CMPSC 448 Final Project Report

Greg Dorazio

December 3, 2023

Task, Dataset, and Preprocessing

For this project, I implemented an image classification deep learning system using the CIFAR-10 dataset from the Keras datasets library. The dataset consists of 10 classes: airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. It contains 50,000 images for training and 10,000 images for testing.

Datasets Explored

Initially, I considered datasets like the Simula image sentiment dataset and the Emotion6 Emotion dataset. However, I ultimately chose the CIFAR-10 dataset due to its simplicity and suitability for image classification tasks.

Implemented Architectures

I implemented two deep learning systems: a Convolutional Neural Network (CNN) and a Vision Transformer (ViT). CNNs are great for image classification because they're designed to work with grid-like data such as images. Out of the 3 choices, CNNs are generally the best option for this task. I also chose the ViT because although transformers are generally used for sequential data, this specific vision transformer is adapted to work with grid-like data. RNNs are similar in that they are commonly used for sequential data, but they cannot be adapted in the same way a transformer can.

CNN Architecture

For the CNN, I used the following architecture:

- For the preprocessing of data, the cifar dataset is divided into training and test sets. Pixel values are normalized [0,1]. Finally, class labels are one-hot encoded using tf.keras.utils.to categorical.
- Three convolutional layers with 32, 64, and 128 filters, each followed by a max-pooling layer.
- A Flatten layer to convert the 3D output to a 1D array.
- Two dense layers, each with 256 units and 'relu' activation functions.
- Dropout layers after the first and second dense layers to reduce overfitting.
- A final dense layer with 10 units for the 10 classes of CIFAR, and softmax activation.

The model was compiled with the Adam optimizer, and categorical cross entropy was used as the loss function. The model was trained for 30 epochs using augmented data and then recompiled and trained for another 10 epochs on the original training data.

ViT Architecture

For the ViT, I used the following architecture:

- The preprocessing is the same, the dataset is divided into training and test sets, pixel values normalized to [0,1]
- The 'vit_132' layer from the ViT-Keras library as the core of the model.

- The 'image_size' parameter set to 32 to make it compatible with the model.
- A Flatten layer to convert the 3D output to a 1D array.
- Two dense layers, each with 256 units and 'relu' activation functions.
- A final dense layer with 10 units for the 10 classes in CIFAR with softmax activation.

The model was compiled using the Adam optimizer, and sparse categorical cross entropy was used as the loss function. The model was trained for 10 epochs using the training data, with validation data provided for evaluation.

Results and Observations

The final score for the CNN was 76.17 percent. The final score for the ViT (after 5 epochs) was 54 percent. In fairness to the vision transformer, if it had run for even 5 more epochs, it was likely to reach at least 70 percent accuracy and likely higher than that. From these observations, it seems that ViT might take much more time to train than a conventional CNN, but can yield similar results in terms of accuracy.

Challenges and Solutions

Originally, the CNN was receiving a score of 71.08 percent. In order to increase the accuracy, I added another convolutional layer, increased the number of units for the first dense layer, and added another dense layer. This increased the accuracy to 75.91 percent. I then added weight regularization to the dense layers to prevent overfitting. I did not consider the very lengthy training time for the ViT. The architecture I made had 10 epochs, but each epoch was around 16 minutes in length. After many hours I would run into errors, so I sacrificed accuracy in order to keep the training length down.