

The slide features a light gray background with several hexagonal shapes: a light blue hexagon and a dark green hexagon in the top left; a large green hexagon in the top center; and a small green hexagon in the bottom center. On the right side, there is a large, abstract graphic composed of overlapping translucent blue and teal geometric shapes. The text 'Selvaraj S' is in black, and 'Final Project' is in green, both in a sans-serif font.

Selvaraj S

Final Project

PROJECT TITLE



CIFAR-10 Image Classification using CNN



AGENDA

- >PROBLEM STATEMENT
- >PROJECT OVERVIEW
- >WHO ARE THE END USERS?
- >YOUR SOLUTION AND ITS VALUE PROPOSITION
- >MODELLING
- >RESULT



PROBLEM STATEMENT

To classify CIFAR-10 images into their respective classes using CNN.





PROJECT OVERVIEW



The project aims to develop a Convolutional Neural Network (CNN) model to classify images from the CIFAR-10 dataset. CIFAR-10 is a well-known benchmark dataset in the field of computer vision, consisting of 60,000 32x32 color images across 10 classes.



WHO ARE THE END USERS?

- 
1. Researchers and Academics
 2. Data Scientists and Machine Learning Engineers
 3. Software Developers
 4. Students and Enthusiasts
 5. Industry Professionals
- 

YOUR SOLUTION AND ITS VALUE PROPOSITION



- - >Our solution offers a robust and accessible implementation of Convolutional Neural Networks (CNNs) tailored for image classification tasks using the CIFAR-10 dataset.
 - >Designed with simplicity and clarity in mind, our code is easily understandable and modifiable, catering to users of varying expertise levels in deep learning and image processing.
 - >Overall, our solution facilitates innovation, learning, and practical impact in the field of computer vision, driving progress and accessibility in CNN-based image classification.

THE WOW IN YOUR SOLUTION

Our solution excels in its seamless integration of accessibility, performance, and versatility, making it an ideal choice for image classification tasks using CNNs. What truly distinguishes our solution is its user-friendly design, which enables individuals of all proficiency levels to easily grasp CNN implementation nuances. Through a meticulously engineered CNN architecture, our solution consistently achieves remarkable accuracy in classifying CIFAR-10 images, instilling confidence in its capabilities and inspiring users with its stellar performance. Furthermore, beyond its practical utility, our solution doubles as an invaluable educational tool, guiding users through the intricacies of CNNs, TensorFlow/Keras, and image classification techniques, fostering learning and skill development in an encouraging environment. Researchers, academics, data scientists, and software developers alike benefit from our solution's adaptability and efficiency, empowering them to tackle diverse image classification challenges across industries with unparalleled ease and effectiveness.



MODELLING

- ✓ Design Convolutional Neural Network (CNN) architecture for image classification on CIFAR-10 dataset.
- ✓ Include convolutional layers for feature extraction and pooling layers for downsampling.
- ✓ Add fully connected layers for classification and an output layer for predicting class probabilities.
- ✓ Compile the model with suitable loss function and optimizer.
- ✓ Train the model on the training dataset.
- ✓ Evaluate the trained model's performance on the test dataset to measure accuracy.

RESULTS

The code trains a Convolutional Neural Network (CNN) model on the CIFAR-10 dataset for image classification. It displays the training progress, including loss and accuracy metrics for each epoch. After training, the model is evaluated on the test dataset, yielding a final accuracy metric. In summary, the result provides insights into the model's training performance and its ability to classify unseen images accurately.

```
Epoch 2/10
1563/1563 [=====] - 74s 47ms/step - loss: 1.1921 - accuracy: 0.5775 - val_loss: 1.1142
Epoch 3/10
1563/1563 [=====] - 70s 45ms/step - loss: 1.0498 - accuracy: 0.6292 - val_loss: 1.0578
Epoch 4/10
1563/1563 [=====] - 70s 45ms/step - loss: 0.9556 - accuracy: 0.6655 - val_loss: 0.9824
Epoch 5/10
1563/1563 [=====] - 70s 45ms/step - loss: 0.8815 - accuracy: 0.6900 - val_loss: 0.9348
Epoch 6/10
1563/1563 [=====] - 71s 45ms/step - loss: 0.8241 - accuracy: 0.7112 - val_loss: 0.9354
Epoch 7/10
1563/1563 [=====] - 69s 44ms/step - loss: 0.7743 - accuracy: 0.7286 - val_loss: 0.8964
Epoch 8/10
1563/1563 [=====] - 70s 45ms/step - loss: 0.7333 - accuracy: 0.7413 - val_loss: 0.9009
Epoch 9/10
1563/1563 [=====] - 71s 46ms/step - loss: 0.6962 - accuracy: 0.7556 - val_loss: 0.8597
Epoch 10/10
1563/1563 [=====] - 69s 44ms/step - loss: 0.6628 - accuracy: 0.7664 - val_loss: 0.8546
313/313 [=====] - 4s 14ms/step - loss: 0.8546 - accuracy: 0.7105
Test accuracy: 0.7105000019073486
```