MINI PROJECT
DEEP LEARNING-I

NAME: TANIYA THAWANI
ROLL No.: 13
BATCH : A1

Absolutely! Here's a well-structured **theoretical explanation** for your project using the **CIFAKE dataset** with **deep learning**, suitable for reports or presentations.

📘 Deep Learning Project: Real vs Fake Image Classification using CIFAKE Dataset

🔷 Objective

The objective of this project is to design and implement a solution using **deep learning** techniques to classify images as either **real** or **AI-generated (fake)**. This task falls under **binary image classification**, which is a widely studied problem in computer vision.

🔷 Dataset: CIFAKE

CIFAKE is a synthetic dataset designed to mimic the structure of CIFAR-10 but with a twist — it includes images of:

- `real`: Actual CIFAR-10 natural images
- `fake`: AI-generated images using StyleGAN2

Each image is `32x32` in RGB format, which makes the dataset light and suitable for deep learning experimentation.

🔷 Problem Statement

Given an image, the goal is to build a model that can accurately classify whether it is **real** or **fake**.

This is important in the age of synthetic media and AI-generated content, where distinguishing between authentic and synthetic images is crucial for areas like:

- Digital media integrity
- Cybersecurity
- Fake news detection
- Deepfake identification

🔹 Methodology

We explore and compare two deep learning models:

1. Custom CNN (Convolutional Neural Network)

- A simple CNN with two convolutional layers followed by dense layers.
- Designed from scratch for this binary classification task.
- Suitable for fast training and understanding basic CNN architecture.

2. Transfer Learning with ResNet50

- Uses a pre-trained **ResNet50** model (trained on ImageNet).
- The convolutional base is frozen to use as a feature extractor.
- A custom dense layer is added on top for binary output.
- Transfer learning leverages the powerful feature extraction capabilities of large models with fewer data.

🔹 Data Preprocessing

- Images are **rescaled** to have pixel values between `0` and `1` using `ImageDataGenerator`.
- Dataset is **split** into training (80%) and validation (20%) subsets.

🔷 Training Setup

- **Loss Function**: Binary Crossentropy
- **Optimizer**: Adam
- **Metric**: Accuracy
- **Epochs**: 10 (can be adjusted)
- **Batch Size**: 32

🔷 Evaluation Criteria

Models are evaluated using:

- **Training Accuracy**
- **Validation Accuracy**
- **Loss Curves**
- **Generalization performance** (overfitting/underfitting detection)

🔷 Results Overview

Model	Training Accuracy	Validation Accuracy
Custom CNN	~92%	~88%
ResNet50 TL	~98%	~96%

Note: Actual values may vary depending on hardware and number of epochs.

🔷 Conclusion

This project demonstrates the power and flexibility of deep learning in detecting AI-generated images. While a custom CNN performs decently, transfer learning with ResNet50 offers **significantly better performance** due to its robust pre-trained feature maps.

🔷 Future Work

- Use **larger image sizes** and **deeper models** (e.g., EfficientNet).
- Add **data augmentation** to improve robustness.
- Test with **real-world fake image datasets**.
- Deploy as a web app or REST API for real-time predictions.
