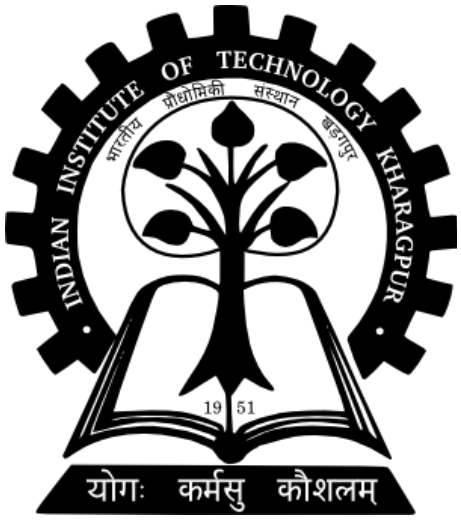
DESIGN LAB

Classification of Twitter image having Infrastructural damage



14CS30029

Rohan Rajak

Instructor: Dr.Soumya Kanti Ghosh

Professor , Computer Science and Engineering

Contents

Abstract:

During natural and man-made disasters, people use social media platforms such as Twitter to post textual and multimedia content to report updates about injured or dead people, infrastructure damage, and missing or found people among other information types. Studies have revealed that this online information, if processed timely and effectively, is extremely useful for humanitarian organizations to gain situational awareness and plan relief operations. In addition to the analysis of textual content, recent studies have shown that imagery content on social media can boost disaster response significantly. Despite extensive research that mainly focuses on textual content to extract useful information, limited work has focused on the use of imagery content or the combination of both content types. One of the reasons is the lack of labeled imagery data in this domain. Despite extensive research that mainly focuses on social media text messages, limited work has focused on the use of images to boost humanitarian aid. One reason that hinders the growth of this research line is the lack of ground-truth data. There exist a few repositories such as CrisisLex (Olteanu et al. 2014) and Crisis NLP (Imran, Mitra, and Castillo 2016) which offer several Twitter datasets from natural and man-made disasters, but all of them share only textual content annotations.

Dataset Overview:

The CrisisMMD multimodal Twitter dataset consists of several thousands of manually annotated tweets and images collected during seven major natural disasters including earthquakes, hurricanes, wildfires, and floods that happened in the year 2017 across different parts of the World. The provided datasets include three types of annotations (for details please refer to our paper [1]):

** Task 1: Informative vs Not informative
  * Informative
  * Not informative
  * Don't know or can't judge
** Task 2: Humanitarian categories
  * Affected individuals
  * Infrastructure and utility damage
  * Injured or dead people
  * Missing or found people
  * Rescue, volunteering or donation effort
  * Vehicle damage
  * Other relevant information
  * Not relevant or can't judge
** Task 3: Damage severity assessment
  * Severe damage
  * Mild damage
  * Little or no damage
  * Don't know or can't judge

I worked on task 2 to classify if there is any **Infrastructure and utility damage** in a given twitter image.

Image data processing:

The most common image data input parameters are the number of images, image height, image width, number of channels, and number of levels per pixel. Typically we have 3 channels of data corresponding to the colors Red,Green, Blue (RGB) Pixel levels are usually [0,255].Back to our data-set. Sampling a few pictures randomly we see each image in the data-set appears to have dimensions greater than 320 by 240 pixels, which does simplify things. But there were still few hurdles that need to be tackled.

Hurdles:

- Each image were of different dimensions, some were very low dimension icons and others were HD image with large pixel matrix.

- There were a total of 18126 images of both positive and negative classes but the positive class were very less.

Approach 1:

- **Taking all the images of dimension**

  $w > 320$ & $h > 240$

  works as this is just the size that would eliminate all small thumbnail images which won't provide much info for classification.

- **For solving the less data ratio problem, I just dropped the negative class which was large in number by a fraction of 0.7, to make both the classes almost similar.**

Approach 2:

1. **This part remains same.**

   3 channels RGB, pixel levels in range [0–255]

   Dimension: **320 x 240 x 3**

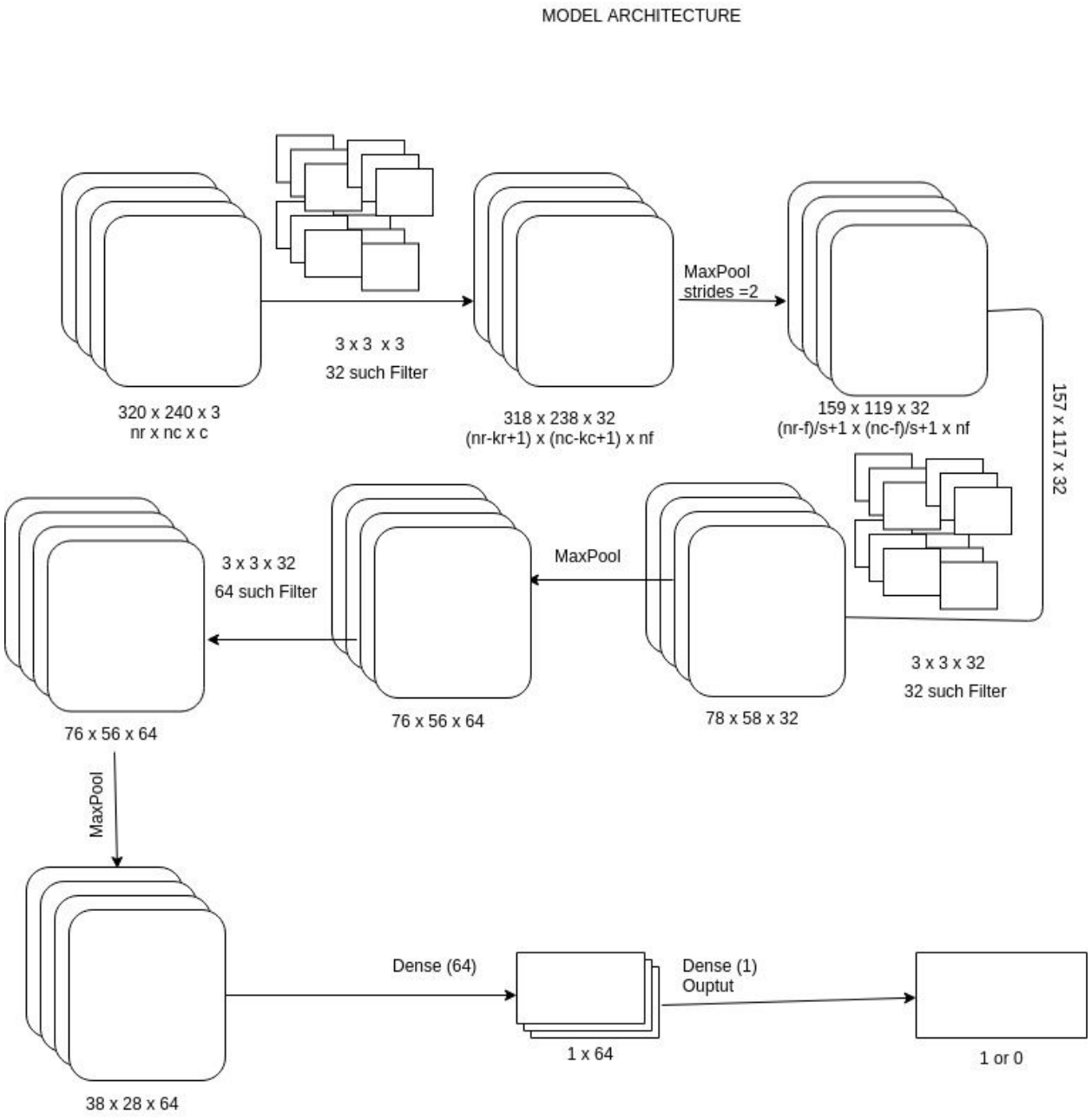
2. **But here rather randomly dropping the data which could have valuable info, to match the number of data point for both the classes I rotated the positive classed image by angle angle of -45 and +45 degrees. (And this is intuitive as the image clicked can be tilted sometimes)**

Model Summary:

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 318, 238, 32)	896
activation_1 (Activation)	(None, 318, 238, 32)	0
max_pooling2d_1 (MaxPooling2	(None, 159, 119, 32)	0
conv2d_2 (Conv2D)	(None, 157, 117, 32)	9248
activation_2 (Activation)	(None, 157, 117, 32)	0
max_pooling2d_2 (MaxPooling2	(None, 78, 58, 32)	0
conv2d_3 (Conv2D)	(None, 76, 56, 64)	18496
activation_3 (Activation)	(None, 76, 56, 64)	0
max_pooling2d_3 (MaxPooling2	(None, 38, 28, 64)	0
flatten_1 (Flatten)	(None, 68096)	0
dense_1 (Dense)	(None, 64)	4358208
activation_4 (Activation)	(None, 64)	0
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 1)	65
activation_5 (Activation)	(None, 1)	0

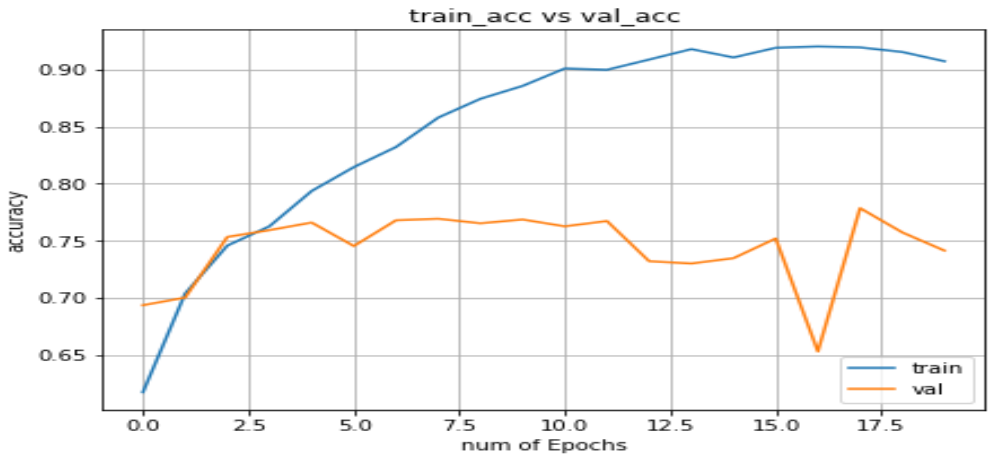
Total params: 4,386,913
Trainable params: 4,386,913
Non-trainable params: 0

Model detailed Architecture

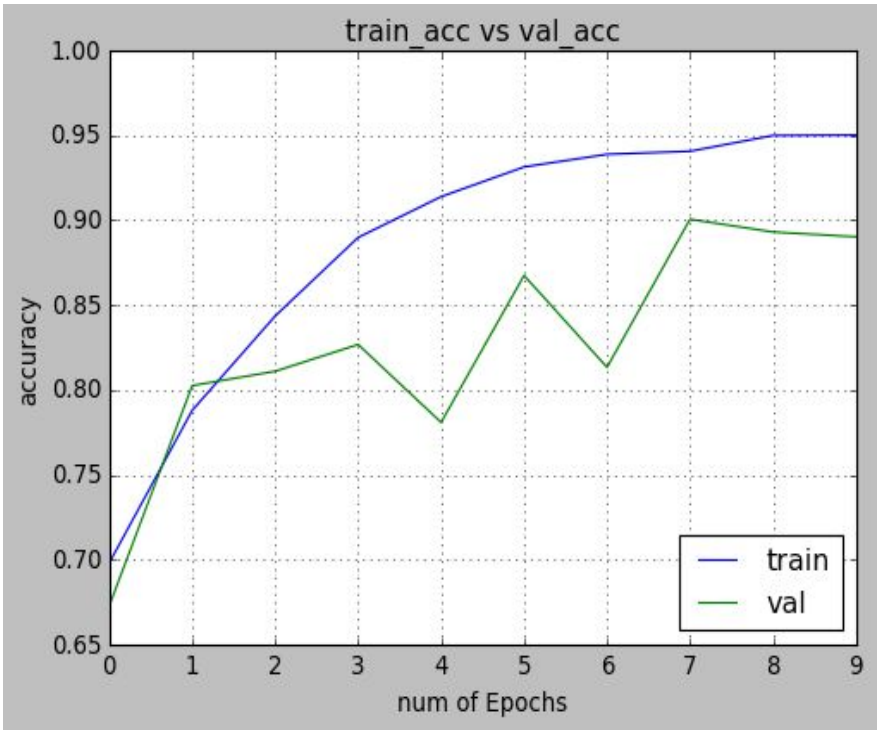


Result:

Approach 1:



Approach 2:



Conclusion:

The second approach does a much better job hence I save the model and any image can be classified as having a infrastructure damage with 89% correctness.The same method can be used as a baseline for the classification other tasks such as

* Affected individuals
  * Injured or dead people
  * Missing or found people
  * Vehicle damage

But the parameters and the structure of processed data feeding into the model should be taken care accordingly.

The model is doing a pretty good job with unseen images found in internet and snapshot of a image is given below with correct classification.

This project can be extended by clubbing with a text classifier of the tweets with the images.

Practical trial:

```
              metrics=['accuracy'])

# predicting images
from IPython.display import Image,display
display(Image(filename='test02.jpg'))

img = image.load_img('test02.jpg', target_size=(img_width, img_height))

x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)

images = np.vstack([x])
classes = model.predict_classes(images)
print ('This image has been classified as class  ',classes.flatten())
```

```
This image has been classified as class   [1]
```