Documents Skew Detection

Introduction

We all would have stumped on a problem where the documents that we have will be misaligned, skewed and also could be wrapped. A lot of images scanners will ask us to rotate the images by ourselves or ask us to choose the four points for modification of perspective.
But what is meant exactly by skew, it's any deviation of the image from that of the original document which is not parallel to the horizontal or vertical.
Skew corrections remain one of the vital parts in Document processing. And there are three types of skew in image:

- Global skew: this come when document have common degree angle orientation
- Multiple skew: documents have different degree of orientation in the different contents
- Non-uniform text line skew: when documents contain several orientations in the single line.

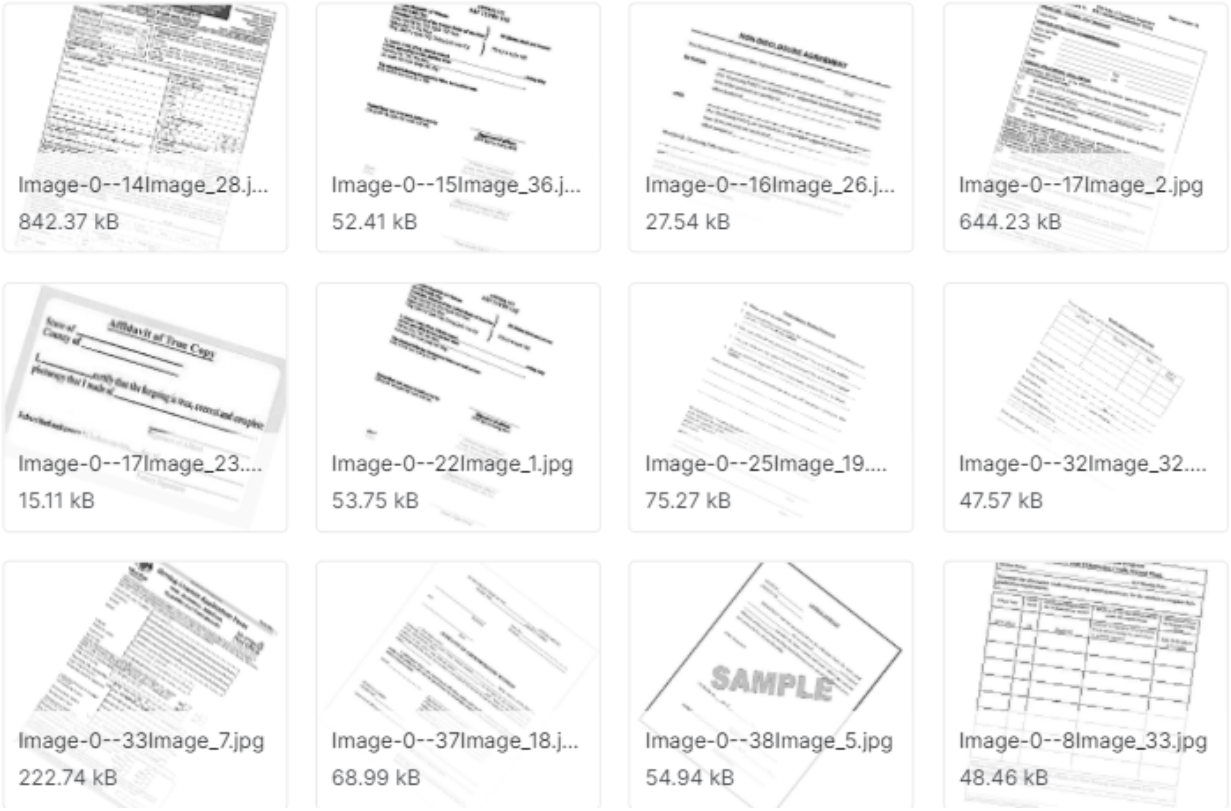
Estimating and rectifying the orientation angle of any image is a pretty challenging task. Problem of skewed documents degrade the performance of OCR and image analysis system so to detect and correct of skew angle is important step of preprocessing
of document analysis.
In order to solve this problem we have various techniques to solve this problem like Projection profile, Hough Transform, Nearest Neighbor connectivity, Fourier transform , and linear Regression analysis and mathematical morphology. But we are going to work on this problem using Deep Learning approaches.

Data Description

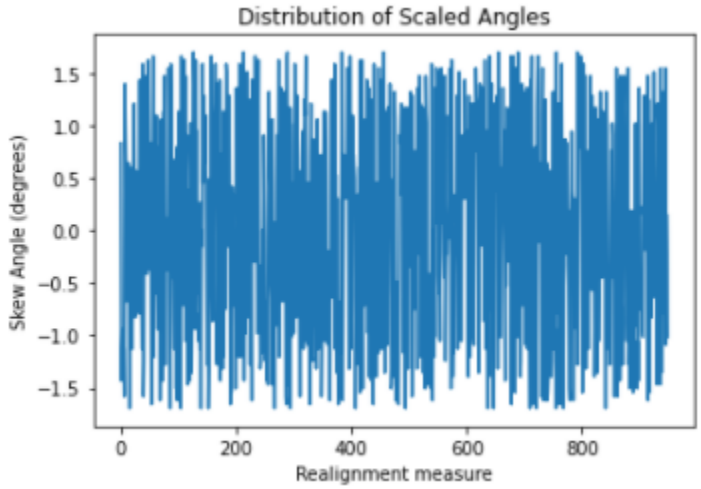
The dataset from kaggle (rdocuments) consists of a CSV file that contains the images "id" which is the image name and their angle of rotation in degrees, with a file length of 950 rows (no. of images).

	id	angle
0	Image-0--22Image_1.jpg	-22
1	Image-0-38Image_10.png	38
2	Image-0-30Image_11.jpg	30
3	Image-0-25Image_12.JPG	25
4	Image-0-29Image_13.png	29

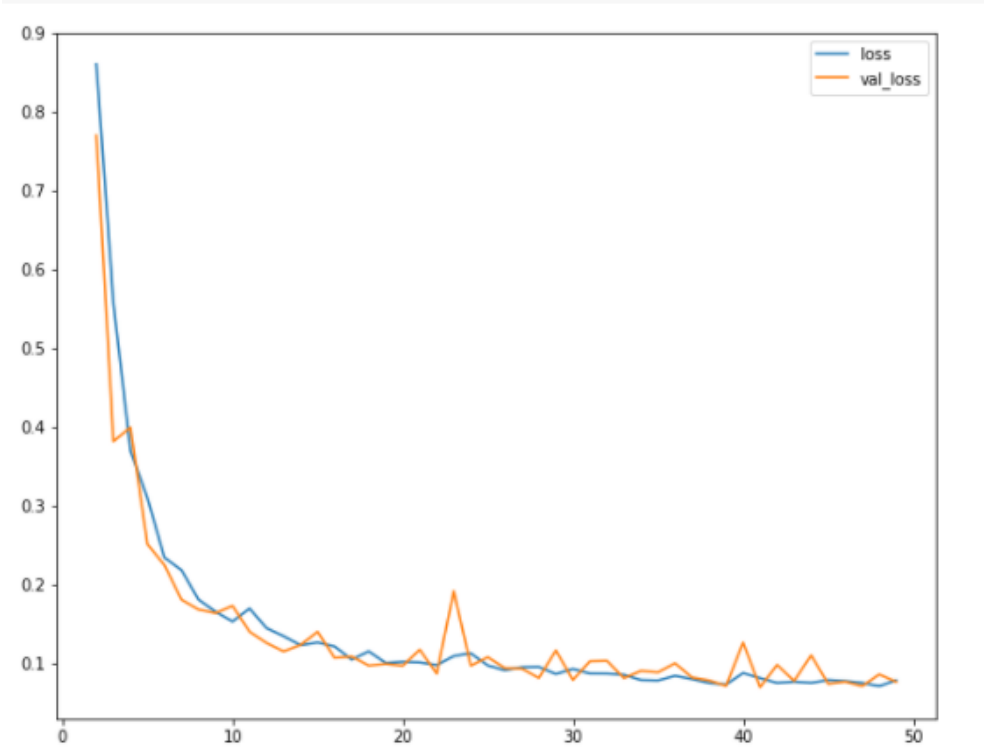
And a Folder of Images “rdocuments” contains 950 images of skewed documents. This is a sample of the images:



The distribution of scaled angles in skewed images:

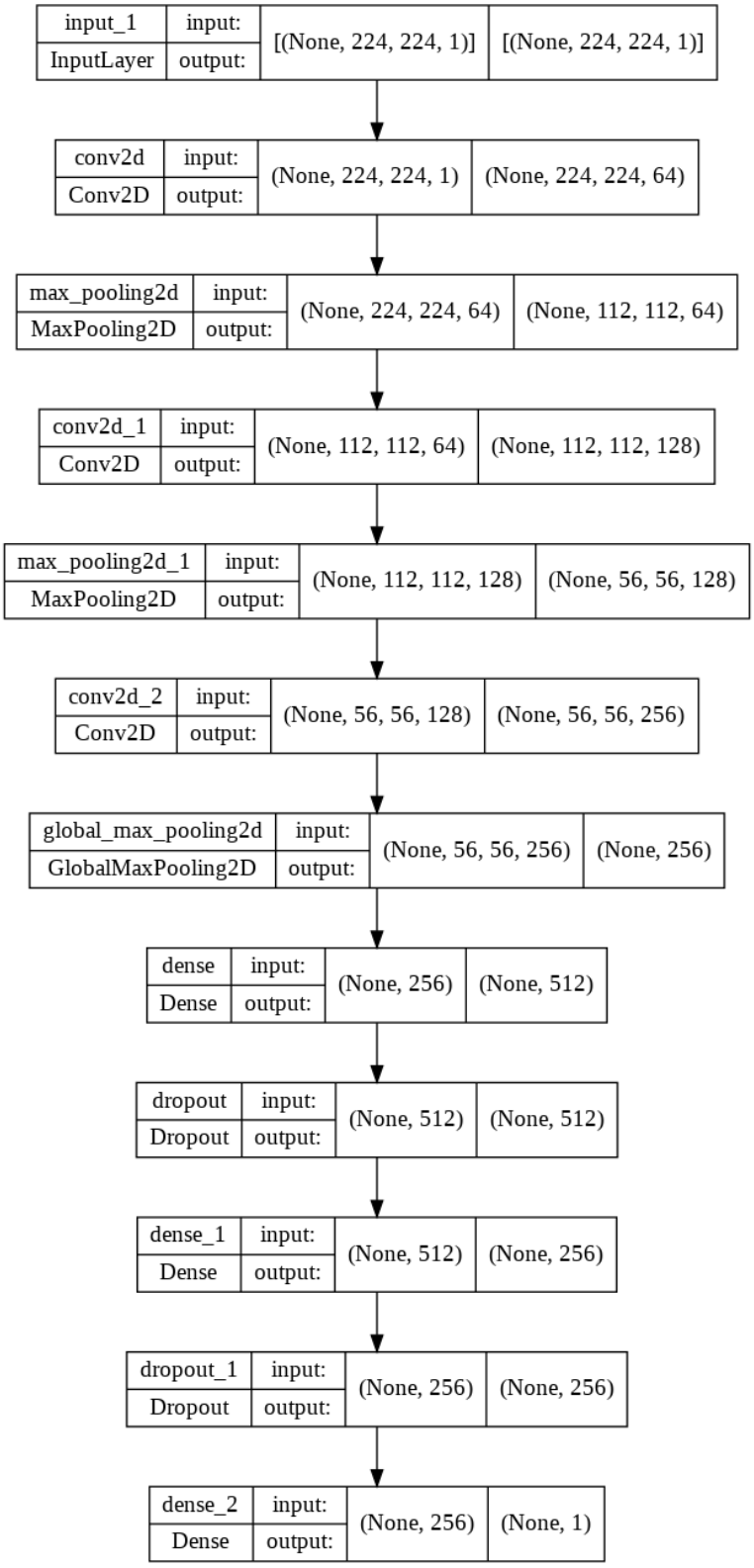


Then I divided the dataset between 760 images for training and 190 for validation.
And I got these results from training the base model without the transfer learning

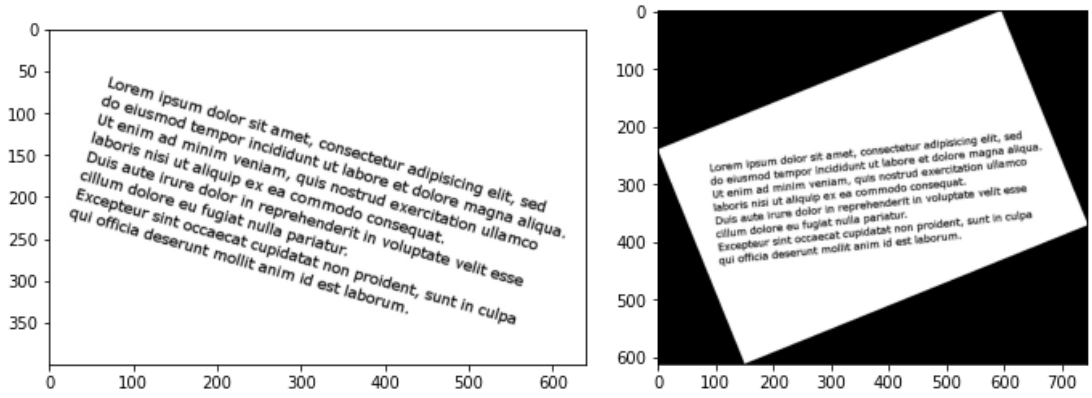


Experiments

From an overview I am using CNN and building a single regression task. Ok how I did that Firstly, I read the images in greyscale and resize them by 224, then transform the angle from Degrees to Radian and apply StandardScaler to normalize the angles.
After that I started building an image generator with some data augmentations like rescaling the images, applying zoom range or changing the brightness range to be (0.8 ~ 1.2).
Then I built the model as following:


input_1 | InputLayer | input: | output: | [(None, 224, 224, 1)] | [(None, 224, 224, 1)]
conv2d | Conv2D | input: | output: | (None, 224, 224, 1) | (None, 224, 224, 64)
max_pooling2d | MaxPooling2D | input: | output: | (None, 224, 224, 64) | (None, 112, 112, 64)
conv2d_1 | Conv2D | input: | output: | (None, 112, 112, 64) | (None, 112, 112, 128)
max_pooling2d_1 | MaxPooling2D | input: | output: | (None, 112, 112, 128) | (None, 56, 56, 128)
conv2d_2 | Conv2D | input: | output: | (None, 56, 56, 128) | (None, 56, 56, 256)
global_max_pooling2d | GlobalMaxPooling2D | input: | output: | (None, 56, 56, 256) | (None, 256)
dense | Dense | input: | output: | (None, 256) | (None, 512)
dropout | Dropout | input: | output: | (None, 512) | (None, 512)
dense_1 | Dense | input: | output: | (None, 512) | (None, 256)
dropout_1 | Dropout | input: | output: | (None, 256) | (None, 256)
dense_2 | Dense | input: | output: | (None, 256) | (None, 1)

I trained the model for 50 epochs with a batch size of 32. And the loss function for the CNN model was Mean Absolute Error (MAE)
The results from the model using a test image is:



I also tried to do **Transfer learning** with a pretrained weight of mobilenet version 3 trained on imagenet, the images were in RGB and we are working on grayscale images so I had to stack 3 channels of the grayscale image but I wasn't able to get better results.

Overall Conclusion

The algorithm has been tested on the data of input document formats and has been found to detect the page orientation and existing skew successfully.

Tools

- Google Colab Notebook which is a data science platform allows you to combine executable code and rich text in a single document along with images
- Google Drive to store data online and read it in Colab
- Python version 2.7.12

External Resources

https://medium.com/mlearning-ai/skew-correction-in-documents-using-deep-learning-8e19609107b6
https://www.researchgate.net/publication/279450553_A_Review_of_Skew_Detection_Techniques_for_Document
https://www.sciencedirect.com/science/article/pii/0923596594900094
https://www.mdpi.com/2079-9292/9/1/55/htm