Deep Learning Lab-1

immediate

September 28, 2024

Abstract

This project focuses on classifying architectural styles in the MexCulture142 dataset, which contains 284 images of 142 Mexican monuments categorized into three styles: Prehispanic, Colonial, and Modern.Using transfer learning, we applied a ResNet50 model pretrained on the ImageNet dataset and fine-tuned it for this classification task. TensorFlow/Keras framework is considered, with the model achieving a highest accuracy of 91% on the validation set. The training process was carried out in a virtual environment at CREMI and Kaggle, leveraging the pretrained model to efficiently adapt to the characteristics of the MexCulture142 dataset.

1 Data Loading Preprocessing

The dataset was divided into training and validation sets, with the images labeled based on their filename prefixes. In total, 236 images were found for training and 48 for validation, split into three classes: Prehispanic, Colonial, and Modern. The training set consists of 150 Colonial images, 52 Prehispanic images, and 34 Modern images. The validation set is balanced with 16 images for each class. All images were resized to 224x224 pixels, and class labels were one-hot encoded. Neural networks, particularly when using categorical cross-entropy as a loss function, expect the labels to be one-hot encoded. This format ensures that the model outputs probabilities for each class, facilitating effective classification.

Fig. 1: Shows 10 images from trainig dataset of MexCulture142

2 Data Augmentation and Generator Setup

Given the relatively small size of the dataset, data augmentation is applied to artificially increase its diversity and improve model generalization. Techniques like horizontal flipping, zoom, rotation, and shifts are used to enhance the training data. The `ImageDataGenerator` also applies `preprocess_input` to prepare the images for the ResNet model.

Training and validation data are processed in batches of 8, 16, and 32 (with 8 being the final choice). Augmentation and shuffling are applied to the training set, while the validation set is only preprocessed without augmentation or shuffling. Class labels (Prehispanic, Colonial, Modern) are mapped to numerical indices for easy interpretation during model evaluation.

3 Model Architecture: ResNet50 with Custom Layers

We used **ResNet50**, pretrained on **ImageNet**, as the base model. Its layers were frozen to retain the powerful feature extraction abilities. On top, we added custom layers: a Global Average Pooling layer, followed by a 512-unit Dense layer with ReLU activation, providing sufficient capacity for the model to learn complex patterns in the data. Finally, a **Dense output layer** with softmax activation was added to classify the images into the three architectural styles (Prehispanic, Colonial, Modern).

The 512-unit dense layer strikes a balance between model complexity and performance, offering enough flexibility to adapt to our specific dataset without overfitting.

In total, the model contains **24,638,339** parameters, of which **1,050,627** are trainable, and **23,587,712** are non-trainable. This structure leverages the pretrained knowledge from ResNet50 while fine-tuning the final layers for the specific task at hand.

4 Model Compilation and Training

The model was compiled using the **Adam optimizer** with a learning rate of **0.01**, after experimenting with other learning rates such as **0.0001** and **0.001**. The loss function chosen for this multiclass classification task is **categorical cross-entropy**, with accuracy as the performance metric.

To avoid overfitting and ensure optimal training, we implemented **early stopping** and **learning rate reduction** callbacks. Early stopping was triggered if validation loss did not improve for **5 consecutive epochs**, restoring the best model weights. The learning rate was reduced by a factor of **0.2** if validation loss plateaued for **3 epochs**, allowing the model to dynamically fine-tune its learning rate.

Training was set for a maximum of **30 epochs**, though the use of early stopping meant training could halt earlier if no further improvement in validation loss was observed.

5 Hyperparameter Tuning and Results

Throughout the training process, various hyperparameters such as learning rate and batch size were experimented with to achieve optimal performance. Early stopping was also applied in certain runs to prevent overfitting and improve generalization. Below is a summary of the different configurations and their respective outcomes in terms of training accuracy, validation accuracy, loss, and misclassifications.

Exp.	LR	Batch	Epochs	Early Stop	T-Acc.	V-Acc.	T-Loss	V-Loss
1 (Initial Run)	0.0001	16	30	No	N/A	86%	N/A	N/A
1 (Early Stop)	0.0001	16	30	Yes	N/A	89%	N/A	N/A
2	0.0001	32	16	Yes	99.58%	87.50%	0.0260	0.4250
3	0.0001	8	11	Yes	100.00%	81.25%	0.0152	0.4173
4	0.001	8	14	Yes	100.00%	89.58%	0.0002	0.6513
5	0.01	8	11	Yes	99.58%	91.67%	0.0045	0.8456

Table 1: Hyperparameter Tuning Results, N/A here i have not recorded

This summary highlights how tuning hyperparameters affected the model's performance, with the highest validation accuracy of **91.67%** achieved using a learning rate of **0.01** and a batch size of **8** with early stopping after 11 epochs.

6 Plots and Confustion Matririx

The training accuracy steadily increases, reaching a final value of **99.58%** by the 11th epoch, indicating that the model is learning and fitting well on the training data. The validation accuracy, however, fluctuates across epochs, starting at around **70%** and gradually rising to **91.67%** by the final epoch. This suggests that while the model performs well on the validation set, the fluctuations indicate some variability in its generalization.

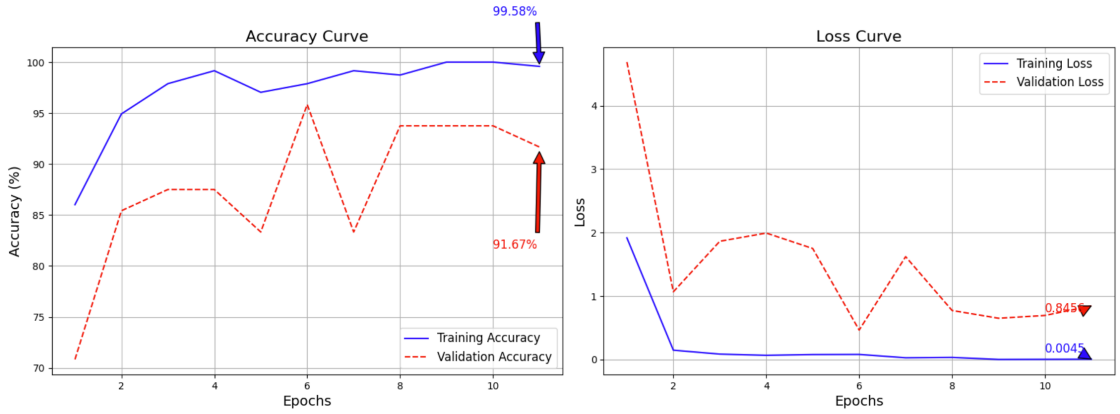
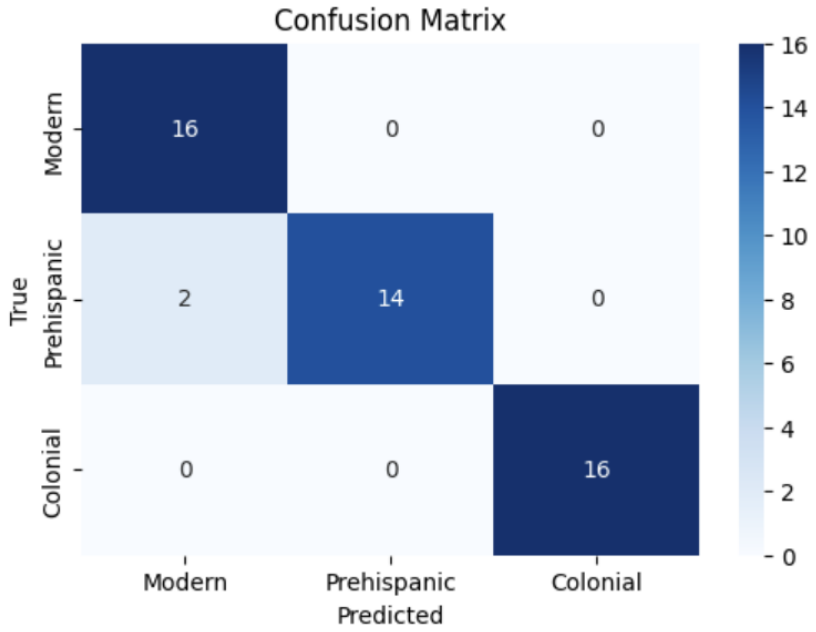


Fig. 2: learning Rate 0.01 and batch size 8



	precision	recall	f1-score	support
Modern	0.89	1.00	0.94	16
Prehispanic	1.00	0.88	0.93	16
Colonial	1.00	1.00	1.00	16
accuracy			0.96	48
macro avg	0.96	0.96	0.96	48
weighted avg	0.96	0.96	0.96	48

Fig. 3: Confusion Matrix

The model performed exceptionally well, achieving a total accuracy of **96%**. Most of the classes, particularly **Modern** and **Colonial**, were predicted with great accuracy, with all images from these classes being correctly classified. The **Prehispanic** class had **2 images** misclassified as **Modern**, but overall, the model still performed strongly, with high precision and recall values for this class. Precision, recall, and F1-scores across all categories were consistently high, with macro and weighted averages at **0.96**. This indicates that the model not only handled the majority of the images correctly but also generalizes well across different architectural styles.

Fig. 4: Misclassified Images

The model likely misclassified these **Prehispanic** images as **Modern** due to visual similarities, such as shared architectural features like geometric shapes or large windows. Another reason could be an imbalance in the training data, where fewer Prehispanic examples led the model to rely more on features common to Modern architecture.

Fig. 5: correclty Classfied Images