EN3150 - Pattern Recognition

Simple convolutional neural network to perform classification

Name	Index
Deshan KAGD	200117T
Dewagiri DKD	200119C
Kariyawasam KKD	200262G
Nimnaka KWH	200426N

Contents

1 CNN for image classification

- **Q). Why CNNs preferable for image classification over multilayered perceptrons (MLPs) or simple feedforward neural networks (NNs)?**

  Spatial Hierarchy: Originally intended for detecting spatial patterns in images, CNNs employ convolutional layers (kernels) to identify local patterns, progressively combining them to discern more complex features. In contrast, MLPs treat input data as flat vectors, leading to the loss of gradient information in images and suboptimal precision in identifying edges and corners.

  Translation Invariance: CNNs achieve the ability to recognize image features independent of their positions by utilizing shared filters in convolutional layers. In contrast, MLPs may encounter challenges in identifying the same feature when it appears in different locations within the image.

  Pooling: CNNs incorporate pooling layers to diminish spatial dimensions while retaining translation invariance, a feature lacking in MLPs. Consequently, MLPs require more layers and parameters to achieve comparable results.

- **Q). Determine the parameters of the above network such as kernel sizes, filter sizes, size of the fully connected layer and dropout rate.**

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 32)        896

 batch_normalization (Batch  (None, 32, 32, 32)        128
 Normalization)

 max_pooling2d (MaxPooling2  (None, 16, 16, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 16, 16, 64)        18496

 batch_normalization_1 (Bat  (None, 16, 16, 64)        256
 chNormalization)

 max_pooling2d_1 (MaxPoolin  (None, 8, 8, 64)          0
 g2D)

 conv2d_2 (Conv2D)           (None, 8, 8, 128)         73856

 batch_normalization_2 (Bat  (None, 8, 8, 128)         512
 chNormalization)

...
Total params: 1280202 (4.88 MB)
Trainable params: 1278218 (4.88 MB)
Non-trainable params: 1984 (7.75 KB)
```

Figure 1

Kernel size = 3×3
Filter sizes used = 32, 64, 128
Sizes of the fully connected (dense) layers:?? with ReLu, ?? with SoftMax

- **Q). Train the model: Train the model using the training data for 20 epochs and plot training and validation loss for with respect to epoch.**

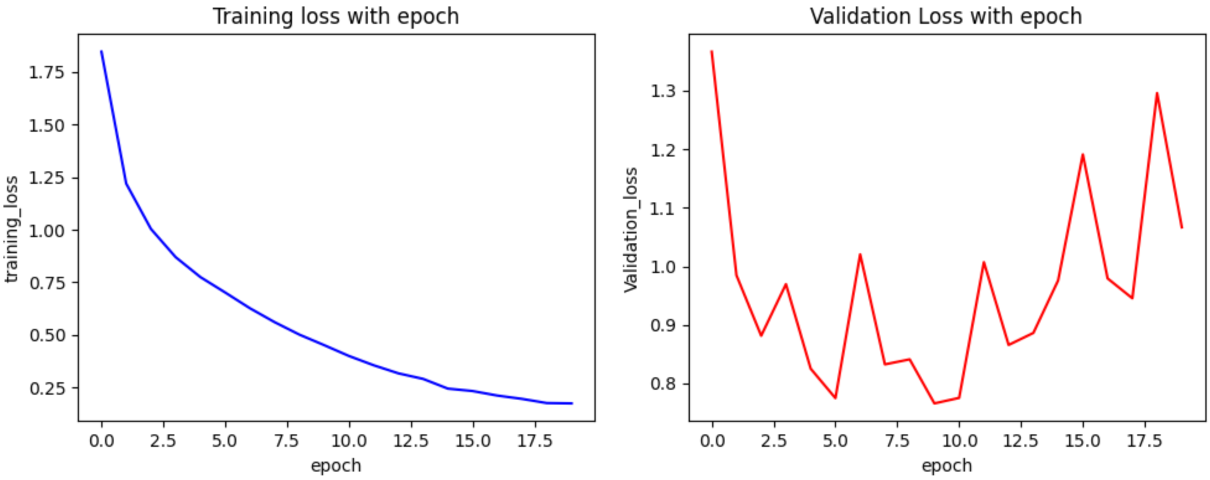

Figure 2: Loss vs Epochs(Learning rate = 0.001)

- **Q). Why we have chosen adam optimizer over SGD?**

The Adam optimizer employs adaptive learning rates by considering historical gradient information, facilitating quicker convergence and improved solutions, particularly in scenarios with fluctuating gradient magnitudes. In contrast, stochastic gradient descent (SGD) necessitates an initial specification of the learning rate, which remains static and is not adjusted adaptively. The Adam optimizer incorporates a gradient scaling feature during backpropagation to stabilize the training process, thereby minimizing the need for manual tuning of the learning rate, a requirement in the SGD algorithm.

- **Q). Why we have chosen sparse categorical crossentropy as the loss function?**

Sparse Categorical Cross Entropy serves as a widely employed loss function for multi-class classification challenges. It operates with integer encoding, which proves to be more efficient, especially when dealing with a large number of categories or labels. This approach helps conserve substantial resources, particularly in terms of RAM usage.

$\text{Loss} = -\sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i$

- **Q). Evaluating the Model - After training, appraise the model's performance on the testing dataset. Record the accuracy for both training and testing, along with metrics like the confusion matrix, precision, and recall?**
F1 Score: 0.7172
Precision Score: 0.7688
Recall Score: 0.6721

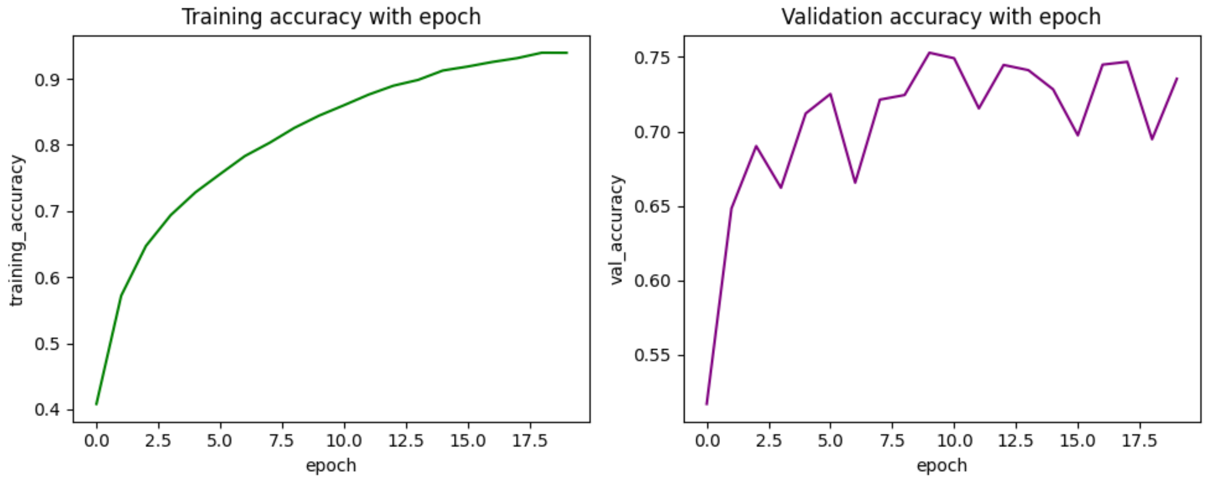

Figure 3: Accuracy vs Epochs

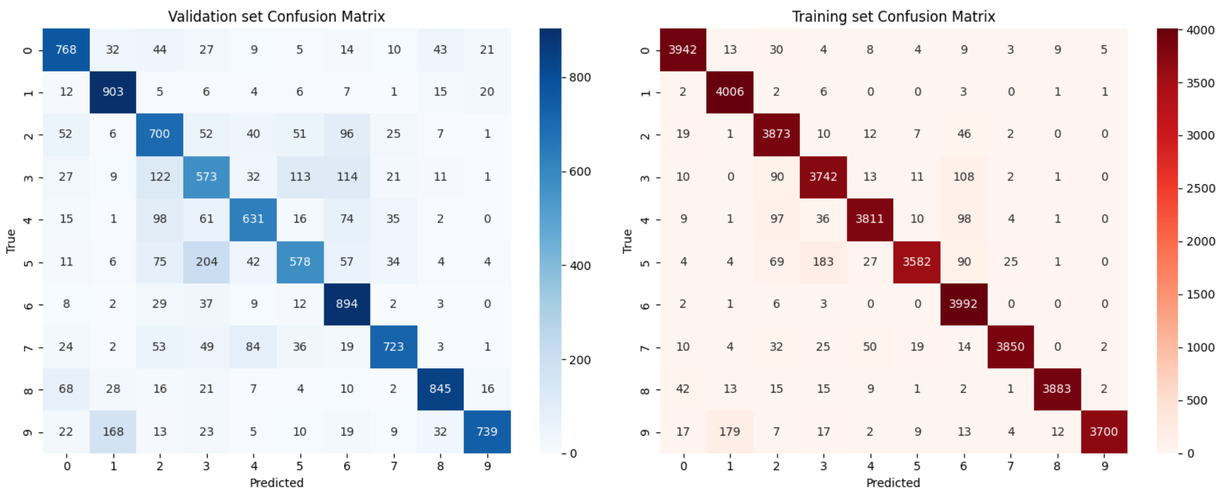

Figure 4: Confusion Matrix

- **Q.) Plot training and validation loss for with respect to epoch for different learning rates such as 0.0001, 0.001, 0.01, and 0.1?**

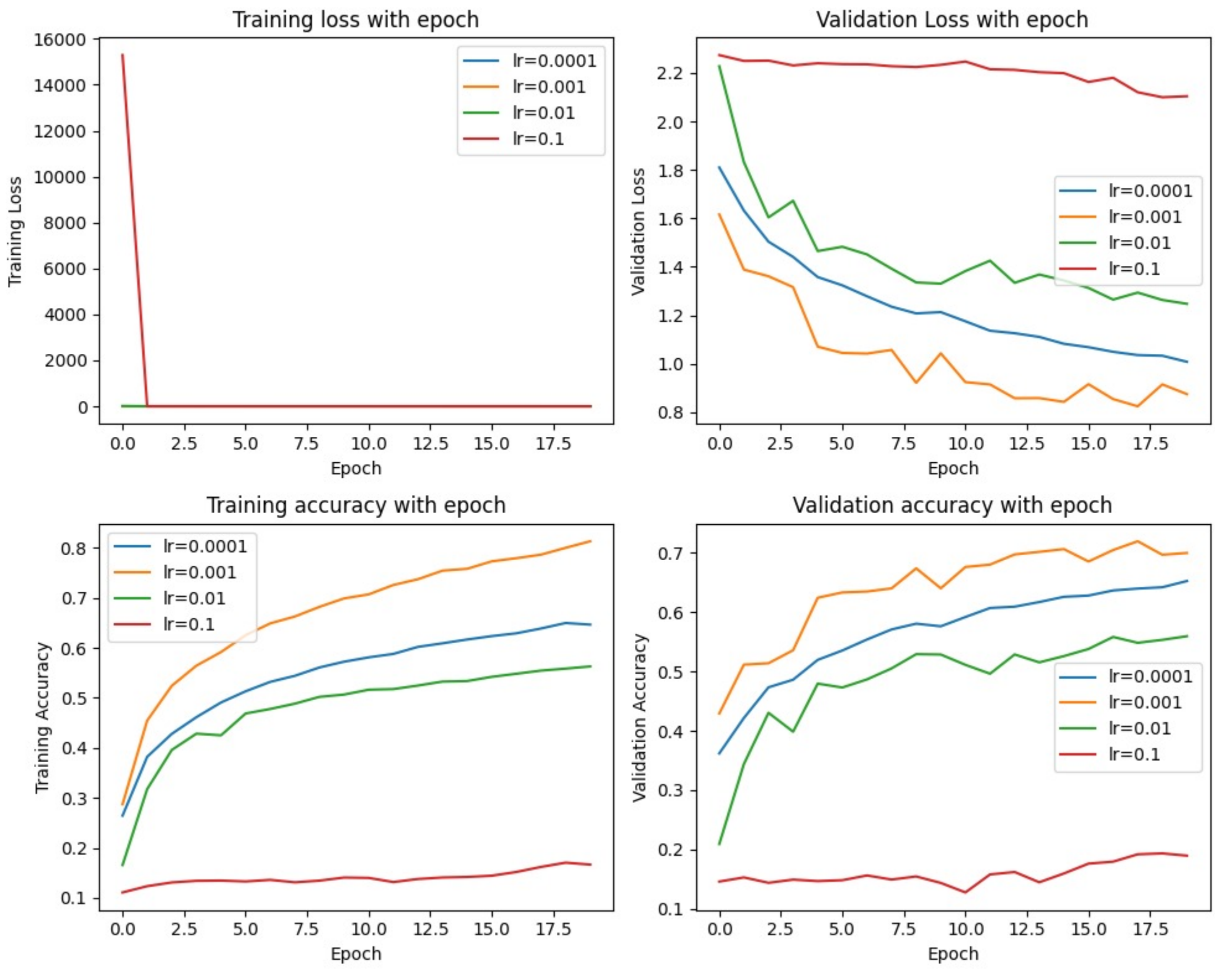

Figure 5: Training Loss and Accuracy vs Epochs

2 Compare your network with state-of-the-art networks

- **Q. Choose two state-of-the-art pre-trained model or architecture like ImageNet, ResNet, Googlenet, AlexNet, DenseNet and VGG?**

  We used ResNet Architecture and VGG

- **Q. Record training and validation loss values for each epoch.?**
- **Q. Evaluate the fine-tuned model on the testing dataset and calculate the test accuracy?**

  Evaluations:
  F1 score:
  Precision score:

Recall score:

- **Q. Compare the test accuracy of your custom CNN model with that of the fine-tuned state-of-the-art model?**

  Test Accuracy of the our custom CNN model:
  F1 score:
  Precision score:
  Recall score:

  Based on the provided statistics, it is evident that our optimized state-of-the-art model achieves superior test accuracy compared to the custom CNN model. Unlike the custom CNN model, the state-of-the-art model meticulously processes the dataset, resulting in more precise categorization. Consequently, state-of-the-art models are generally favored, yet custom models can prove advantageous for specialized or domain-specific applications.

- **Q. Discuss trade-offs, advantages, and limitations of using a custom model versus a pre-trained model.?**

Custom CNN Model	**Pre-Trained CNN Model**
We have the flexibility to make personalized modifications to the model according to our preferences. We possess full control over the architecture,hyperparameters, and training procedures, enabling us to tailor the model to meet your specific needs.	In this scenario, we are presented with a pre-established structure, and our influence over parameter tuning operations is limited.
Ability to address tiny differences and wants for our use case.	They have learned useful features from a wide range of data, which of ten leads to good performance when we have limited data for a specific problem.
Custom model training often requires a wide and varied dataset, which can be challenging and costly to gather and preprocess.	Prailable and easy to integrate into our applications.