Research Internship Report

Arghya Mazumdar
NIT ROURKELA

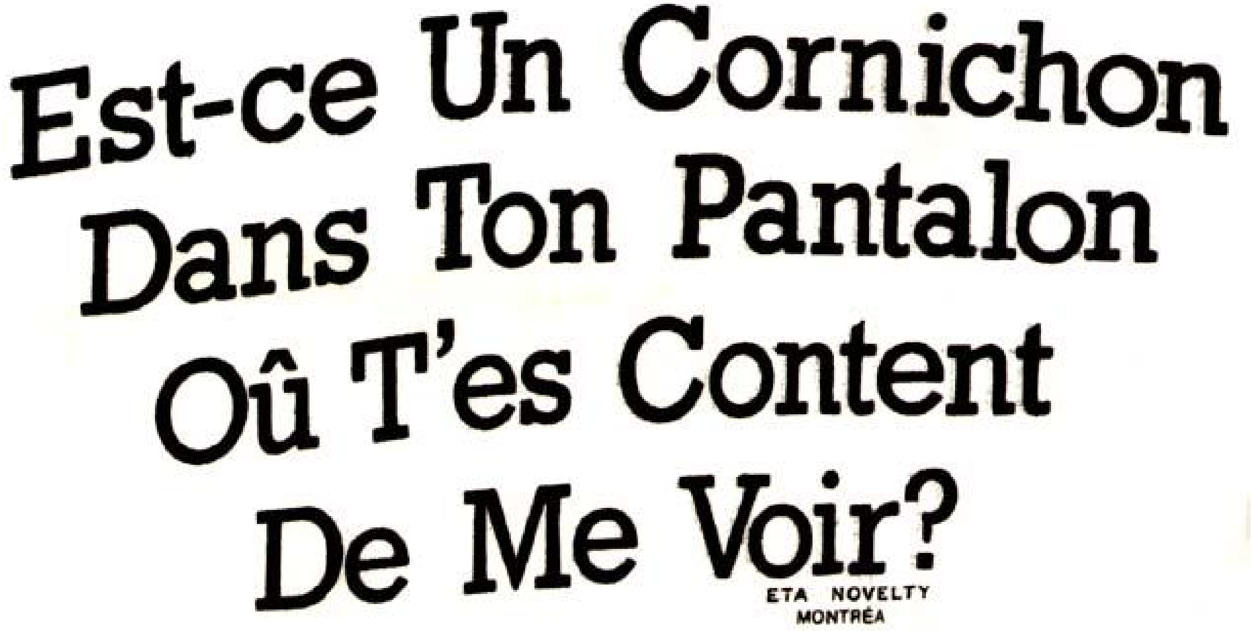
Goal

The primary work of my internship was multi-lingual text classification from scene text images. The major problem tackled was that many languages form a part of the Indian Society and on signboards it becomes very difficult to know which language it actually is. The goal was to pass the image into an OCR reader and then use a language detection algorithm on it which would detect which language the text was written in it.

Task:

The first part of my job was to see the available libraries and packages and test their limitations and performances. I used the Tesseract OCR and the language Detector Class of Java by using the java wrapper of these two packages.

Result:



Est-ce Un Cornichon
Dans Ton Pantalon
Oû T'es Content
De Me Voir?

ETA NOVELTY
MONTRÉA

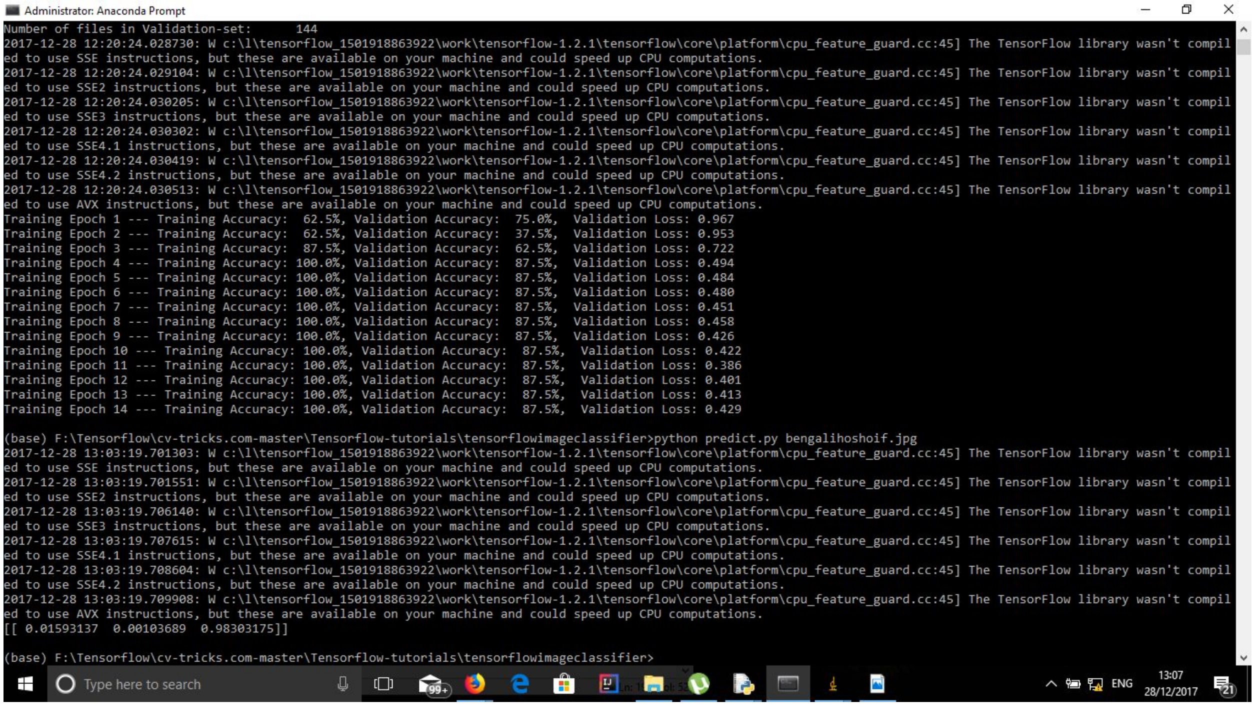
```
Image has been read
Time taken(s) 1.854
Est-Ce Un Cornichon
Dans Ton Pantalon
o? T’es Content
De Me V011”

Language Detected=French
```

The primary limitation of the tesseract ocr was that it was unable to process complex images where text localization had to be done. Also the package was not working well with Indian languages such as bengali and hindi.

Task:

I was asked to come out with a classifier of my own for detecting bengali language by which I can tell which letter of the bengali alphabet it is when I get a character segmented data. I implemented it using convolutional nets using the CMATR.db.3.1.2 and got a validation accuracy of over 80%. I could not optimize the model for further improvements due to hardware issues.The MLP approach discussed which included extracting features from the text could not be completed as the features were not yet ready.This is an aspect I would work on the future.

Result:



```
Number of files in Validation-set:      144
2017-12-28 12:20:24.028730: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 12:20:24.029104: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE2 instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 12:20:24.030205: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE3 instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 12:20:24.030302: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 12:20:24.030419: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 12:20:24.030513: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use AVX instructions, but these are available on your machine and could speed up CPU computations.
Training Epoch 1 --- Training Accuracy:  62.5%, Validation Accuracy:  75.0%,  Validation Loss: 0.967
Training Epoch 2 --- Training Accuracy:  62.5%, Validation Accuracy:  37.5%,  Validation Loss: 0.953
Training Epoch 3 --- Training Accuracy:  87.5%, Validation Accuracy:  62.5%,  Validation Loss: 0.722
Training Epoch 4 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.494
Training Epoch 5 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.484
Training Epoch 6 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.480
Training Epoch 7 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.451
Training Epoch 8 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.458
Training Epoch 9 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.426
Training Epoch 10 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.422
Training Epoch 11 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.386
Training Epoch 12 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.401
Training Epoch 13 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.413
Training Epoch 14 --- Training Accuracy: 100.0%, Validation Accuracy:  87.5%,  Validation Loss: 0.429

(base) F:\Tensorflow\cv-tricks.com-master\Tensorflow-tutorials\tensorflowimageclassifier>python predict.py bengalihoshoif.jpg
2017-12-28 13:03:19.701303: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 13:03:19.701551: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE2 instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 13:03:19.706140: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE3 instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 13:03:19.707615: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 13:03:19.708604: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations.
2017-12-28 13:03:19.709908: W c:\l\tensorflow_1501918863922\work\tensorflow-1.2.1\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compil
ed to use AVX instructions, but these are available on your machine and could speed up CPU computations.
[[ 0.01593137  0.00103689  0.98303175]]

(base) F:\Tensorflow\cv-tricks.com-master\Tensorflow-tutorials\tensorflowimageclassifier>
```

This is a slice of the dataset i trained on using a convolution net structure of three layers and one fully connected layer. The dimensionality of the two layers were 3*3 and the other 5*5 for greater accuracy. The optimization algorithm used was Adam and learning rate of 0.05 and trained for 1000 iterations. The three letters I tested were the first three letters oo,aa and hoshoi. Then I gave the classifier an image of hoshoi which showed me a 0.99 probability of that particular class. The overall classifier showed a training accuracy of 100%,validation accuracy of 87.5% and validation loss of 0.420. The entire dataset model needs a gpu to train on which was beyond my cpu's capability.
Further improvements can be made by adding more layers or by using a convolutional net with inception so that we get further accuracy due to the combined effect of the convo nets.

ACKNOWLEDGEMENT:

I thank Prof Ram Sarkar for the support and guidance given by him to carry out my work. I also like to thank Neelotpal Chakroborty , the PHD scholar under whom I worked. I hope my work can contribute to their research purposes.