Deep Learning and Its Applications to Signal and Image Processing and Analysis - Assignment 3

361.2.1120

May 1, 2025

Introduction

In this assignment, you will perform an image classification task on the CIFAR-10 dataset using two model families: Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs). The objectives are to develop models, apply explainability tools (Grad-CAM and attention visualization), and evaluate comparative performance using confusion matrices and other metrics. You will also learn how to use the `pytorch-lightning` library, which simplifies model building and training and supports automatic logging to Weights & Biases. A complementary notebook is provided with this assignment for your convenience; you are free to modify it as needed.

Important: Please review the instructions file before addressing the questions. **(5 Pts)**

1 CNN Classification and Grad-CAM Explainability

In this section, you will implement a CNN from scratch and apply Grad-CAM to explain the model's predictions.

1.1 Import CIFAR-10

a. Import the CIFAR-10 dataset. Show an example image for each label.

b. Split the dataset into training, validation, and test sets.

c. Show a histogram of the categorical distribution for training, validation, and test sets.

1.2 Building the CNN

a. Implement a CNN in PyTorch Lightning. Do not use pretrained models or architectures from `torchvision`.

b. Train the model on CIFAR-10. Use PyTorch Lightning's `ModelCheckpoint` callback to save the model weights corresponding to the epoch with the lowest validation loss. Display the loss convergence plot.

c. Test the model. Report the F1 score and confusion matrix.

1.3 Explainability with Grad-CAM

a. Choose the NN layer for which you apply the Grad-Cam. Explain your choice. (Grad-CAM Paper)

b. Use the `pytorch-grad-cam` library to generate Grad-CAM heatmaps for several test images (correct and incorrect predictions). See Figure 1 for an example.

c. Discuss whether the highlighted regions are semantically meaningful and whether the model attends to relevant features.

2 Vision Transformer (ViT) and Attention Visualization

In this section, you will implement a Vision Transformer (ViT) from scratch, compare it to the CNN model developed in Section 1, and visualize attention maps to gain insights into the model's decision process.

2.1 Implementing the Vision Transformer

a. Implement a Vision Transformer (ViT) using PyTorch Lightning. Your implementation should include:

- A patch embedding layer that divides the image into non-overlapping patches and projects them into embeddings.
- A positional encoding mechanism (learned or fixed).
- Transformer encoder blocks with multi-head self-attention and feed-forward layers.
- A classification token and an output head for CIFAR-10 classification.

b. Train the model on CIFAR-10. Use PyTorch Lightning's `ModelCheckpoint` callback to save the model weights corresponding to the epoch with the lowest validation loss. Display the loss convergence plot.

c. Test the model. Report the F1 score and confusion matrix.

2.2 Visualizing Attention Maps

a. Extract and visualize attention heatmaps for several test images (correct and incorrect predictions).

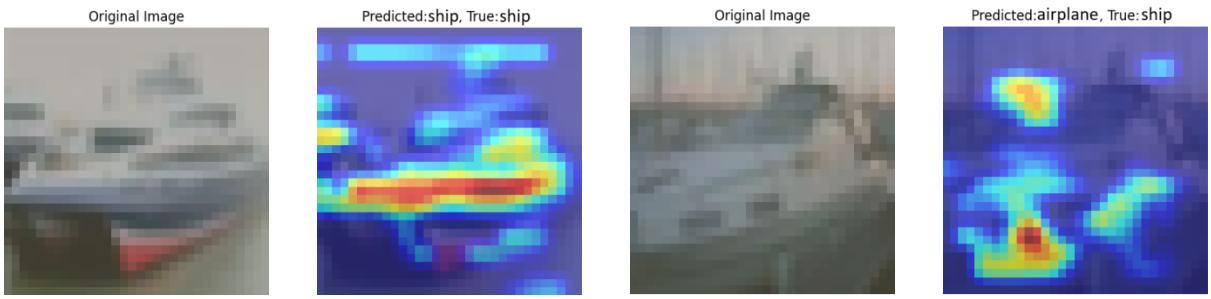
b. Discuss whether the highlighted regions are semantically meaningful and whether the model attends to relevant features.

c. Select an example where both the CNN and ViT made correct predictions. Compare the Grad-CAM visualization (from Section 1) with the attention maps (from Section 2) for this example. Discuss the similarities and differences between the two explainability methods in terms of the regions they highlight and how they reflect the models' decision processes.

2.3 Comparison and Analysis

a. Compare the CNN and ViT models in terms of:

- Number of learnable parameters in each model.
- Accuracy and confusion matrix.
- Training time and convergence speed.
- Explainability via Grad-CAM (for CNNs) and attention maps (for ViTs).



(a) (b) (c) (d)

Figure 1: Visualization examples: (a) Original input image (correct prediction); (b) Grad-CAM for correct prediction; (c) Original input image (incorrect prediction); (d) Grad-CAM for incorrect prediction.