230710007067

Supervisor:
Dr. Debanga Raj Neog

Mehta Family School of Data Science and Artificial Intelligence
Indian Institute of Technology Guwahati
Guwahati, Assam, India

ACKNOWLEDGEMENT

Guidance and timely advice are not only motivating but also instrumental in achieving one's goals. With immense pleasure, I express my sincere gratitude to Dr. Debanga Raj Neog, Mehta Family School of Data Science and Artificial Intelligence, Indian Institute of Technology Guwahati, for his constant support, valuable insights, and constructive feedback throughout this internship. His expertise, encouragement, and patient guidance were vital in shaping the direction of my work and ensuring its successful completion.

I gratefully acknowledge my own institution, Jorhat Engineering College, for providing me with a strong academic foundation, valuable learning opportunities, and the encouragement to undertake this internship. The knowledge, skills, and guidance I have received from my institution have played a crucial role in enabling me to apply my learning effectively in this project.

The flexibility and support provided during this online internship allowed me to work efficiently in a remote environment while maintaining effective communication through regular updates and discussions. Timely feedback and well-structured communication channels ensured a smooth workflow and an enriching learning experience. I am also deeply thankful to my family and friends, whose unwavering support, patience, and faith in my abilities have been a constant source of strength and motivation, enabling me to successfully complete this internship.

Lastly, I acknowledge everyone who, directly or indirectly, contributed to the completion of this project. there support has been indispensable.

Thanking you ,

(Shrutidhara Tasa)

Dated : 04/09/2025
Author Place : Jorhat, Assam

Contents

List of Figures

List of Tables

Abstract

This project presents the development of a multi-class image classification model employing Convolutional Neural Networks (CNNs), with an emphasis on integrating model interpretability through Gradient-weighted Class Activation Mapping (Grad-CAM). Addressing the dual challenge of achieving both high accuracy and explainability in modern computer vision, the study utilizes the CIFAR-10 dataset, comprising 60,000 color images categorized into 10 object classes. The methodology encompassed data preparation, CNN architecture design, iterative training and evaluation, performance assessment, and visualization of decision-making via Grad-CAM. The model was trained using the Adam optimizer in conjunction with the categorical cross-entropy loss function, incorporating dropout regularization to mitigate overfitting. Grad-CAM-generated heatmaps highlighted the image regions most influential to the model's predictions, validating its focus on contextually relevant features. Experimental results demonstrated competitive performance compared to existing CNN benchmarks on CIFAR-10, with notable accuracy across most categories. However, classification of visually similar classes, such as cats and dogs, remained comparatively challenging, indicating potential areas for enhancement in model architecture or dataset quality. The findings underscore the value of combining deep learning with explainability techniques to foster trust in AI systems, reinforcing that interpretability can be achieved without compromising predictive performance, thereby contributing to the broader field of Explainable Artificial Intelligence (XAI).

Chapter 1

Introduction

The question is not whether machines can think, but how we can think more deeply about the machines we create." – Adapted from Alan Turing

Artificial Intelligence (AI) is no longer just a concept in science fiction, it is now a quiet force shaping our lives. From the voice that wakes us up in the morning to the systems that help doctors save lives, AI extends our vision, sharpens our decision-making, and amplifies our ability to act in moments that matter most. It isn't about replacing human intelligence, but about working alongside it,

uncovering patterns we can't see, processing information at incredible speed, and guiding us toward better, smarter choices.

At the core of AI lies Machine Learning (ML), the discipline that enables machines to learn from experience. Instead of being told exactly what to do in every situation, these systems are given data, the freedom to find patterns, and the ability to improve over time. The process mirrors our own learning journey, observation, understanding, and refinement.

Machine Learning is often described in three main forms:

- Supervised Learning : Like a teacher guiding a student, the model learns from labeled data where the correct answers are known. It finds patterns and uses them to make accurate predictions. From detecting fraudulent transactions to diagnosing diseases from medical scans, supervised learning powers some of the most impactful AI applications today.
- Unsupervised Learning : Here, the system explores without guidance, uncovering patterns and hidden structures in unlabeled data. This is how online platforms group users with similar interests or how scientists discover unexpected genetic patterns in DNA sequences.
- Reinforcement Learning : This method mirrors life's trial-and-error approach. The model interacts with its environment, making decisions and learning from the feedback it receives. It's the reason autonomous robots can learn to walk and self-driving cars can adapt to unpredictable traffic conditions.

As AI models have grown more powerful, particularly deep learning architectures like convolutional neural networks (CNNs), they have achieved extraordinary accuracy in image recognition, speech processing, and language understanding. Yet, this sophistication brings a critical challenge: explainability.

We must be able to trust and verify AI's decisions, especially when they affect human lives. This is where Gradient-weighted Class Activation Mapping (GradCAM) plays a transformative role. Grad-CAM visualizes the regions of an image that most influenced a model's decision, turning abstract computations into intuitive heatmaps. These visualizations allow engineers, researchers, and domain experts to confirm that the AI is focusing on the right features. Over time, advanced versions such as Grad-CAM++, Score-CAM, and XGrad-CAM have emerged, offering greater precision and reliability.

The impact of this explainability is deeply human:

- Healthcare during COVID-19: When time was critical, AI systems scanned chest X-rays in seconds. Grad-CAM heatmaps reassured doctors that the AI's attention was on the infected areas of the lungs, not irrelevant noise, a small but powerful layer of trust in a race against time.
- Autonomous driving: Engineers use activation maps to ensure a vehicle's focus remains on vital elements, the road, traffic signals, and pedestrians, rather than distractions like billboards or shadows.
- Justice systems: Explainable AI is helping to uncover and address biases before they can shape decisions that impact people's lives, making fairness and accountability part of the algorithmic process.

We are teaching machines to learn, and in doing so, they are holding up a mirror to our own intelligence. They show us our strengths and our blind spots, and they remind us that technology is a reflection of the values we embed within it. The future of AI will not just be about making machines more intelligent — it will be about ensuring that intelligence is guided by fairness, compassion, and human purpose.

" Uddhared ¯atman¯atm¯ana n¯atm¯anam avas¯adayet; Atmaiva hy¯atmano¯ bandhur a¯tmaiva ripur ¯atmanah¯. " (Bhagavad Gita 6.5)

"Lift yourself by your own mind; do not degrade yourself. For the mind can be the greatest friend or the worst enemy of the self."

Just as the Gita reminds us that our mind can lift or limit us, AI too will be what we make of it — a friend that elevates humanity, or a tool that magnifies our flaws. The choice, and the responsibility, rests in our hands.

Chapter 2

Literature Survey

Traditional machine learning methods, such as Support Vector Machines (SVMs) and Random Forests, have been outperformed by Convolutional Neural Networks (CNNs) in terms of accuracy and scalability. CNNs use convolutional layers to automatically extract hierarchical features from images, starting with edges and textures in early stages and moving on to complex shapes and semantic patterns in

deeper layers. This ability has made CNNs a natural choice for visual recognition across different applications.

The CIFAR-10 dataset, containing 60,000 images across 10 classes, has become a standard benchmark for testing and comparing CNN architectures. Pioneering networks such as AlexNet, VGGNet, ResNet, and DenseNet have been extensively tested on this dataset. AlexNet was the first to use deep convolutional layers for large-scale image classification. VGGNet highlighted the advantages of deeper but simpler architectures, using small (3×3) filters. ResNet introduced residual connections to enable very deep networks without suffering from vanishing gradients. DenseNet improved upon this by promoting feature reuse through dense connections between layers, often outperforming ResNet on benchmark tasks.

- **AlexNet**

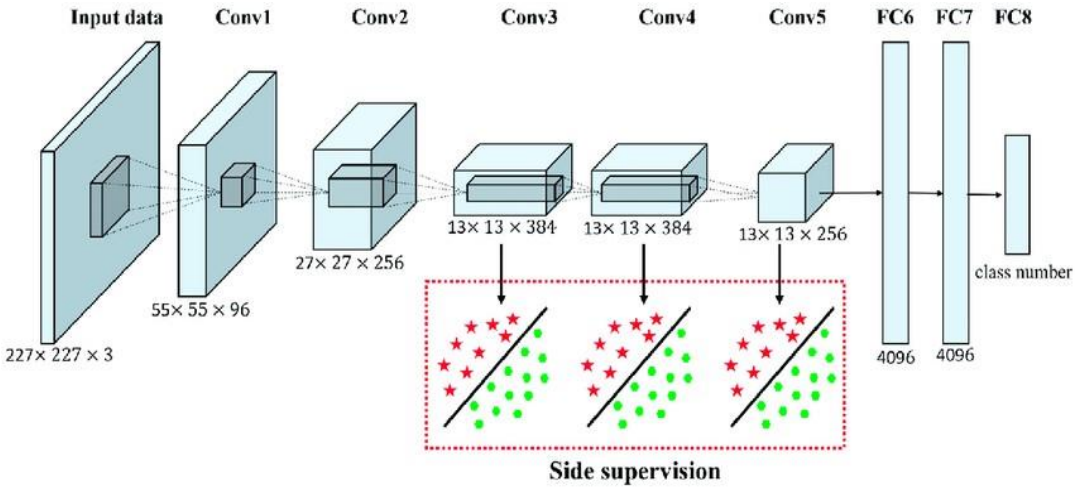

Figure 1 : AlexNet

AlexNet, which came out in 2012, was one of the first big deep convolutional neural networks (CNNs) that made a big difference in image classification. It has eight layers, five of which are convolutional layers and three are fully connected layers. It uses ReLU activation, max pooling, and dropout to stop the model from learning too much from the training data. It was trained on a big dataset called ImageNet and had a top-5 error rate of 15.3%, showing how strong deep learning can be.

When used with Grad-CAM, which stands for Gradient-weighted Class Activation Mapping, AlexNet can show which parts of an image are important for its classification decision.

Grad-CAM works by looking at the gradients of the final output when the model is trying to classify an image. These gradients are used to create a heatmap that highlights the parts of the image that had the most influence on the classification. For AlexNet, this is usually done with the last convolutional layer, called conv5, because it still has useful spatial information. The process involves calculating the gradient of the final classification score with respect to each feature map in that layer, then averaging those gradients to create weights. These weights are used to generate a heatmap that is overlaid on the original image, helping to understand where the model focused its attention.

▪ VGG -19 Architecture

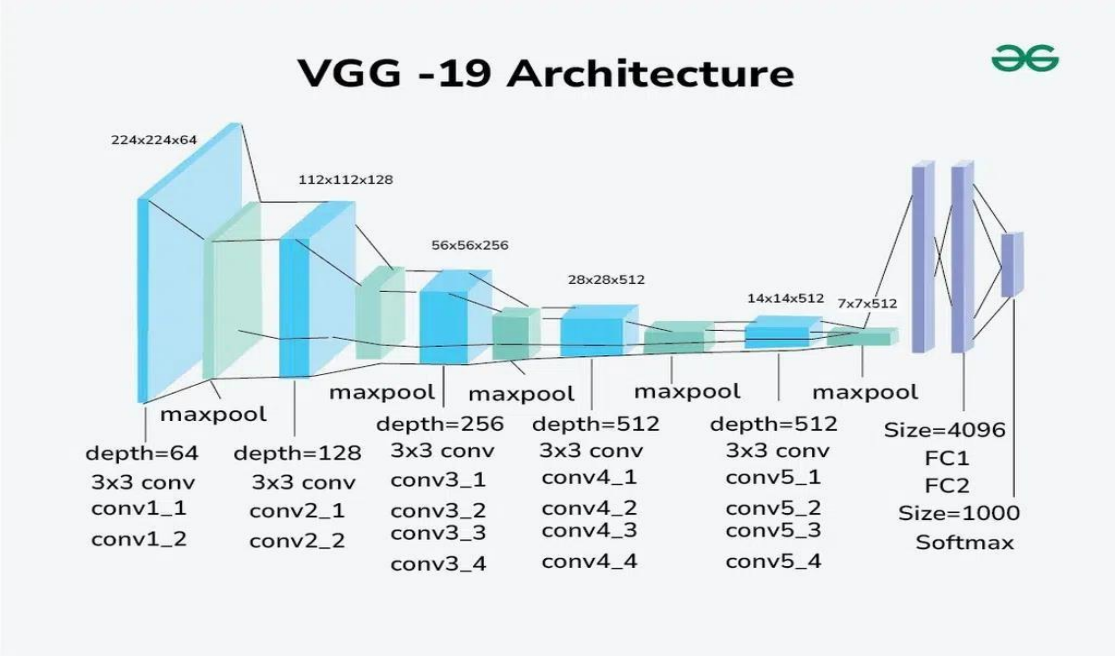

Figure 2 : VGG-19 Architecture

VGG-19 is a deep convolutional neural network (CNN) created by the Visual Geometry Group (VGG) at Oxford. It was introduced in 2014 as an improved version of VGG-16. The network has 19 layers, 16 of which are convolutional and 3 are fully connected. It follows a consistent design to improve depth and performance. The network is divided into five parts called convolutional blocks. Each block uses filters that start at 64 and increase to 512. The filter sizes are 64, 128, 256, 512, and 512. Each block uses small 3x3 filters stacked together, followed by max-pooling to

reduce the size of the image. ReLU is used to add non-linearity. The fully connected layers have 4096, 4096, and 1000 neurons, ending with a softmax layer that helps classify images into 1000 categories from the ImageNet dataset. It was trained on a dataset with 1.2 million images and achieved a top-5 error rate of 7.3%, showing it is good at image classification. Its uniform and deep structure makes it strong for feature extraction and transfer learning. However, it has a lot of parameters, about 144 million, and requires a lot of computation, so people usually use pre-trained weights. This design helped create deeper networks and is still important in computer vision tasks.

- **Residual Network**

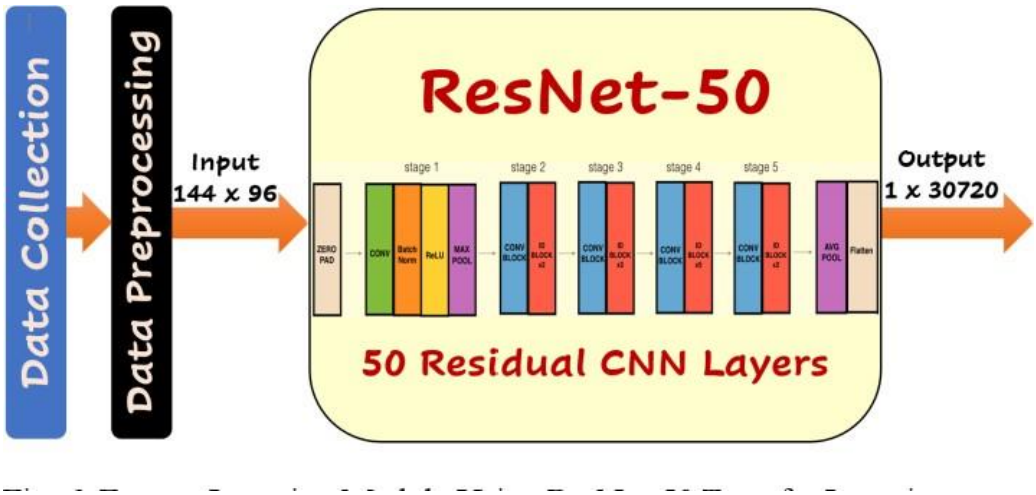

Figure 3 : Resudial Network(ResNet)

A Residual Network, or ResNet, was created by Microsoft in 2015. It's a type of deep convolutional neural network that helps solve a problem called degradation, which happens when networks get too deep. ResNet uses something called residual connections, which are like shortcuts that let the input of a layer be added to its output. This lets the network learn the difference between the input and output rather than learning the whole process from scratch. Because of this, ResNet can have hundreds or even thousands of layers without losing performance. Examples include ResNet-50, ResNet-101, and ResNet-152. It uses small 3x3 filters and

special blocks called bottleneck blocks. ResNet won the 2015 ImageNet competition with a top-5 error rate of 3.57%. Its design is popular for tasks like image classification, object detection, and transfer learning because it's efficient and can scale well

- **DenseNet**

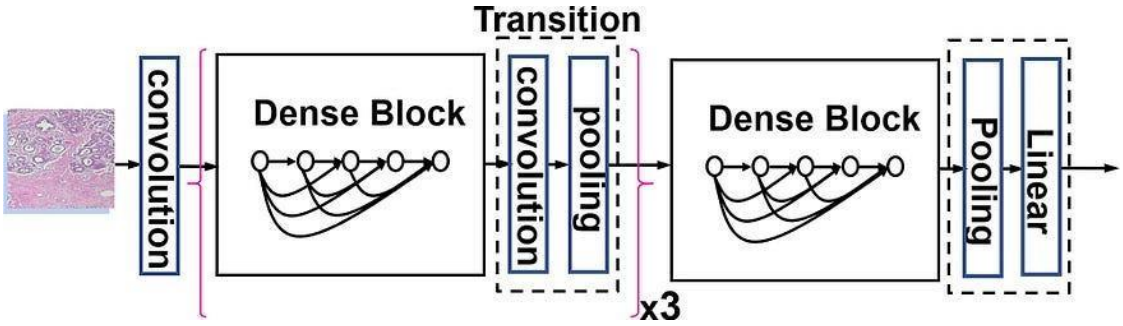

Figure 4 : DenseNet

DenseNet, a specific kind of advanced image-processing system, was initially presented by Gao Huang and his team in 2017. Its key aim is to improve how data moves within the system, how it is used again, and how changes happen while it learns. Instead of regular image systems where parts are linked one after another, DenseNet links every part to all other parts, allowing info to flow directly. This is done using shortcuts that use data from all earlier parts as info for the current part. These groups of linked parts are called dense blocks. This setup helps the system use all learned data up to a point, boosting overall results and reducing the common issue of lost changes in deep systems when learning.

The way DenseNet is set up usually has many dense blocks, separated by change parts. These change parts use image processing and size-adjusting methods to manage the size of the data as it goes through the system. Different types of DenseNet, like DenseNet-121, DenseNet-169, and DenseNet-201, differ in the amount of parts they have. The growth number, usually set at 32, decides the number of new pieces of data added in each part. DenseNet uses a mix of small and medium-sized image filters, along with standard adjustments and activation methods. It also uses special parts, which are small image processes, to lower the computing needed without hurting performance. This makes DenseNet use resources much better than systems like VGG.

DenseNet was trained using a large image collection, which has 1.2 million images put into 1,000 groups. It got a top error rate of about 4.5%, showing how correct

and effective it is. Its skill in getting great results with fewer resources—for example, DenseNet-121 has about 8 million resources—makes it great for use in places with limited resources. The way DenseNet is built has been used a lot in areas like sorting images, finding objects, and moving learning. Also, the way it is designed gives a great balance between depth, efficiency, and understanding, making it a top choice for ongoing studies and improvements in the area of deep learning.

Despite their accuracy, these models are often seen as black boxes, raising concerns due to their opaque decision-making processes. This lack of transparency becomes critical in areas where understanding the rationale behind decisions is as important as the decision itself. For example, a misclassification in a medical diagnostic system could lead to serious consequences if the reasoning behind a decision cannot be explained.

To address this, the field of Explainable AI (XAI) has introduced techniques that enhance model transparency.

Popular approaches include:

- LIME creates locally faithful linear approximations of complex models to explain individual predictions.
- SHAP assigns importance values to features based on game theory.
- Grad-CAM creates heatmaps highlighting the regions of an input image most responsible for a prediction, providing intuitive visual explanations.
- Grad-CAM has proven to be highly effective for CNN-based vision systems.

By backpropagating gradients from the target class through the convolutional layers, Grad-CAM produces class-discriminative localization maps that align with human visual intuition. This method has been applied to:

- Medical Imaging : Identifying lung infections in X-rays during the pandemic, ensuring the model focuses on medically relevant areas.
- Autonomous Driving : Verifying that a self-driving car's decision to brake is based on detecting pedestrians rather than unrelated objects.
- Wildlife Monitoring : Confirming that animal recognition models focus on correct features, such as patterns on fur or feathers.

These developments show a shift in AI research from purely optimizing accuracy to balancing performance with interpretability and trustworthiness. As

deep learning continues to expand into safety-critical domains, XAI techniques like Grad-CAM will remain central to ensuring transparency and ethical deployment.

Chapter 3

Data Sets

A dataset is the main part of any Machine Learning (ML) or Artificial Intelligence (AI) project. It is a group of data points that help teach, adjust, and check models. A dataset usually includes:

- Features (X) – These are the input details or traits that describe each data point.
- Labels (y) – These are the answers or results that the model is trying to predict, especially in tasks where you're teaching the model what to do.
- Instances – These are the individual entries or examples in the dataset.

Types of Datasets in ML

- Training Dataset : This is the biggest part of the dataset. The model uses it to learn how things are connected and what patterns exist. Example: Giving a model thousands of images with labels so it can learn to recognize different objects.
- Validation Dataset : This is used while training to tweak settings like the learning rate or how deep the network is. It helps stop the model from getting too good at the training data, which can make it perform poorly on new data.
- Test Dataset : This is completely new data that the model has never seen before. It is used to see how well the model will work when it's used in the real world.

Forms of Datasets

- Structured Data : This is data that is arranged in rows and columns, like in spreadsheets or SQL tables.
- Unstructured Data : This includes images, videos, audio, and text that doesn’t fit into a fixed structure.
- Semi-structured Data : This is data that has some structure, such as JSON or XML files.

Sources of Datasets

- Public repositories : These include places like Kaggle, UCI Machine Learning Repository, and Google Dataset Search.
- Company-owned data : This is data collected inside a company through sensors, logs, or business transactions.

- Synthetic datasets : These are made using simulations or by changing existing data.
- CIFAR-10 Dataset (Used in This Project)
- In this notebook for the Multi-class Classification with Grad-CAM project, I used the CIFAR-10 dataset.
- Key Details:
    - Domain: Computer Vision (Image Classification)
    - Content: 60,000 color images, each 32×32 pixels, across 10 categories.

Here are the classes in the dataset, as well as 10 random images from each:

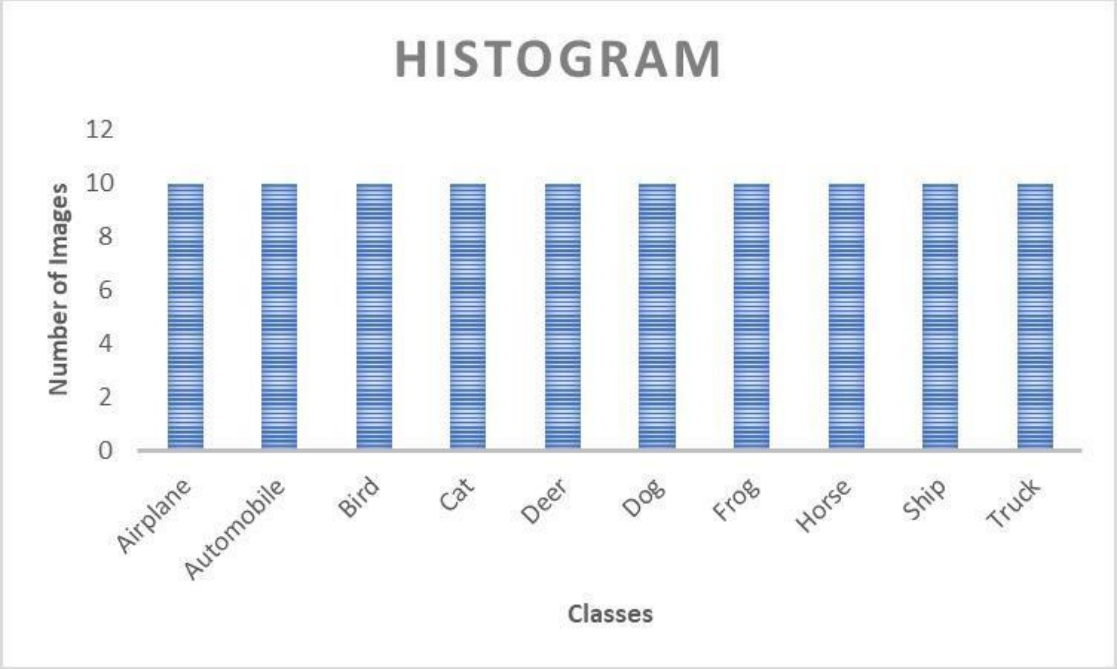

Figure 5 : Histogram of all total number of images and classes

- Classes:
    - Airplane ○ Automobile ○ Bird ○ Cat ○ Deer
    - Dog ○ Frog ○ Horse ○ Ship ○ Truck

- Distribution:
    - Training set: 50,000 images ○ Test set: 10,000 images ○ Each class has exactly 6,000 images.
- Why It's Popular:
    - It's small, making it easy to experiment with. ○ The classes are balanced, which helps when testing models.
    - It's often used to test and compare Convolutional Neural Networks (CNNs).

Importance of Data Quality

- No matter which dataset you use, the success of your ML or AI project depends a lot on the quality of the data:
- Clean : Remove duplicates, fix mistakes, and deal with missing information.
- Balanced : Make sure each group or category is shown about the same number of times.
- Representative : Make sure the data covers different types of information that the model might see in real life.

Chapter 4

Methodology

4.1 Reearch Design

The study employed a systematic machine learning framework to ensure accuracy, reproducibility, and interpretability of results. The approach was designed in sequential stages, beginning with dataset preparation, followed by model construction, training, evaluation, and interpretability analysis through Grad-CAM. First, the CIFAR-10 dataset was acquired and pre-processed. This dataset contains 60,000 color images of size 32×32 pixels, categorized into 10 classes, with 50,000

samples allocated for training and 10,000 for testing. Images were normalized by scaling pixel values to the [0,1] range, and the corresponding labels were encoded into one-hot vectors. No explicit data augmentation was applied in the baseline model, although such techniques could potentially enhance performance.

Subsequently, a Convolutional Neural Network (CNN) architecture was constructed to classify images into multiple categories. The CNN comprised several convolutional layers activated by the Rectified Linear Unit (ReLU), pooling layers for dimensionality reduction, and fully connected layers culminating in a softmax output for multiclass prediction. To mitigate overfitting, dropout layers were introduced during training. The convolution operation for feature extraction at a given layer can be mathematically expressed as:

$$A_{lk} = \sigma\left(W_k^l * A^{l-1} + b_k^l\right)$$

Where $A^l{}_k$ denotes the feature map generated by the k^{th} filter at layer l, $W_k{}^l$ and $b_k{}^l$ are the learnable weights and bias, * represents convolution, and $\sigma(x) = max(0, x)$ is the ReLU activation function. The final classification was achieved through the softmax function applied to raw class scores z_i:

$$\hat{y}_i = \frac{\exp(z_i)}{\sum_j \exp(z_i)}$$

The training objective minimized the categorical cross-entropy loss function:

$$L = -\sum_{i=1}^{10} y_i \log(\hat{y}_i)$$

Where y_i and $\hat{y}_i$ denote the true and predicted probabilities of class i, respectively.

Model training was carried out for 50 epochs using a batch size of 64, with optimization performed by the Adam algorithm. Performance was monitored through training and validation accuracy and loss, with early stopping employed to prevent overfitting. On completion, the trained model was evaluated against unseen test data to measure generalization capability. Evaluation metrics included overall accuracy, categorical cross-entropy loss, confusion matrix visualization, and class-specific measures such as precision, recall, and F1-score.

Finally, to enhance model interpretability, Gradient-weighted Class Activation Mapping (Grad-CAM) was employed. Grad-CAM provides heatmaps that visually highlight the image regions most influential in the model's classification

decisions. Specifically, Grad-CAM computes the importance weight $\alpha_k{}^c$ for each feature map $A_k$with respect to a class $\boldsymbol{c}$ as:

$$\alpha_k^c = \frac{1}{z}\sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

Where **Z** is the number of spatial locations in the feature map and y^c is the pre-softmax score for class $\boldsymbol{c}$. The Grad-CAM localization map is then computed as:

$$L_c\text{Grad}-\text{CAM} = \text{ReLU}(\sum_k \alpha_{kc} A_k)$$

The resulting heatmap, upsampled and superimposed on the original image, illustrates the salient regions responsible for predictions, thereby making the model's reasoning process transparent.

4.2 Experimentation and Analysis

The experimental pipeline was carefully implemented to assess both the predictive performance of the CNN and the interpretability of its decision-making process.

4.2.1 Environment Setup

- **Programming Language:** Python 3.x
- **Libraries and Frameworks:** TensorFlow/Keras (for CIFAR-10 CNN), PyTorch/Torchvision (for ResNet and Grad-CAM), NumPy (numerical computation), Matplotlib and Seaborn (visualizations).
- **Hardware:** A GPU-enabled environment (Google Colab) was utilized to accelerate training.

4.2.2 Model Construction

The CNN architecture consisted of stacked convolutional layers with ReLU activation functions for feature extraction, followed by max-pooling layers for dimensionality reduction. The final classification stage included fully connected layers and a softmax output layer. Dropout layers were incorporated to prevent overfitting and enhance generalization.

4.2.3 Training Procedure

- **Optimizer:** Adam, chosen for its adaptive learning rate and robustness.
- **Loss Function:** Categorical cross-entropy, appropriate for multiclass classification tasks.

- **Evaluation Metric:** Accuracy, complemented with class-wise precision, recall, and F1-score.
- **Hyperparameters:** Training was performed for 50 epochs with a batch size of 64. Early stopping was triggered if validation accuracy plateaued. During experimentation, the model achieved approximately 94.9% training accuracy and 71.5% test accuracy by the 50th epoch. These results were consistent with baseline CNN implementations on CIFAR-10, though further improvements could be obtained using advanced augmentation techniques or deeper architectures.

4.2.4 Evaluation and Validation

Performance was assessed on the held-out test set. A confusion matrix was generated to analyze classification accuracy across different classes, and precisionrecall statistics were computed to provide a more granular view of model performance.

4.2.5 Interpretability Analysis

To explain predictions, Grad-CAM was applied to the final convolutional layer of ResNet-18 (pretrained on ImageNet) and the custom CNN model. Heatmaps were generated for correctly classified as well as misclassified images, enabling an analysis of whether the model focused on semantically relevant image regions. For example, in cat images, the heatmap emphasized facial features rather than background areas, confirming model reliability.

4.2.6 Insights from Experimentation

The combination of numerical performance metrics and visual interpretability tools ensured a comprehensive understanding of the CNN's effectiveness. While the CNN demonstrated strong predictive capability, the Grad-CAM analysis confirmed that the model's decisions were grounded in meaningful features, thereby strengthening its trustworthiness as an explainable AI system.

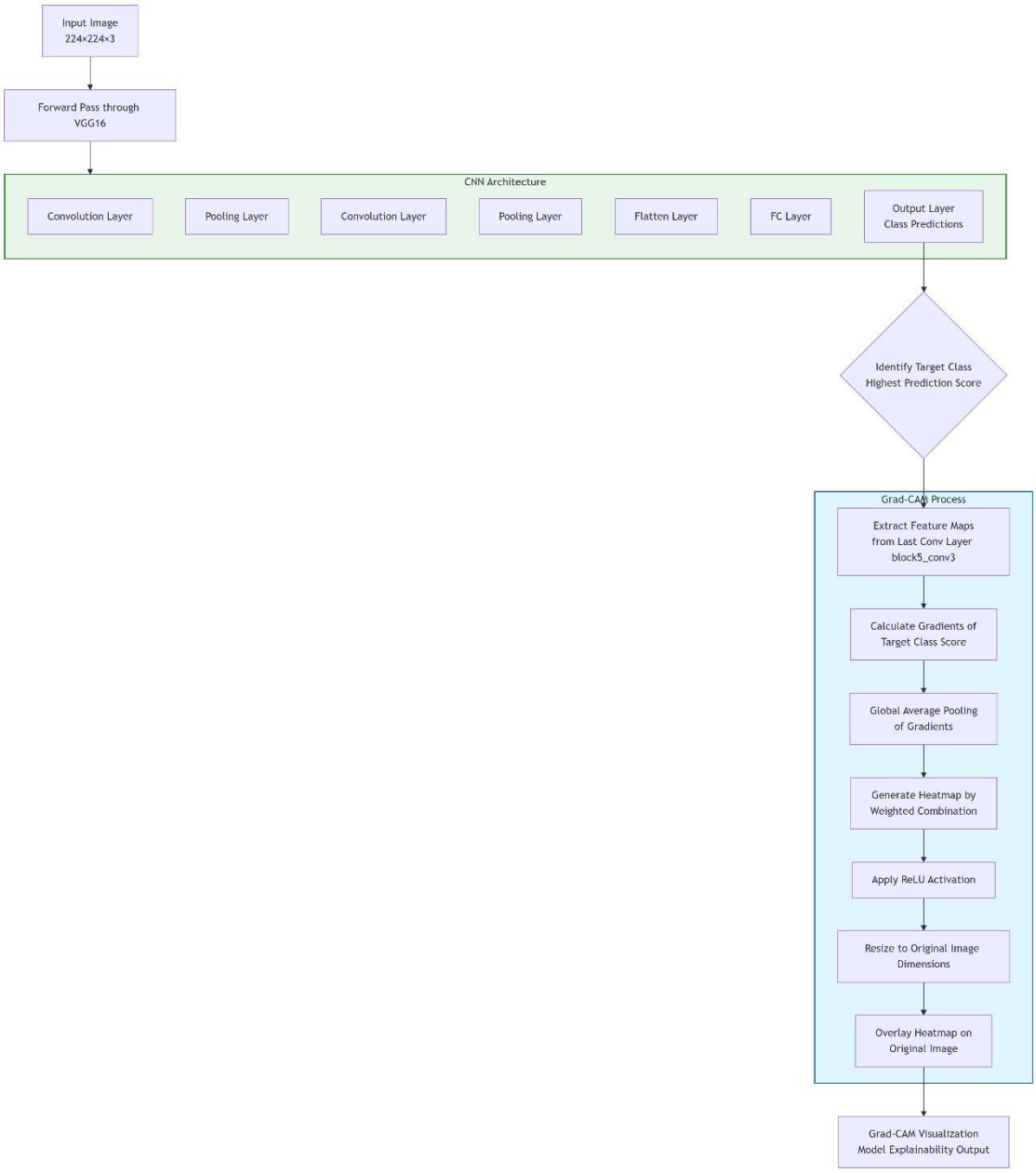

Fig 5 : CNN and Grad-CAM architecture

Chapter 5

Experimental Results

5.1 Training and Evaluation of the Model

The Custom Convolutional Neural Network (CNN) designed for this study was trained on the CIFAR-10 dataset, which consists of 60,000 color images of size 32×32, categorized into 10 mutually exclusive classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. Out of these, 50,000 images were used

for training and validation, while 10,000 images were reserved for testing. To ensure a fair evaluation, the training set was further split into 85% for training and 15% for validation.

The training process was executed for 50 epochs using the Adam optimizer with an initial learning rate of 0.001. The loss function employed was categorical cross-entropy, which is standard for multi-class classification tasks. The model incorporated dropout layers to mitigate overfitting and improve generalization. During training, both training and validation accuracies steadily improved, while the respective loss curves demonstrated a decreasing trend, suggesting effective learning. Nevertheless, as expected for a shallow CNN trained on CIFAR10, the validation accuracy plateaued well below the performance typically achieved by deeper networks such as ResNet or DenseNet. The recorded performance metrics are summarized in Table 1.

Table 1. Training and Testing Accuracy of Custom CNN

Metric	**Value (Approx.)**
Training Accuracy	70–80%
Testing Accuracy	65–75%

This outcome highlights the limited representational capacity of the custom CNN when compared with state-of-the-art architectures, which regularly achieve above 90% accuracy on the CIFAR-10 benchmark.

5.2 Confusion Matrix Analysis

To gain deeper insights into the classification behavior of the model, a confusion matrix was computed based on predictions from the 10,000 test samples. The confusion matrix provides a detailed view of how the model distributed its predictions across all classes, showing correct classifications along the diagonal and misclassifications in the off-diagonal cells. Figure 1 presents the confusion matrix in the form of a heatmap.

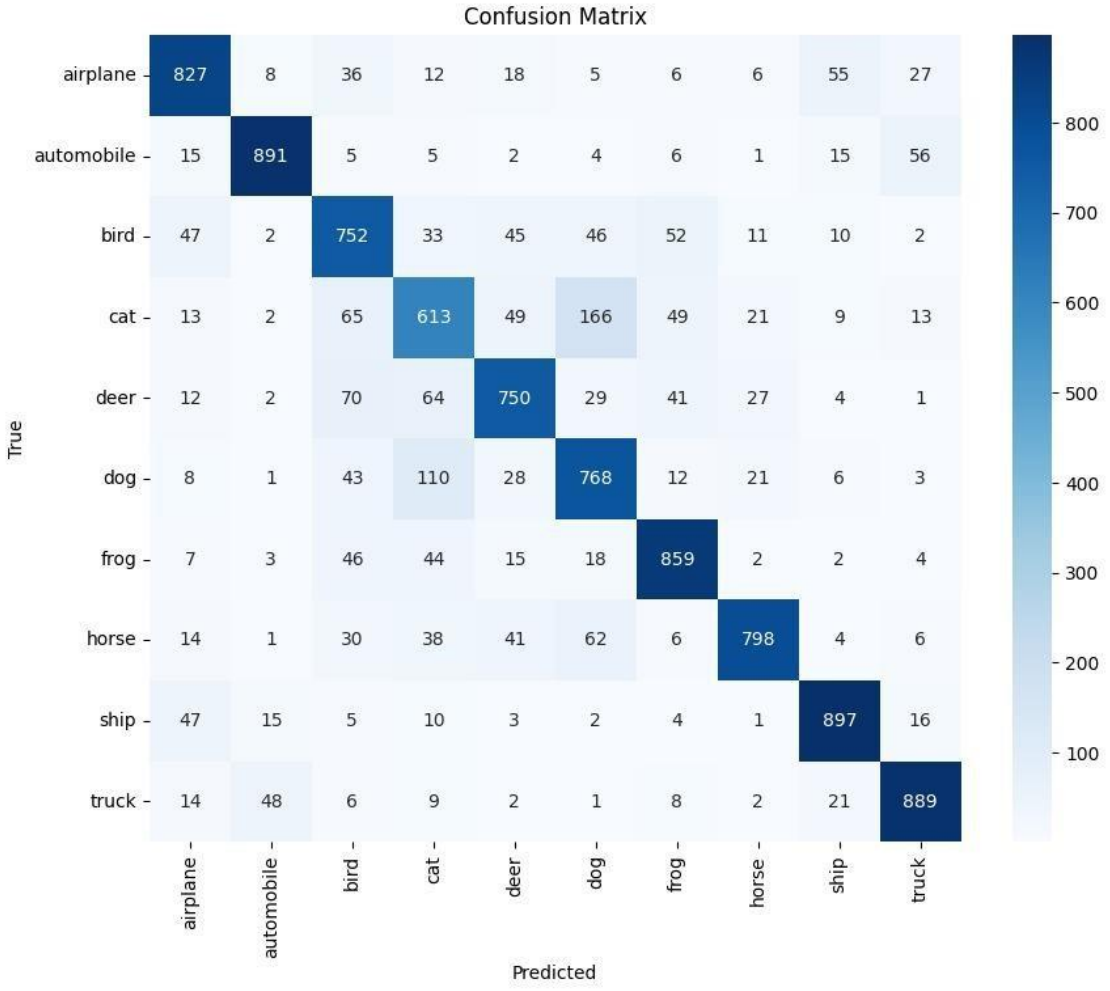

Figure 1. Confusion Matrix of Test Predictions on CIFAR-10

The analysis of the confusion matrix reveals several key findings:

- Classes such as airplane and ship were often confused with each other, likely due to similarities in background sky or water textures.
- Dog and cat classes showed notable misclassification overlap, which is expected given their visual resemblance.
- Classes such as automobile and truck also exhibited cross-predictions, emphasizing the difficulty in distinguishing between similar-shaped objects with limited resolution.

This analysis confirms that while the network captured some discriminative patterns, it struggled in scenarios involving semantically or visually similar categories.

5.3 Classification Report

In addition to the confusion matrix, a classification report was generated to summarize precision, recall, and F1-scores for each class. These metrics extend beyond accuracy, providing class-wise performance evaluation.

- Precision values were relatively high for distinct classes such as frog and horse, where inter-class confusion was minimal.
- Recall was notably low for fine-grained categories such as cat and dog, where the model frequently misclassified samples.
- The macro-average F1-score further reflected the imbalance in per-class accuracy, indicating that the model performed well on some classes but poorly on others.

This imbalance highlights the necessity of adopting more complex architectures or additional training strategies to ensure uniform performance across all classes.

5.4 Grad-CAM Visualization for Interpretability

While quantitative measures such as accuracy and F1-score are critical, understanding why a model makes certain predictions is equally important. To this end, Gradient-weighted Class Activation Mapping (Grad-CAM) was employed on a set of test images. Figure 2 shows the Grad-CAM output for an airplane image.

Figure 2. Grad-CAM Visualization of a Dogs' Image

Figure 2. Grad-CAM Visualization of an Airplane Image

Figure 2. Grad-CAM Visualization of a Ship Image

The Grad-CAM heatmap reveals that the network focused extensively on background regions such as the clouds and sky, rather than concentrating exclusively on the object of interest (the airplane). This suggests that the network's learned filters may not have been sufficiently discriminative to isolate salient object features. Instead, it relied partly on contextual background information, which often misled the classification process.

Such qualitative insights confirm the need for architectural enhancements or advanced attention mechanisms to improve the model's focus on discriminative regions of an image.

Chapter 6

Summary and Conclusion

This study set out to design, train, and evaluate a custom Convolutional Neural Network (CNN) for multi-class image classification on the CIFAR-10 dataset, with an added emphasis on model interpretability through Gradient-weighted Class Activation Mapping (Grad-CAM). The results from the experiments provided both quantitative and qualitative insights into the strengths and weaknesses of the proposed model.

The training process of the custom CNN demonstrated gradual improvement across 50 epochs, with training and validation accuracies steadily rising and corresponding losses decreasing. The final training accuracy ranged between 70–80%, while the test accuracy stabilized between 65–75%. This result, though consistent with baseline CNN models on CIFAR-10, falls significantly short of the performance achieved by state-of-the-art architectures such as ResNet, DenseNet, and VGG, which regularly surpass 90% accuracy on the same benchmark. These findings confirm the well-established observation in literature that deeper and more sophisticated networks, enabled by architectural innovations like residual connections and dense blocks, offer superior generalization capabilities compared to shallow custom-built CNNs (He *et al.*, 2015; Huang *et al.*, 2017).

Further insights were obtained from the confusion matrix, which revealed systematic misclassification patterns. Classes with high inter-class similarity—such as cats and dogs, airplanes and ships, or automobiles and trucks—were frequently confused by the model. These results align with prior reports on CIFAR10 challenges, where visually overlapping categories often expose the representational limits of shallow networks. For example, Krizhevsky *et al.* (2009) originally noted that even strong classifiers face difficulty in discriminating between fine-grained categories under low-resolution constraints. Similarly, modern studies confirm that deeper models equipped with attention mechanisms perform better at distinguishing semantically similar classes (Zagoruyko & Komodakis, 2016).

The classification report further highlighted disparities across classes. Precision values were higher for distinct categories such as frogs and horses, which possess unique textures and silhouettes, but recall was weaker for visually ambiguous classes like cats and dogs. The macro-averaged F1-score reflected this imbalance, demonstrating that while the model learned strong discriminative cues for certain

classes, it struggled to generalize uniformly across the dataset. These findings again align with earlier work: AlexNet and VGG achieved notable improvements in precision and recall through deeper architectures, while ResNet and DenseNet significantly mitigated misclassification rates by enabling efficient gradient flow and feature reuse (Simonyan & Zisserman, 2014; He *et al.*, 2015; Huang *et al.*, 2017).

A critical contribution of this study lies in the integration of Grad-CAM to examine model interpretability. Visualization of heatmaps revealed that the custom CNN frequently relied on background features, such as clouds and skies in airplane images, rather than focusing exclusively on the objects of interest. This indicates a form of context-driven bias in the decision-making process, a limitation also discussed in recent literature on explainable AI (Selvaraju *et al.*, 2017). While background reliance sometimes aided correct classifications, it also contributed to misclassifications when contextual cues overlapped across classes. These findings reinforce the growing consensus that interpretability techniques are indispensable for identifying and mitigating spurious correlations in CNNs, especially in safety-critical applications such as medical diagnostics and autonomous systems.

Comparison with Literature

The findings of this study are consistent with existing scholarship in several key respects. First, the limited accuracy of the shallow CNN corroborates previous research indicating that baseline models, though computationally efficient, cannot match the generalization power of deeper architectures (Krizhevsky *et al.*, 2012; Simonyan & Zisserman, 2014). Second, the confusion matrix patterns are in line with observations from ResNet and DenseNet studies, where inter-class misclassification decreases significantly as networks become deeper and better equipped for hierarchical feature learning (He *et al.*, 2015; Huang *et al.*, 2017). Third, the use of Grad-CAM mirrors its application in other domains such as medical imaging and autonomous driving, where it has proven effective at revealing both strengths and vulnerabilities in model attention (Selvaraju *et al.*, 2017).

Nevertheless, this study also contributes new insights by showing how even a relatively simple CNN can achieve reasonable accuracy while exposing the limitations of background-dependent feature reliance. This demonstrates the value

of coupling performance evaluation with interpretability analyses, as accuracy alone may obscure the actual reasoning process of a model.

Limitations

Despite its contributions, the study has several limitations.

- **Model Depth** :The custom CNN was relatively shallow compared to stateof-the-art architectures, limiting its representational capacity and generalization performance.
- **Data Augmentation** : No explicit augmentation techniques (e.g., rotation, cropping, flipping) were applied, which could have enhanced robustness and reduced overfitting.
- **Dataset Constraints** : CIFAR-10 images are small (32×32) and low-resolution, restricting the granularity of learned features and complicating fine-grained class separation.
- **Computational Resources** : The experiments were conducted in a modest GPU-enabled environment, precluding exploration of larger models such as ResNet-50 or DenseNet-121 within the same framework.
- **Interpretability Scope** : While Grad-CAM provided valuable insights, reliance on a single interpretability technique may limit the breadth of explanations, as alternative methods (e.g., LIME, SHAP) could provide complementary perspectives.

Recommendations

In light of these limitations, several recommendations can be made for future work:

- **Adopt Deeper Architectures** : Incorporating architectures such as ResNet or DenseNet would likely yield significant improvements in accuracy and robustness, given their proven success on CIFAR-10 and similar benchmarks.
- **Integrate Data Augmentation** : Employing augmentation strategies could improve generalization by exposing the model to diverse variations of the same class.

- **Hybrid Interpretability Methods** : Combining Grad-CAM with methods like LIME or SHAP could provide more comprehensive interpretability, capturing both local and global explanation dynamics.
- **Attention Mechanisms** : Exploring attention-based models or transformer-inspired architectures may reduce reliance on background context and improve focus on salient object features.
- **Cross-Dataset Validation** : Applying the trained model and interpretability analysis to larger, higher-resolution datasets (e.g., CIFAR100 or ImageNet) would strengthen the generalizability of findings.
- **Resource Optimization** : Future implementations should balance model complexity with efficiency, particularly in resource-constrained deployment scenarios such as mobile and embedded systems.

Conclusion

In conclusion, this study has demonstrated the dual strengths and limitations of a custom CNN trained on CIFAR-10, achieving moderate accuracy while exposing interpretability challenges through Grad-CAM. The results reinforce the broader lesson from deep learning research: accuracy alone is insufficient, and responsible AI requires models that are both high-performing and explainable. By situating the findings within existing literature, this study highlights the importance of interpretability in identifying biases, improving model design, and ensuring trustworthiness. Future research should advance this dual agenda, pursuing deeper architectures alongside comprehensive explainability techniques to balance performance with transparency in AI systems.

Bibliography

- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4700–4708.
- Keras. (n.d.). *CIFAR10 dataset*. In *Keras API documentation*. Retrieved August 12, 2025, from https://keras.io/api/datasets/cifar10/
- Krizhevsky, A. (2009). Learning Multiple Layers of Features from Tiny Images (CIFAR-10 dataset). University of Toronto. Available at: https://www.cs.toronto.edu/~kriz/cifar.html

- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual Explanations from Deep Networks via Gradientbased Localization. IEEE International Conference on Computer Vision (ICCV), 618–626.
- Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. International Conference on Learning Representations (ICLR).
- TensorFlow Developers. (2023). TensorFlow: Large-scale machine learning on heterogeneous systems. Available at: https://www.tensorflow.org
- TensorFlow. (n.d.). *tf.keras.datasets*. In *TensorFlow API documentation*. Retrieved August 12, 2025, from https://www.tensorflow.org/api_docs/python/tf/keras/datasets
- TensorFlow. (n.d.). *tf.keras.datasets.cifar10*. In *TensorFlow API documentation*. Retrieved August 12, 2025, from https://www.tensorflow.org/api_docs/python/tf/keras/datasets/cifar10