Scenic Images Classification using CNN

Naga Chandra Mouli Thammineni
University of Central Florida
na172658@ucf.edu

Raja Rajeshwari Porumalla
University of Central Florida
ra459414@ucf.edu

1 ABSTRACT

For this project, we created an image categorization model Convolution Neural Network models to group scenic images into six classes: mountain, street, glacier, buildings, sea, and forest. We investigated several methods, such as an ensemble of neural networks, a simple Convolutional Neural Network (CNN), and feature extraction with VGG16. Our experiments showed that the ensemble model achieved the best performance, with an accuracy of 87.4 percent on the test set. The key contributions of this work are (1) a comprehensive comparison of different deep learning approaches for scenic image classification, (2) insights into the strengths and weaknesses of each model, and (3) a robust ensemble model that outperforms individual models.

2 KEYWORDS

Convolutional Neural Network (CNN), VGG16, Ensemble of Neural Networks, OverFitting, Fine Tuning imageNet, Image Classification, Deep Learning, Confusion Matrix.

3 INTRODUCTION

The rapid growth of digital imagery across several fields has increased the demand for strong and efficient image classification systems. The main aim of this project is to predict and classify images into predefined classes. This project addresses some of the challenges involved in categorizing the images into predefined classes by using cutting-edge machine learning techniques and utilizing Convolutional Neural Networks (CNN). The project involves pre-processing the dataset to enhance the model, followed by fine-tuning a pre-trained neural network for specific classification tasks, we also explore different architectures, hyperparameters, and optimization techniques for the performance of classification after successful implementation focus on applying the learned model in real-world applications in an effective and scalable manner.

4 DATASET

In this project, we are using data from Intel and Google, which contains around 25k images of size 150x150 distributed, which were further divided into six categories, such as buildings, forests, glaciers, mountains, sea, and streets. All the train, test, and prediction data are separated in each zip file. There are around 6k images in the training dataset, 3k in the test, and 3k in the prediction. The data can be accessed through a Python package.

5 RELATED WORK

Numerous studies have been conducted on the application of deep learning for image classification. Convolutional Neural Networks (CNNs) have emerged as a dominant approach, with architectures like AlexNet, VGG, and ResNet achieving state-of-the-art results on benchmark datasets. In the context of scenic image classification, researchers have explored the use of transfer learning, where pre-trained models on large-scale datasets, such as ImageNet, are fine-tuned on the target domain. Additionally, ensemble methods, which combine the predictions of multiple models, have been shown to improve classification performance. Our work builds upon these existing studies and investigates the effectiveness of different deep learning techniques, including a basic CNN, feature extraction using VGG16, and an ensemble of neural networks, for the specific task of scenic image classification.

6 APPROACH

A. Convolutional Neural Network (CNN) Model:

We are defining CNN model for the classification task, where the input images are expected to have dimensions of 150x150 pixels with 3 color channels (RGB), and the model outputs the probability distribution over 6 classes.

Steps Include:

- Build the model,
- Compile the model,
- Train / fit the data to the model,
- Evaluate the model on the testing set,
- Perform an error analysis of our model.

CNN Model Approach

We are building our model composed of different layers such as: Conv2D, MaxPooling2D, Flatten, Relu, and Softmax:

- Conv2D: (32 filters of size 3 by 3) The features will be "extracted" from the image.
- MaxPooling2D: The images get half-sized.
- Flatten: Transforms the format of the images from a 2d-array to a 1d-array of 150 150 3 pixel values.
- Relu : given a value x, returns max(x, 0).
- Softmax: 6 neurons, probability that the image belongs to one of the classes.

When compiling the model, we opt for the Adam optimizer, which combines the Momentum and RMSProp techniques. Momentum integrates past gradients for more effective updates, while RMSProp adjusts the learning rate based on the magnitude of gradients, ensuring stability. For the loss function, we use sparse categorical cross-entropy, ideal for single-label classification tasks. This setup equips the model with robust optimization capabilities for efficient training and accurate classification.

Figure 1: CNN Model Accuracy Comparison:

- (**Fig1**)In the accuracy plot, we can see that both the training and validation accuracy start low and gradually increase with more training epochs. The training accuracy increases more rapidly and reaches higher values compared to the validation accuracy, which is expected as the model is optimized on the training data.

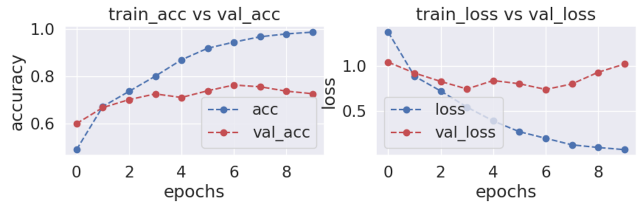
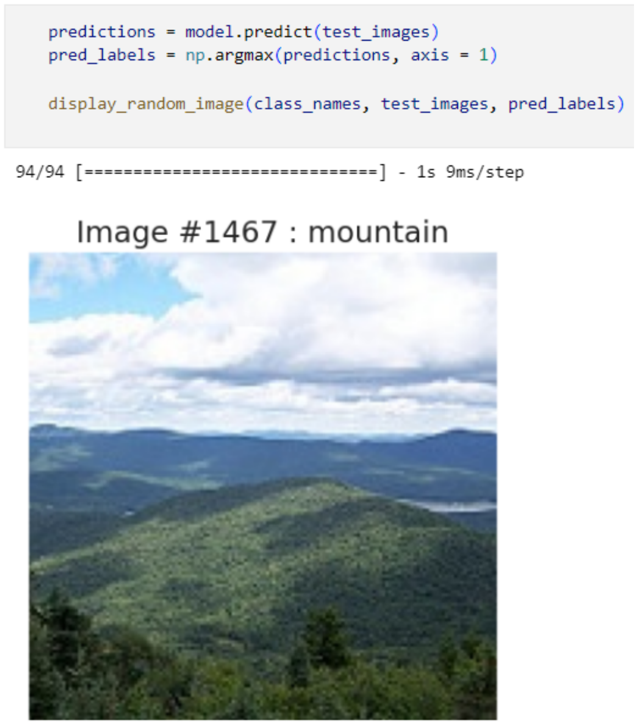


Figure 1: Accuracy comparison for CNN model

- In the loss plot, both the training and validation loss start high and decrease over the training epochs. The training loss decreases more rapidly and reaches lower values than the validation loss, which is again expected since the model is being optimized to minimize the loss on the training data.

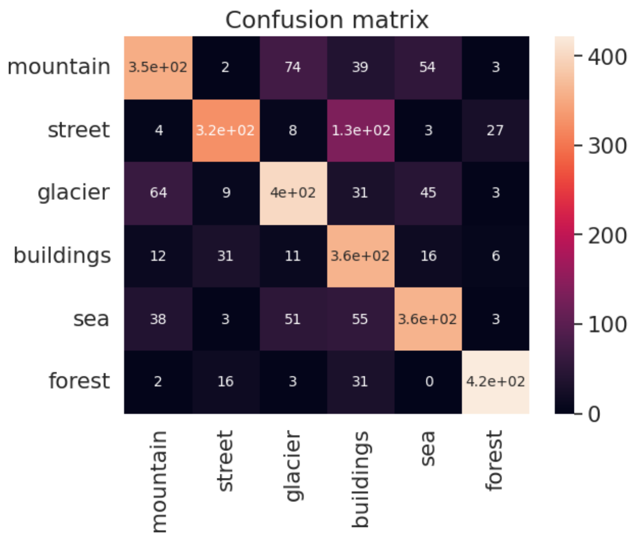


```
predictions = model.predict(test_images)
pred_labels = np.argmax(predictions, axis = 1)

display_random_image(class_names, test_images, pred_labels)
```

```
94/94 [==============================] - 1s 9ms/step
```

Figure 2: sample dataset image



	mountain	street	glacier	buildings	sea	forest
mountain	3.5e+02	2	74	39	54	3
street	4	3.2e+02	8	1.3e+02	3	27
glacier	64	9	4e+02	31	45	3
buildings	12	31	11	3.6e+02	16	6
sea	38	3	51	55	3.6e+02	3
forest	2	16	3	31	0	4.2e+02

Figure 3: Confusion Matrix

Confusion Matrix Conclusion

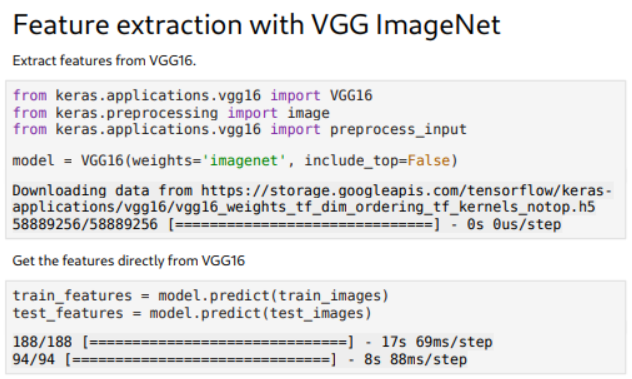
The Model (**Fig3**) has trouble with two kinds of images. The model has trouble with streets and buildings, As buildings are part of streets it is facing problems with them. Along with these it also has problems with the sea, glaciers, and mountains as well.

As our base model has a problem with those datasets, we are choosing another model (VGG) for better results

B. Feature Extraction with VGG16: To leverage the powerful feature extraction capabilities of pre-trained models, we used the VGG16 architecture pre-trained on the ImageNet dataset. We removed the top fully connected layers and used the remaining convolutional base to extract features from the input images. These features were then used to train a simple one-layer neural network for the final classification task.

VGG Model Approach: We implement the below steps to improve the results on our datasets.

- Feature extraction with VGG16 trained on ImageNet.
- Ensemble models of Neural Networks with the features extracted from VGG.
- Fine Tuning with VGG16 trained on ImageNet.



Feature extraction with VGG ImageNet

Extract features from VGG16.

```
from keras.applications.vgg16 import VGG16
from keras.preprocessing import image
from keras.applications.vgg16 import preprocess_input

model = VGG16(weights='imagenet', include_top=False)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
58889256/58889256 [==============================] - 0s 0us/step
```

Get the features directly from VGG16

```
train_features = model.predict(train_images)
test_features = model.predict(test_images)
```

```
188/188 [==============================] - 17s 69ms/step
94/94 [==============================] - 8s 88ms/step
```

Figure 4: B. Feature Extraction with VGG Imagenet

PCA Projection Analysis: We see that glacier and mountain points are very close to each other from (**Fig5**), as VGG sees them as very similar, and here We can observe there is no distinction between building and street.

Figure 6: VGG Model Accuracy Comparison

- The plot on the left shows the training accuracy (acc) and validation accuracy (value acc) over 8 epochs. The training accuracy increases steadily and reaches nearly 1.0 by the final epoch, indicating that the model fits the training data

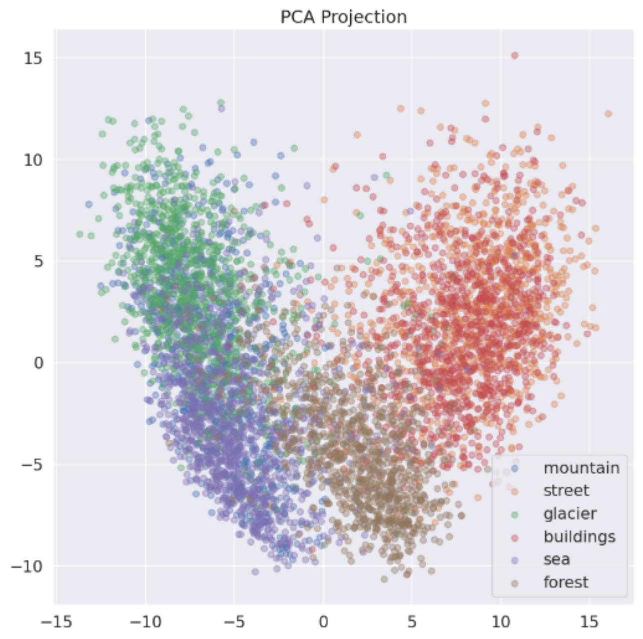


Figure 5: PCA Projection

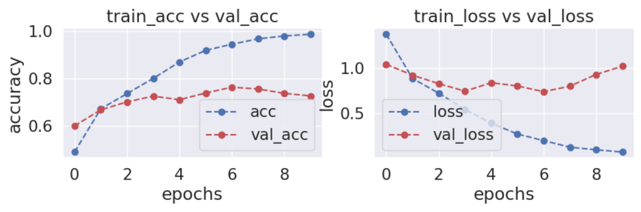


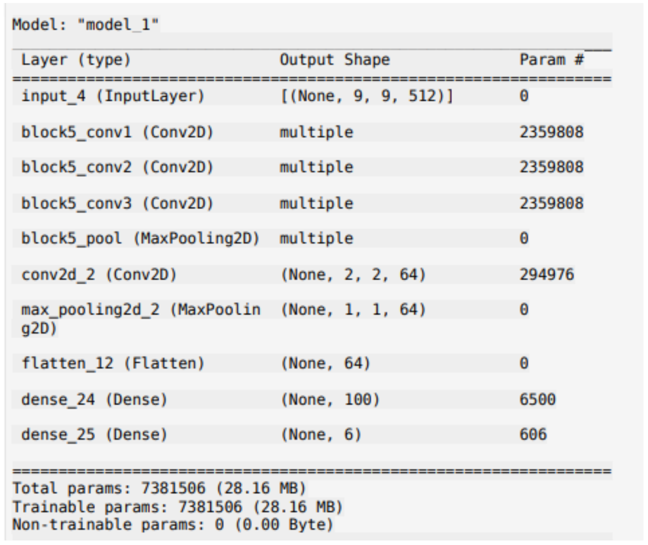
Figure 6: VGG Model Comparison

very well. However, the validation accuracy plateaus around 0.65, which suggests the model may be overfitting to the training data and not generalizing well to unseen data.

- The plot on the right shows the training loss (loss) and validation loss (value loss) over the same 8 epochs. The training loss decreases consistently as the model learns from the training data. However, the validation loss does not decrease as smoothly and starts to increase after around 4 epochs, which is another potential sign of overfitting (**Fig6**)

C. Ensemble of Neural Networks: We implemented an ensemble of neural networks to further improve the classification performance. We trained 10 individual models, each with a different number of units in the hidden layer (randomly selected between 50 and 100). Each model was trained on a random subset of the training data, containing 800 percent of the samples. The final prediction was obtained by averaging the outputs of the ensemble. Each Neural Network will be trained on random subsets of the training dataset. Each subset contains max samples samples.

D. Fine Tuning VGGImageNet Fine-tuning ImageNet involves adapting pre-trained convolutional neural networks (CNNs) on the ImageNet dataset to new tasks or datasets. It utilizes learned features from ImageNet to boost performance on specific tasks with limited labeled data. By replacing or retraining the final layers while keeping earlier layers fixed, fine-tuning optimizes the model for the target task. This technique is widely used across various domains, leveraging the rich feature representations learned from ImageNet for improved performance in tasks like object detection, segmentation, and natural language processing (**Fig7**)



```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_4 (InputLayer)        [(None, 9, 9, 512)]       0

 block5_conv1 (Conv2D)       multiple                  2359808

 block5_conv2 (Conv2D)       multiple                  2359808

 block5_conv3 (Conv2D)       multiple                  2359808

 block5_pool (MaxPooling2D)  multiple                  0

 conv2d_2 (Conv2D)           (None, 2, 2, 64)          294976

 max_pooling2d_2 (MaxPoolin  (None, 1, 1, 64)          0
 g2D)

 flatten_12 (Flatten)        (None, 64)                0

 dense_24 (Dense)            (None, 100)               6500

 dense_25 (Dense)            (None, 6)                 606

=================================================================
Total params: 7381506 (28.16 MB)
Trainable params: 7381506 (28.16 MB)
Non-trainable params: 0 (0.00 Byte)
```

Figure 7: Fine tuning VGG

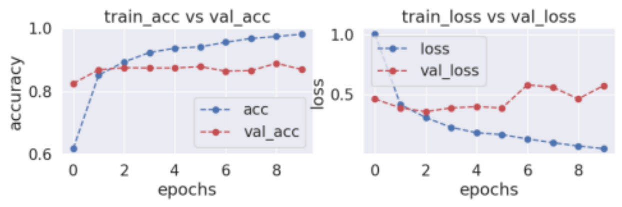


Figure 8: Fine tuning Comparison

Figure8: Finetuning Accuracy comparision

- The plot on the left displays the training accuracy (acc) and validation accuracy (value acc). The training accuracy starts around 0.6 and increases rapidly, reaching nearly 1.0 by epoch 8. However, the validation accuracy remains relatively low, plateauing around 0.65. This divergence between training and validation accuracy indicates the model is overfitting to the training data and failing to generalize well to unseen data.
- The plot on the right shows the training loss (loss) and validation loss (value loss). The training loss decreases steadily as expected, but the validation loss does not decrease smoothly. Instead, it fluctuates and starts increasing after around epoch 4, which is another potential sign of overfitting.(**Fig8**)

7 EXPERIMENT

We evaluated the performance of the three approaches on the scenic image classification dataset, which consists of 6,000 training images and 3,000 test images, equally distributed across the six classes.

Baseline Model Comparison We first compared the performance of the basic CNN model with the feature extraction approach using VGG16. The results are summarized in the table below:

Table 1: Model Comparison for Accuracy

Model	Accuracy (Test Set)
CNN	74.07%
VGG16 + One-layer NN	86.67%
Ensemble Neural networks	87.4%

- **CNN Model:**
  - The basic Convolutional Neural Network (CNN) model achieved an accuracy of 74.07
- **VGG16-based Model:**
  - The VGG16-based approach, where the extracted features were used to train a simple one-layer neural network, achieved an accuracy of 86.67 percent on the test set.This VGG16-based model significantly outperformed the basic CNN model.
- **Ensemble of Neural Networks:**
  - The ensemble of 10 neural networks achieved an accuracy of 87.4 percent on the test set, outperforming both the basic CNN and the VGG16-based models.
  - **Fine Tuning VGG imageNet**
  - With fine tuning we are getting nearly 87 percent Accuracy which is also an improved mode to the baseline model.

8 CONCLUSION AND FUTURE WORK

In this project, we explored the use of Convolution Neural Network techniques for scenic image classification. Initially, we used a base model taking Conv2D, MaxPooling2D, Flatten, Relu, and Dense activation layers with it we got 74.07 percent accuracy. As we are having problems with certain datasets such as sea, glaciers, buildings, and streets. we selected another model that is feature extraction with VGG Imagenet. By visualizing the features through PCA followed by training we got 86.67 percent accuracy. We also tried another model called ensemble neural networks, Each Neural Network will be trained on random subsets of the training dataset here An ensemble of neural networks, built upon the feature extraction capabilities of the VGG16 model, achieved the best performance, with an accuracy of 87.4 percent on the test set which outperformed all models. Along with these we also performed Fine-tuning VGG ImageNet to improve the accuracy of datasets with it we got an accuracy of nearly 87 percent.

The key contributions of this work are: A comprehensive comparison of different deep learning approaches for scenic image classification, including a basic CNN, feature extraction using VGG16, and an ensemble of neural networks. Insights into the strengths and weaknesses of each model, which can guide future research and development in this domain.

- A robust ensemble model that outperforms individual models, showcasing the benefits of combining multiple neural networks for improved classification accuracy.

For future work, we plan to investigate the following:

- Exploring more advanced ensemble techniques, such as weighted averaging or stacking, to further enhance the model's performance.
- Experimenting with different pretrained models such as ResNet,Inception and their impact on the feature extraction and classification tasks.
- incorporating additional data augmentation techniques to improve the model's generalization capabilities.
- Analyzing the model's performance on specific image categories to identify areas for further improvement.
- The findings of this project can be valuable for researchers and practitioners working on similar image classification tasks, particularly in the context of scenic image analysis.

9 REFERENCES

- Krizhevsky, A., Sutskever, I.and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105.
- Simonyan, K.and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
- Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are featured in deep neural networks?. Advances in neural information processing systems, 27.
- Dietterich, T. G. (2000). Ensemble methods in machine learning. In International workshop on multiple classifier systems (pp. 1-15). Springer, Berlin, Heidelberg.
- Rokach, L. (2010). Ensemble-based classifiers. Artificial Intelligence Review, 33(1), 1-39.
- Karen Simonyan, Andrew Zisserman. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
- Keiichi Kuroki, Hisashi Kashima. (2018). An Empirical Study on Hyperparameters of Deep Neural Networks for Large-Scale Image Classification. IEEE Access, 6, 19785-19798.
- Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich. (2015). Going Deeper with Convolutions. Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR).