CIFAR-100 Image Classification

Members: Bhakti Pasnani, Vaishnavi Bhutada, Vishwa Bhadiyadara

Abstract:

This project investigates approaches to classify images in the CIFAR-100 dataset. We implemented Logistic Regression, Logistic Regression with Principal Component Analysis (PCA), and Convolutional Neural Networks (CNNs) to analyze classification performance. Logistic Regression achieved an accuracy of 17.98%, which improved slightly to 18.47% with PCA. The CNN, leveraging spatial feature extraction, achieved the highest test accuracy of 52.57%.

Overview:

The CIFAR-100 dataset is a challenging benchmark for image classification, consisting of images from 100 diverse categories. This project explored three approaches to solve the classification problem: Logistic Regression, Logistic Regression with PCA, and CNNs. Starting with baseline methods provided insights into the dataset's structure and challenges. The CNN architecture significantly outperformed the other methods due to its ability to capture spatial relationships within images. This work highlights the limitations of traditional techniques and the power of deep learning in addressing complex visual tasks.

Problem Statement:

Accurate classification of images into 100 classes is a non-trivial task, given the dataset's complexity, including inter-class similarity and intra-class variability. The goal is to analyze the efficacy of traditional and deep learning-based methods in solving this task and identify the most effective approach.

Why is the problem interesting?

Image classification is central to numerous applications, such as:

- Healthcare: Analyzing medical images for disease detection.
- Autonomous Systems: Recognizing objects in real-world environments.
- Content Moderation: Classifying digital content for better curation.
  Studying CIFAR-100 provides a platform to develop and refine models for general-purpose classification tasks, with real-world implications**.**

Proposed Approach:

1. Logistic Regression: Used as a baseline to assess raw dataset features.
2. Logistic Regression with PCA: Applied dimensionality reduction to highlight critical features.
3. Convolutional Neural Networks (CNNs): Implemented to leverage spatial hierarchies and feature maps in the images.

These approaches were chosen to compare traditional classification methods with modern deep learning techniques.

Rationale Behind the Approach:

Logistic Regression provides a baseline to understand the dataset and limitations of linear classifiers. Adding PCA highlights the role of feature reduction in classification. CNNs, being specifically designed for visual tasks, represent the state-of-the-art in image classification and demonstrate significant performance improvements over traditional methods.

Key Components:

1. **Logistic Regression**: Simple yet interpretable baseline model.
2. **Logistic Regression with PCA**: Reduced computational complexity by reducing dimensionality.
3. **CNN**: Introduced convolutional layers, pooling, and dropout for robust feature extraction and generalization.

Limitations:

- **Logistic Regression**: Limited by its inability to model non-linear features effectively.
- **PCA**: While it slightly improved performance, it failed to capture the dataset's intricate patterns.
- **CNN**: Despite being the best-performing model, the accuracy (52.57%) indicates the need for further tuning and exploration of advanced architectures.

Experiment Setup:

Dataset:

The CIFAR-100 dataset contains:

- **50,000 training images** and **10,000 test images**, each labeled with one of 100 categories.
- Categories include animals, vehicles, and everyday objects.
- Data preprocessing included normalization and one-hot encoding of labels.

Implementation:

- **Logistic Regression**: Implemented using scikit-learn with default settings.
- **PCA**: Reduced feature dimensions from 3,072 (32x32x3) to 500 principal components before applying Logistic Regression.
- **CNN**: Utilized three convolutional blocks, each followed by pooling and dropout layers, culminating in a fully connected softmax layer.

Computing Environment:

- **Platform**: Google Colab with NVIDIA Tesla T4 GPU.

- **Libraries**: TensorFlow, Keras, scikit-learn, and Matplotlib.

Model Architectures:

Logistic Regression

- Standard implementation using all pixel values as features.
- Regularization enabled to avoid overfitting.

Logistic Regression with PCA

- Features reduced to 500 components before applying Logistic Regression.

CNN Architecture

1. **Input Layer**:
    - Input shape: (32, 32, 3) (images from CIFAR-100 dataset, 32x32 pixels, 3 color channels).
2. **First Convolutional Block**:
    - **Conv2D Layer**: 64 filters of size (3, 3) with ReLU activation, using 'same' padding. This ensures the output spatial dimensions remain the same as the input.
    - **Batch Normalization**: Normalizes activations to speed up training and improve stability.
    - **MaxPooling2D**: Pooling size (2, 2) to reduce spatial dimensions by half.
    - **Dropout**: 30% dropout to prevent overfitting.
3. **Second Convolutional Block**:
    - **Conv2D Layer**: 128 filters of size (3, 3) with ReLU activation and 'same' padding.
    - **Batch Normalization**: Normalizes activations for better convergence.
    - **MaxPooling2D**: Pooling size (2, 2) to further reduce spatial dimensions.
    - **Dropout**: 40% dropout for regularization.
4. **Third Convolutional Block**:
    - **Conv2D Layer**: 256 filters of size (3, 3) with ReLU activation and 'same' padding.
    - **Batch Normalization**: Maintains stability during training.
    - **MaxPooling2D**: Pooling size (2, 2) reduces spatial dimensions to a smaller size.

- **Dropout**: 50% dropout to combat overfitting.

5. **Flattening Layer**:
    - Converts the 3D feature map into a 1D feature vector for the fully connected layers.

6. **Fully Connected Layers**:
    - **Dense Layer**: 512 neurons with ReLU activation.
    - **Dropout**: 50% dropout to further reduce overfitting.
    - **Dense Layer**: 100 neurons with softmax activation for classification into 100 classes (one class for each CIFAR-100 category).

7. **Optimizer and Loss Function**:
    - **Optimizer**: Adam optimizer with a learning rate of 1e-3.
    - **Loss Function**: Categorical crossentropy, suitable for multi-class classification problems.
    - **Metrics**: Accuracy is used to track performance during training and validation.

Experiment Results:

Main Results

- **Logistic Regression**: Achieved 17.98% accuracy.
- **Logistic Regression with PCA**: Slight improvement to 18.47%.
- **CNN**: Outperformed the traditional methods, achieving 52.57% accuracy.

Supplementary Results

- **Data Augmentation**: Techniques like rotation, zooming, and flipping were applied during CNN training to improve robustness.
- **Hyperparameter Tuning**: Learning rate adjustments, dropout rates, and batch sizes were experimented with in CNNs to enhance generalization.

Discussion:

The project underscores the limitations of traditional approaches like Logistic Regression for complex datasets like CIFAR-100. Adding PCA improved feature relevance but fell short in capturing spatial hierarchies. CNNs, with their hierarchical feature extraction, significantly outperformed Logistic Regression, achieving 52.57% accuracy. However, the performance suggests potential for further improvement through deeper architectures (e.g., ResNet, Wide ResNet) or ensemble methods.

Conclusion:

This project explored three classification methods on CIFAR-100: Logistic Regression, Logistic Regression with PCA, and CNNs. While Logistic Regression served as a baseline, CNNs demonstrated the power of deep learning for visual tasks, achieving 52.57% accuracy. The results emphasize the importance of advanced architectures for solving complex classification problems.

References:

1. CIFAR-100 Dataset: https://www.cs.toronto.edu/~kriz/cifar.html
2. Keras Documentation: https://keras.io
3. scikit-learn PCA Implementation: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html
4. CNN : https://www.tensorflow.org/tutorials