Classification of Foggy Images

Student - Manisha Chaurasia (221345)* and Supervisor - Prof. Mahendra K Verma†
Department of Physics, Indian Institute of Technology Kanpur, Uttar Pradesh 208016, India
(Dated: April 28 2024)

Fog prediction is essential for transportation and agriculture. We collected data manually through cameras and preprocessed it for machine learning analysis. Model 1 served as the baseline for subsequent comparisons. Models 2 and 3, divided by day and night data, lacked significant improvement. Consequently, we balanced the fog:no-fog and day: night image ratio and employed data augmentation for an 80:20 training-testing split. Remarkably, Model 4, trained on redistributed data, outperformed the baseline. Model-5, on augmented data, didn't meet expectations. Model-6, using VGG-16 transfer learning, performed outstanding. Overall, Model-6 excelled.

INTRODUCTION

Fog is a meteorological phenomenon consisting of a visible mass of water droplets or ice crystals suspended near the Earth's surface. It reduces Visibility. Visibility refers to the distance one can see in the atmosphere, typically it measured horizontally. The reason why it's prediction is essential -

- Transportation safety - Fog prediction can reduce road accidents.
- Air travel - Prediction can manage Air travel much more effectively.
- Agriculture - Fog can affect the crop's health and management. Accurate forecast helps farmers plan their activities.

UNDERSTANDING DATA

Data was collected by Professor Shivam Tripathi and his research assistant, Roshan Banbariya. Data is in image format. Data has foggy and non-foggy images captured by the camera. Along with the Data also, an Excel file is there which have parameters -

- date_time- As the column name suggests, it contains the date and time of the image.
- Vis:- having visibility value of 10 min interval.
- Status:- Having value 0 or 1, 0 shows that the image is non-foggy, and one is foggy.
- Category:- It contains the integer value 0-5, which shows the intensity of fog; this column can be used in multi-level image classification.

The fog prediction can be made in three ways-

- Classification - The aim is to find whether or not there is fog.
- Multi-level Classification - Here, the output is Light, Heavy, or No fog. Numerically, its range is from 0-5.
- Regression - The outcome will be any number from 0-1, which represents the fog index value, which shows the impact of fog.

I was instructed to work on the Classification problem. So, I used the Excel file to separate the images according to their status as a folder in my device.

Data Sample

FIG. 1. Foggy Image

FIG. 2. Non-Foggy Image

Data duration:

- Training date: January 10, 2023, to January 31, 2023, totalling 21 days of data.
- Testing data: January 7, 2023, to January 10, 2023.
- Validation date: December 19, 2022, to January 7, 2023.

The data interval is 10 minutes, and data is taken for November, December, January and February. By selecting only these months, the data Imbalance problem can be eradicated.

METHODOLOGY

Problem statement - I aim to classify the foggy and non-foggy images successfully.

Evaluation metrics - I use Accuracy and Number of Misclassified Images to evaluate the results.

Preprocessing - We need to prepare the data for classification for training and testing purposes. Data is separated into different folders according to their status using Python code. Code creates two folders, fog and no fog, for each dataset - training, testing and Validation and names them by their date-time and visibility value so that things will be easy while checking it through the model. Initially, Data distribution is as follows -

	Foggy Images	**Non-Foggy Images**
Training Data	3570	3785
Testing Data	736	345
Validation Data	805	1355

TABLE I. Initial Data

FOG MODELING

Convolution Neural network - I used the CNN model [2] to classify the images; before training, all images were resized into sizes (128,128) and then normalized. Data augmentation is employed to reduce overfitting. The parameters of data augmentation are zoom_range = 0.2, shear_range = 0.2, and horizontal_flip = True. Augmented data is used during model training, not for testing and Validation.

Description of CNN Model - Layers that are used in CNN architecture - [3]

- **Convolutional layer (conv) -**
  This layer extracts features from input images. It slides a filter of Size 3×3 over the input, computing dot products. This produces a feature map revealing image details like edges and corners. This map is then used in subsequent layers to learn more features. These layers maintain pixel spatial relationships, improving feature extraction.
  Parameters of Convolutional layer- Number of filters, Size of filters, stride, padding, Activation functions - ReLU (Rectified Linear Activation), Sigmoid, Tanh (Hyperbolic Tangent), Softmax, Leaky ReLU, ELU (Exponential Linear Unit) and Swish.

- **Pooling layer - [4]**
  The Pooling Layer typically follows a Convolutional Layer and aims to reduce the size of the convolved feature map, reducing computational costs. It operates independently on each feature map and summarizes features generated by the convolutional layer. Max Pooling selects the most significant element from each feature map. This layer bridges the Convolutional Layer and the Fully Connected (FC) Layer, aiding the network in feature recognition and computation reduction.
  Parameters of Max Pooling - Pool size, Stride and Padding.

- **Fully Connected Layer - [1]**
  The Fully Connected (FC) layer links neurons between different layers using weights, biases, and neurons. Typically positioned before the output layer, it forms part of the final layers in a CNN architecture. Here, input images from previous layers are flattened and passed to the FC layer. Subsequently, the flattened vector undergoes additional FC layers where mathematical operations typically occur, initiating the classification process. Connecting two layers enhances performance compared to a single connected layer. These layers reduce the need for human supervision.
  Parameters of Fully Connected Layer - Number of neurons, Weights, Biases and Activation functions.

The best output is found on this provided CNN architecture for our Input in all those variable parameters. It consists of a total of eight layers.

The first layer consists of an input image with dimensions of 128×128. It is convolved with 32 filters of size 3×3, with reLU Activation function resulting in 126x126x32. The second layer is a Max-Pooling operation, which filters size 2×2 and stride 2. Hence, the resulting image dimension will be 63x63x32.

Similarly, the third layer involves a convolution operation with 64 filters of size 3×3 with reLU Activation function followed by a fourth max-pooling layer with a similar filter size of 2×2 and stride of 2. Thus, the resulting image dimension will be reduced to 30x30×64. Again, the fifth layer has a convolution operation with 128 filters of size 3×3 with reLU Activation function followed by a sixth max-pooling layer with a similar filter size of 2×2 and stride of 2. So, the output image

dimension will be 14x14×128.

The seventh layer is also a fully connected layer with 128 neurons, and the final eighth layer is the sigmoid output layer with two possible classes.

CNN Architecture

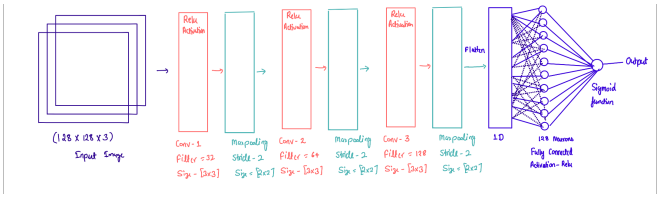

FIG. 3. CNN Architecture

Python Implementation of the Architecture

```
    # Build the CNN model
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(1, activation='sigmoid')
])
```

FIG. 4. Python Code

Model Summary

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 126, 126, 32)      896
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 63, 63, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 61, 61, 64)        18496
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 30, 30, 64)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 28, 28, 128)       73856
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 14, 14, 128)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 25088)             0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               3211392
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 129
=================================================================
Total params: 3,304,769
Trainable params: 3,304,769
Non-trainable params: 0
_________________________________________________________________
```

FIG. 5. Model Summary

After creating a good architecture, different models were developed-

- **Model-1:-** It's trained on the training and testing dataset. During training, the augmented images were used instead of the actual ones.

- **Models -2 and 3:-** Data have day and night images, so all the data is redistributed accordingly. Model -2 is trained on the data of Day data distribution as below -

	Foggy Images	**Non-Foggy Images**
Training Data	806	2423
Testing Data	252	345
Validation Data	84	854

TABLE II. Day Data

Model -3 is trained on the data of Night data distribution as below -

	Foggy Images	**Non-Foggy Images**
Training Data	2764	1362
Testing Data	484	0
Validation Data	721	501

TABLE III. Night Data

- **Model -4 and 5:-** It is prepared by redistributing the whole data but without dividing them into Day and night images-
Model-4 is trained on the training dataset, whereas Model-5 is trained on an Augmented dataset. Data augmentation is employed here to increase the number of data such that the ratio of train: the test will be 80:20 while keeping the Day: night and fog:no-fog as 50:50 for the training and testing dataset. The new distribution is as follows-

	Status	Day Images	Night Images
Training Data	**Fog**	614	614
	No-Fog	614	614
Augumented Data	**Fog**	1280	1280
	No-Fog	1280	1280
Testing Data	**Fog**	300	300
	No-Fog	300	300
Validation Data	**Fog**	228	2708
	No-Fog	3055	949

TABLE IV. New Distributed Data

For data augmentation, the employed features are-

1. rotation_range = 40
2. width_shift_range = 0.2
3. height_shift_range = 0.2
4. shear_range = 0.2
5. zoom_range = 0.2
6. horizontal_flip = True
7. fill_mode = 'nearest'

- **Model-6:-** Pre-trained model VGG-16 is used while freezing the trained weights to extract the relevant features of the input. Trained on training and testing dataset of new data distribution and validating on its validation Data set.

Model Summary of VGG-16

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 150, 150, 3)]     0
block1_conv1 (Conv2D)        (None, 150, 150, 64)      1792
block1_conv2 (Conv2D)        (None, 150, 150, 64)      36928
block1_pool (MaxPooling2D)   (None, 75, 75, 64)        0
block2_conv1 (Conv2D)        (None, 75, 75, 128)       73856
block2_conv2 (Conv2D)        (None, 75, 75, 128)       147584
block2_pool (MaxPooling2D)   (None, 37, 37, 128)       0
block3_conv1 (Conv2D)        (None, 37, 37, 256)       295168
block3_conv2 (Conv2D)        (None, 37, 37, 256)       590080
block3_conv3 (Conv2D)        (None, 37, 37, 256)       590080
block3_pool (MaxPooling2D)   (None, 18, 18, 256)       0
block4_conv1 (Conv2D)        (None, 18, 18, 512)       1180160
block4_conv2 (Conv2D)        (None, 18, 18, 512)       2359808
block4_conv3 (Conv2D)        (None, 18, 18, 512)       2359808
block4_pool (MaxPooling2D)   (None, 9, 9, 512)         0
block5_conv1 (Conv2D)        (None, 9, 9, 512)         2359808
block5_conv2 (Conv2D)        (None, 9, 9, 512)         2359808
block5_conv3 (Conv2D)        (None, 9, 9, 512)         2359808
block5_pool (MaxPooling2D)   (None, 4, 4, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
```

FIG. 6. Model Summary of VGG-16

EVALUATION CRITERIA

Accuracy - The accuracy can be calculated using the formula:

$$\text{Accuracy} = \frac{\text{Number of correctly classified images}}{\text{Total number of images classified}} * 100\%$$

Misclassified Images - Number of misclassified images

RESULTS

Result of Base Model

		Accuracy	Misclassified
Training Data	**Fog**	93.8	219
	No-Fog	97.8	80
Testing Data	**Fog**	96	736
	No-Fog	100	0
Validation Data	**Fog**	65.4	278
	No-Fog	99.4	7

TABLE V. Model-1

Result of Day Model

		Accuracy	Misclassified
Training Data	**Fog**	97.9	97
	No-Fog	75.2	1197
Testing Data	**Fog**	99	1
	No-Fog	86	47
Validation Data	**Fog**	86.9	11
	No-Fog	96.7	28

TABLE VI. Model-2

Result of Night Model

		Accuracy	Misclassified
Training Data	**Fog**	95.1	135
	No-Fog	98.7	32
Testing Data	**Fog**	68	152
	No-Fog	99.7	1
Validation Data	**Fog**	64.2	258
	No-Fog	66.2	169

TABLE VII. Model-3

Result of Modified Data Model trained on training data

		Accuracy	Misclassified
Training Data	**Fog**	97.5	30
	No-Fog	81.8	223
Testing Data	**Fog**	94.8	31
	No-Fog	73.6	158
Validation Data	**Fog**	96.8	105
	No-Fog	74.1	954

TABLE VIII. Model-4

Result of Modified Data Model trained on Augmented training data

		Accuracy	Misclassified
Training Data	**Fog**	92.1	96
	No-Fog	76.3	290
Testing Data	**Fog**	83.1	101
	No-Fog	51.1	293
Validation Data	**Fog**	92.3	250
	No-Fog	70.0	1094

TABLE IX. Model-5

Result of VGG-16 Model

		Accuracy	Misclassified
Training Data	**Fog**	96.6	41
	No-Fog	86.7	163
Testing Data	**Fog**	91.5	51
	No-Fog	67.6	194
Validation Data	**Fog**	97.7	75
	No-Fog	81.1	691

TABLE X. Model-6

ANALYSIS & CONCLUSION

Comparing the performance of Model 1 as the baseline, we observe that Models 2 and 3 do not demonstrate

significant improvement. This suggests that partitioning the data into day and night categories does not enhance performance.

However, we cannot disregard this aspect, so we redistributed the entire dataset, maintaining a balanced ratio of foggy to non-foggy and day-to-night images at 50:50. Additionally, we applied data augmentation to achieve an 80:20 ratio for training and testing data.

Model 4, trained on the redistributed data, exhibits notably improved performance compared to the baseline. Surprisingly, Model 5, trained on augmented data with an 80:20 ratio, did not perform better than Model 4, failing to resolve overfitting issues as anticipated.

Model-6, employing transfer learning with the pre-trained VGG-16 model, preserves the convolutional layer weights for feature extraction from input images while training the fully connected layers. This approach yields performance superior to all models.

In conclusion, Model 6 achieves the best performance among the evaluated models.

* manishac22@iitk.ac.in
† mkv@iitk.ac.in

[1] Andriy Burkov. *The hundred-page machine learning book*, volume 1. Andriy Burkov Quebec City, QC, Canada, 2019.
[2] Francois Chollet. Deep learning with python, vol. 1. *Greenwich, CT: Manning Publications CO*, 2017.
[3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.
[4] Aurélien Géron Hands-On Machine Learning. with scikit-learn, keras, and tensorflow: Concepts, tools, and techniques to build intelligent systems, 2019.