**Training Tesseract on Custom Font**

For Ubuntu:

1. Installing Tesseract

sudo apt-get update

sudo apt install tesseract-ocr

1. Gathering the images

Collect all the images and place them in the /data folder.

1. Renaming the images

Run the following python script in the same directory and change the language and font according to the dataset.

import os

images = [f for f in os.listdir()[:2] if f.endswith(('.jpg', '.jpeg', '.png', '.tif', '.bmp'))]

print(f"{len(images)} number of images found")

lang = “eng” # Change it according to the dataset

font = “dotted” # Change it according to the dataset

part1 = f"{lang}.{font}.exp"

for i, image in enumerate(images):

filename = f"{part1}{i}.{image[-3:]}"

print(filename)

os.rename(os.path(image), os.path(filename))

1. Generating Box Files

Once we gather the images, we can make tesseract generate box files for the text in the images. Tesseract will try its best to draw boxes around the characters and to identify them. This obviously won't be 100% accurate. This will produce a list of box files for each image file with the file format.

*[language name].[font name].exp[number].box*

These box files will contain boxes around the characters in the image and what each character is. Run the script below to generate the boxfiles.

import os

lang = ”eng” # Change according to the dataset

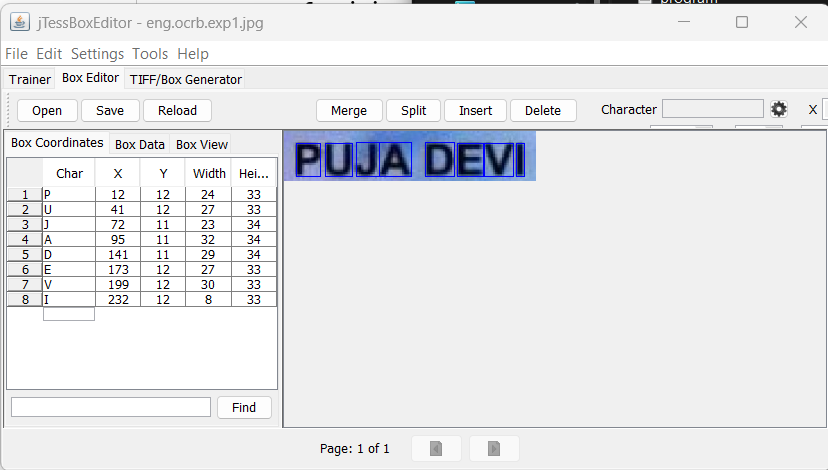
font = ”dotted” # Change according to the dataset

number\_of\_files = len(os.listdir())

for i in range(0, number\_of\_files):

os.system(f"tesseract {lang}.{font}.exp{i}.jpg {lang}.{font}.exp{i} batch.nochop makebox")

1. Annotating the boxes

We can use a program like jTessBoxEditor to open each image file and check the bounding boxes that tesseract has made. We can correct the boxes and the character in them. We can also add new boxes around characters that tesseract has missed.

1. Training Tesseract on the images

Run the script given below in the same directory, it will return a font.traineddata file.

import os

import subprocess

font\_properties="font\_properties.txt"

lang="eng"

font="ocrb"

# Creating font\_properties.txt

fontPropFile = open(font\_properties, "w")

fontPropFile.write("font 0 0 0 0 0")

fontPropFile.close()

files = os.listdir()

jpgs = [x for x in files if x.endswith('.jpg')]

boxes = [x for x in files if x.endswith('.box')]

trainfiles = list(zip(jpgs, boxes))

# Generating .tr files

for image,box in trainfiles:

os.system(f"tesseract {image} {image[:-4]} nobatch box.train")

print("\nGenerated .tr files\n")

# Generating unicharset file

unichar="unicharset\_extractor "

for image,box in trainfiles:

unichar=unichar+box+" "

subprocess.run(unichar,shell=True)

print("\nGenerated unicharset files\n")

# Generating shapetable,pffmtable,inttemp,normproto file

trFiles = [x for x in os.listdir() if (x.endswith('.tr'))]

print(trFiles)

shapetable\_command = f"shapeclustering -F {font\_properties} -U unicharset -O {font}.unicharset "

pffmtable\_command = f"mftraining -F {font\_properties} -U unicharset -O {font}.unicharset "

np\_command = f"cntraining "

for filePath in trFiles:

shapetable\_command=shapetable\_command+filePath+" "

pffmtable\_command = pffmtable\_command+filePath+" "

np\_command = np\_command+filePath+" "

# print(shapetable\_command)

subprocess.run(shapetable\_command,shell=True)

print("\nGenerated shapetable file\n")

# print(pffmtable\_command)

subprocess.run(pffmtable\_command,shell=True)

print("\nGenerated pffmtable,inttemp file\n")

# print(np\_command)

subprocess.run(np\_command,shell=True)

print("\nGenerated normproto file\n")

# Renaming the generated files

os.rename("inttemp",f"{font}.inttemp")

os.rename("normproto", f"{font}.normproto")

os.rename("pffmtable", f"{font}.pffmtable")

os.rename("shapetable",f"{font}.shapetable")

subprocess.run(f"combine\_tessdata {font}.",shell=True)

print(f"\nGenerated {font}.traineddata file")

1. Testing the model

Copy the created font.traineddata file to /usr/shared/tesseract/4.0/tessdata. Now, run the below script to test the model.

try:

import Image

except ImportError:

from PIL import Image

import pytesseract as tes

font="ocrb" # Change according to the dataset

print(tes.image\_to\_string(Image.open("eng.ocrb.exp1.jpg"),lang=font))