UNIVERSITY OF SOUTHERN CALIFORNIA

DSCI 552 MACHINE LEARNING FOR DATA SCIENCE

PROJECT REPORT

CATEGORY AND LANDMARK CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS AND TRANSFER LEARNING

Abstract:

The project aims to develop a model to predict categories and landmarks for a given image. The dataset consists of images of famous landmarks, organized into six categories and 30 landmarks. Several models were trained with varying degrees of augmentation and transfer learning, and their performance was evaluated on both the training and test data. The results show that using transfer learning and data augmentation can significantly improve the model's performance on the test set. Hyperparameter tuning and selecting the right pre-trained model can also affect the results.

Project Team 9

S No	Name	MAJOR/USC ID
1	SAKETH REDDY REGATTE	ADS / 4101368705
2	SAI MANIHAR REDDY NALLAGONDU	ADS / 6249448125
3	PAVITHRA KOLLIPARA	ADS / 7183732161

1. INTRODUCTION

In this project, we explore the impact of transfer learning and data augmentation on the performance of a machine learning model. We aim to build a model that can perform a specific task with high accuracy and generalization ability. We experiment with different training approaches, such as without augmentation without transfer learning, with augmentation without transfer learning, with augmentation with transfer learning, and with augmentation with transfer learning using EfficientNetB0. We evaluate the models' performance on both the training and test data and analyze the results to gain insights into the impact of different training approaches on the model's performance.

2. DATASET DESCRIPTION

The dataset consists of images of famous landmarks organized into a two-level hierarchy structure. The first level is categories, and the second level is landmarks. There are 6 categories - Gothic, Modern, Mughal, Neoclassical, Pagodas and Pyramids. Each category consists of 5 landmarks for a total of 30 landmarks. Each landmark has 14 images indicating that the dataset has a total of 420 images.

The dataset consists of images of multiple dimensions that must be resized to a uniform size to train the model. A scatter plot between the height and width of each image and histogram has been plotted to observe the distribution of the dimensions of the data.

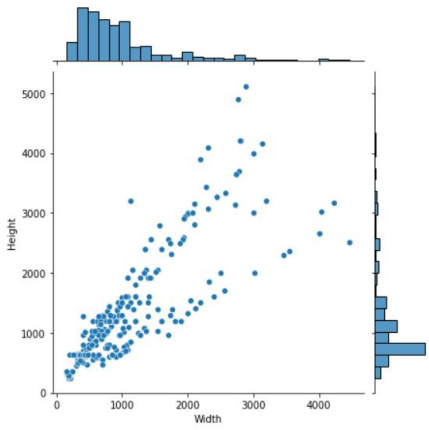

Figure 2.1 Distribution of sizes of images

2.1 Data Preprocessing:

Every image in the dataset is resized to a fixed dimension of 224 x 224. Each pixel is represented by a value between 0 and 255. Therefore, we normalize each pixel value by rescaling the images by a factor of 1/255. Finally, the processed dataset is divided into a training set (357) and a validation set (63) with a ratio of 85:15.

2.2 Data Augmentation:

The size of the dataset is very small and will result in over-fitting, so we use data augmentation to overcome this problem. Data augmentation is a technique used to generate new training data by applying random transformations to existing data without changing the label or target. The

various transformations like flipping, rotating, zooming, shifting, shearing, cropping, adding noise, and adjusting brightness and contrast are applied to our current images to generate additional images.

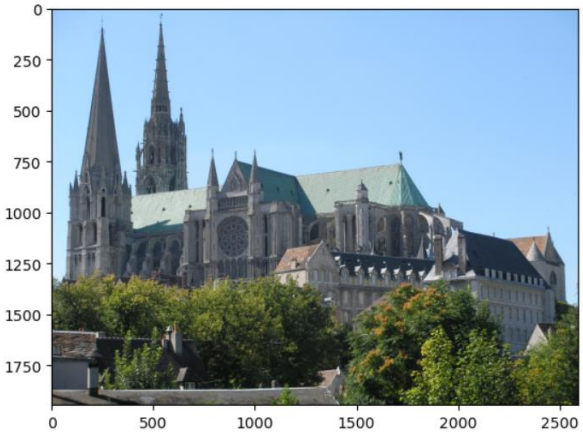

Figure 2.2 Original Image

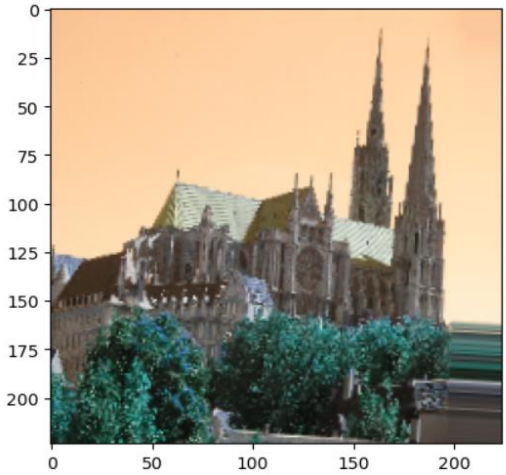

Figure 2.3 Final Transformed Image

3. RELATED WORK

The dataset has been initially experimented with different models and training approaches. The models trained without transfer learning and without augmentation are overfitting, as the training accuracy is significantly higher than the test accuracy. This suggests that the model is memorizing the training data instead of generalizing it to new data. Adding augmentation to the training data improved the model's performance on the training set but did not improve its generalization ability, as the test accuracy actually decreased. Using transfer learning along with data augmentation improved the test accuracy significantly, but the training accuracy is still higher than the test accuracy, indicating some level of overfitting. Tuning the hyperparameters could potentially further improve the model's performance. Using a specific pre-trained model (EfficientNetB0) did not lead to satisfactory results, as the test accuracy is low compared to the previous model.

Model	Train Accuracy	Test Accuracy	Comments
Without augmentation without transfer learning	86.75	56.19	Overfitting model
With augmentation without transfer learning	99.2	47.6	Overfitting model
With augmentation with transfer learning	98.08	80.09	Need to tune hyperparameters
With augmentation with transfer learning - EfficientNetB0	80.6	60.53	Results not satisfactory

Table 3.1: Experimenting with different models

4. METHODOLOGIES USED

4.1 Method 1

We have constructed a model that utilizes VGG16 as its foundation and extends the architecture further. The final two layers are dedicated to categories and landmark classes, respectively. This dual-output configuration enables the model to generate predictions for both the category and landmark of a given image simultaneously. We are not sure if category and landmark values are influencing each other or not. And if they are influencing each other, we don't know whether it is a positive or negative impact. Hence we have decided to use two different models for category prediction and landmark prediction.

4.2 Method 2

Two separate models were trained for predicting categories and landmarks, both utilizing VGG as their base architecture. Additional layers were constructed on top of the base model, with the final dense layer tailored to the number of output classes for each task. When processing an image, it is first input to the category prediction model, followed by the landmark prediction model. This has resulted in better test accuracy for both classifications. The model's accuracy was sensitive to minor alterations in its parameters.

To achieve the highest validation accuracy, we employed **RandomSearch** from Keras for hyperparameter tuning. RandomSearch is a method that systematically searches through different combinations of hyperparameter values to identify the best-performing configuration. The model's performance was optimized, ensuring that it delivers accurate predictions even when parameters vary slightly.

4.3 Method 3

The results of category classification using a category model built with **VGG16** as its base model are fed to the landmark classification model. The second model will result in 30 classes giving the prediction of landmarks. Since the landmark model has access to the category predictions, it can leverage this information to refine its classification. For example, if the category model predicts that a given image belongs to a specific category, the landmark model might focus more on the landmarks that are associated with that category. In other words, the category predictions can act as a context that helps the landmark model make more accurate predictions.

Multimodal learning is a technique that leverages information from multiple sources or modalities to improve a model's performance, allowing it to capture more complex relationships and make better predictions. In this case, the category information helps to guide the landmark classification, potentially leading to more accurate and robust results.

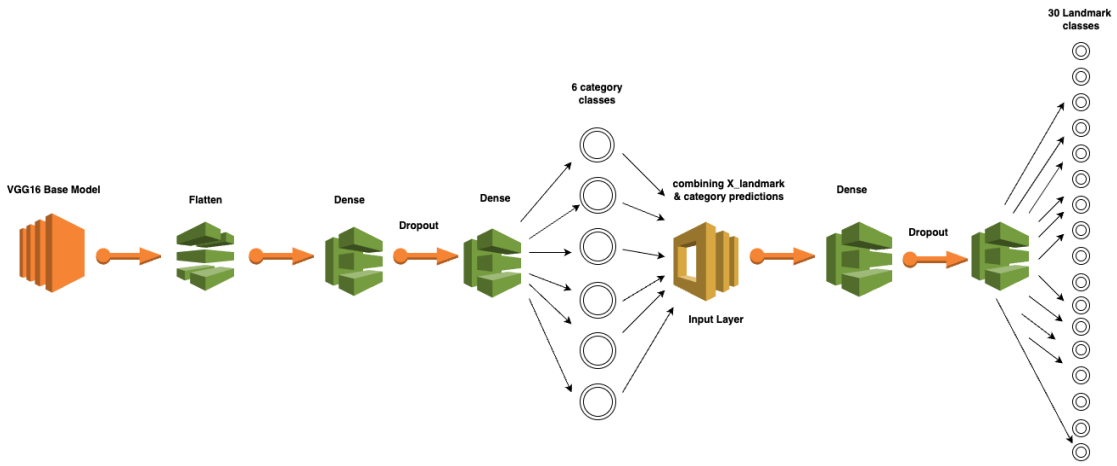

Fig 4.1 Model Architecture

5. RESULTS

The accuracies and F1-scores obtained on the test set for the above three methods are summarized below.

	Category		Landmark	
Model	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
Method 1 (Test)	91.67	0.8	70.74	68.7
Method 2 (Test)	90	0.86	85.2	62.89
Method 3 (Vocareum)	92.86	86.19	82.14	72.93

Table 5.1: Evaluation Results

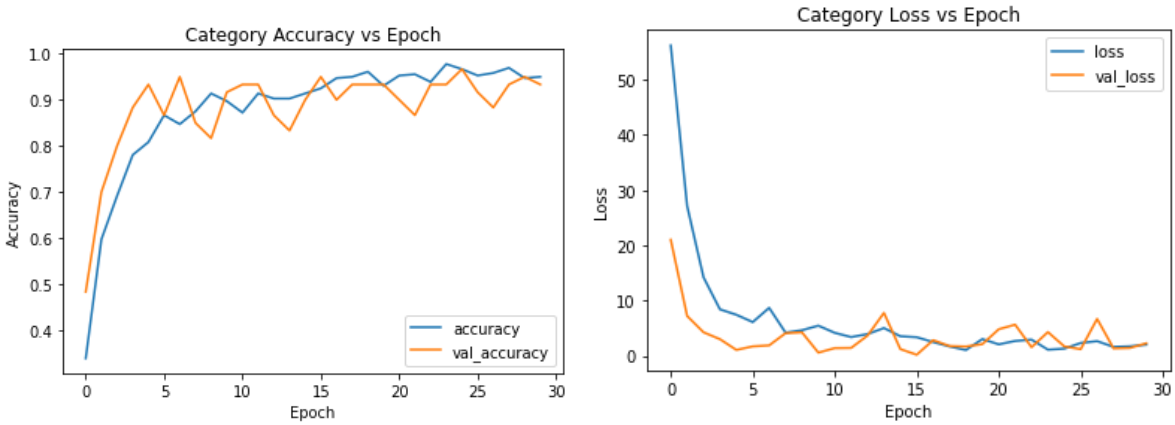

Fig 5.1 Accuracy/Loss v/s Epochs GRAPHs for Category Classification

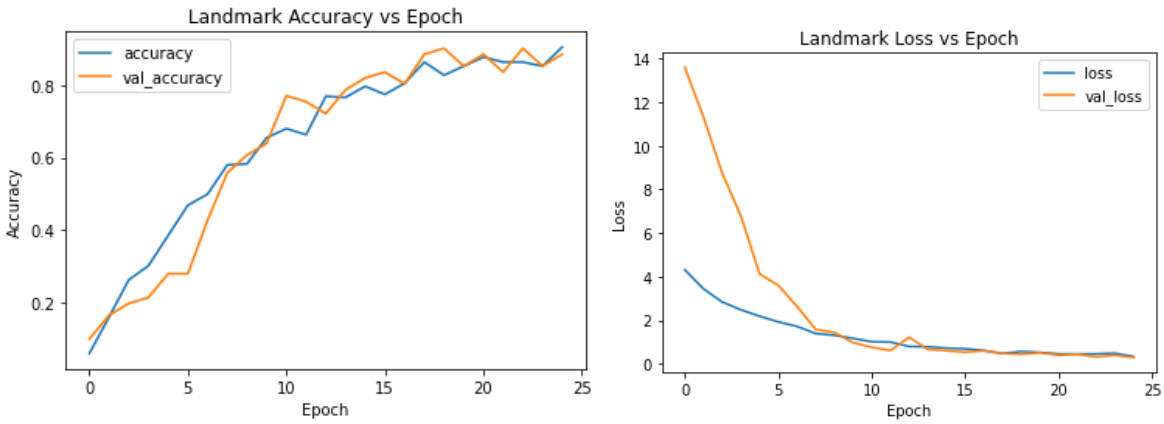

Fig 5.2 Accuracy/Loss v/s Epochs GRAPHs for Landmark Classification

6. CONCLUSION

By comparing the results of all the 3 models we have developed, we feel that the model selection depends on the requirements. For the dataset provided to us, we feel that using Multi-Modal Learning by feeding the results of one model into another would be a better solution as we might be able to capture more complex relationships between the data, potentially leading to more accurate and robust predictions. However, the implementation may be more complex than the other methods, and the benefits of multi-modal learning would need to be carefully evaluated on the specific dataset. But this method still has its cons like prediction accuracy, computational resources, and ease of implementation.

7. FUTURE SCOPE

We must further investigate advanced model architectures, attention mechanisms, or optimization techniques to enhance the performance of the model on both category and landmark classification tasks. The dataset has to be expanded with new categories and landmarks to generalize. The developed model has real-time applications in the field of tourism, image retrieval, and virtual reality. The model can be further improved by incorporating more advanced techniques, such as ensemble methods, to achieve even higher accuracy in landmark recognition. It can be used for the development of virtual reality applications providing a realistic experience of visiting famous landmarks around the world.

8. BIBLIOGRAPHY

1. Yang, S., Xiao, W., Zhang, M., Guo, S., Zhao, J., Shen, F. (2022, April 19). Image Data Augmentation for Deep Learning: A Survey. arXiv.org. Retrieved April 27, 2023, from https://arxiv.org/abs/2204.08610

2. Miller, S., Howard, J., Adams, P., Schwan, M., & Slater, R. (2020). Multi-Modal Classification Using Images and Text. SMU Data Science Review, 3(3), 6. https://scholar.smu.edu/cgi/viewcontent.cgi?article=1165&context=datasciencereview

3. Keras Documentation: Randomsearch Tuner. Keras. Retrieved April 27, 2023, from https://keras.io/api/keras_tuner/tuners/random/

4. Keras Documentation: VGG16 and VGG19. Keras. Retrieved April 27, 2023, from https://keras.io/api/applications/vgg/

5. Keras Documentation: Tf.keras.preprocessing.image.imagedatagenerator tensorflow V2.12.0. TensorFlow. Retrieved April 27, 2023, from https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/ImageDataGenerator