DSP-Project

Machine Learning for Classification of Receipts

Srikrishna KrishnaRao
Indra Reddy Gayam
Jashwanth Gottipati

Introduction

Receipt(the hard copies) is a very regular item that many customers receive and contain huge amount of information. But the most difficult challenge is identification of data which is unstructured. This solution deals with classification of receipts based on an item purchased by the user- in this scenario receipt images are classified based on Milk buying or Milk Non-buying groups. Classifying the receipts could be very challenging due to fuzzy visibility when compared with the regular documents and a wide variety of fonts.

This model could be very useful for the reimbursement companies to validate and provide reimbursement for particular products identified in the receipts. For instance this particular model could be used for the USDA Milk Donation Reimbursement Program(MDRP). And intuition can be applied in many similar circumstances.

This paper narrates the data pipeline for identifying and separating the receipt in the scanned image from the background, cropping the image and extracting the core region of the image to identify if milk is present in the receipt and augmenting the images for generating data required for model. Finally Deep learning models are applied to classify and predict the presence of the product in the image.

We have attempted to implement the concept of Input image data > Output classes (instead of Input data > Extract required text > Output classes) in this project (reference Andrew Ng's video on Deep learning)

Related work

Academic:

- This research paper by Dr.Alex Yue from Princeton University details out the process of reading receipts using Machine learning
  - **https://www.cs.princeton.edu/courses/archive/spring18/cos598B/public/projects/System/COS598B_spr2018_ReceiptParsing.pdf**
- This research paper focused on Convolutional Neural Network for Image classification at scale
  - **http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf**
- Even though the following is not receipt related, it is related to invoice and some of the concepts could be used
  - **http://cs229.stanford.edu/proj2016/report/LiuWanZhang-UnstructuredDocumentRecognitionOnBusinessInvoice-report.pdf**

- Focus Enhanced Scene Text Recognition with Deformable Convolutional Neural Network
    - https://arxiv.org/pdf/1908.10998v2.pdf
- Detecting text in images using convolutional networks
    - http://cs231n.stanford.edu/reports/2015/pdfs/vikesh_final.pdf
- Scene text detection and recognition: recent advances and future trends
    - https://web.archive.org/web/20170321094647/http://mc.eistar.net/uploadfiles/Papers/FCS_TextSurvey_2015.pdf
-

Industry

- Reading Invoices from OCR
    - https://nanonets.com/blog/invoice-ocr/

- Comcast: https://databricks.com/session/how-to-utilize-mlflow-and-kubernetes-to-build-an-enterprise-ml-platform

Methodology

This project is comprised of methods and systems to automate image processing and feed to the Deep learning models. The purpose of doing it is to identify if a receipt contains a particular item in the list of items bought .

The primary steps involved are:

1. Preprocessing
2. Data Augmentation
3. Model Building

Preprocessing:

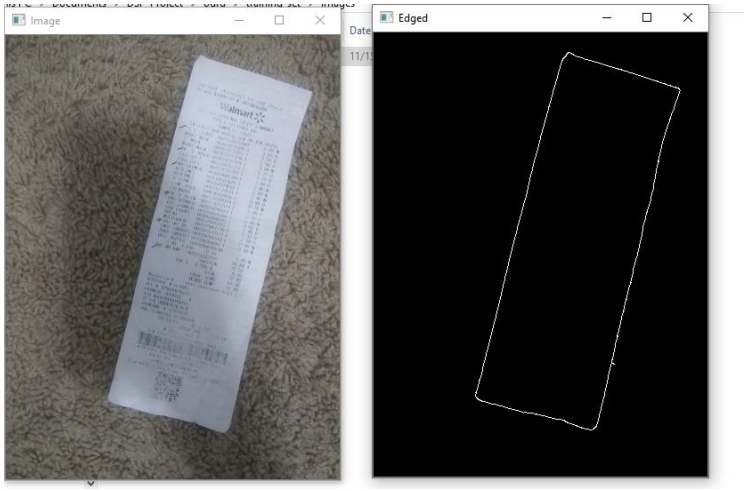
Pre-processing of data is tremendously important, especially in the case of receipt related image classification models.

Edge Detection: Generally, the receipts are scanned with diverse background. Edge detection involves determining the foreground versus background for any given image. It assumes

1. Receipt is clearly distinguishable from the background.
2. Scanned receipt is Rectangular

Steps Involved:

- To make edge detection more accurate and quicker the receipt images are resized to smaller dimensions by keeping track of the ratio resized
- The colored image is converted into grayscale and blurring is performed using gaussian filter to remove high frequency noise components reduction.
- Canny Edge detector from OpenCV library is used to detect the edges of the receipt



Original Image and the final image with detected edges

Finding Contours:

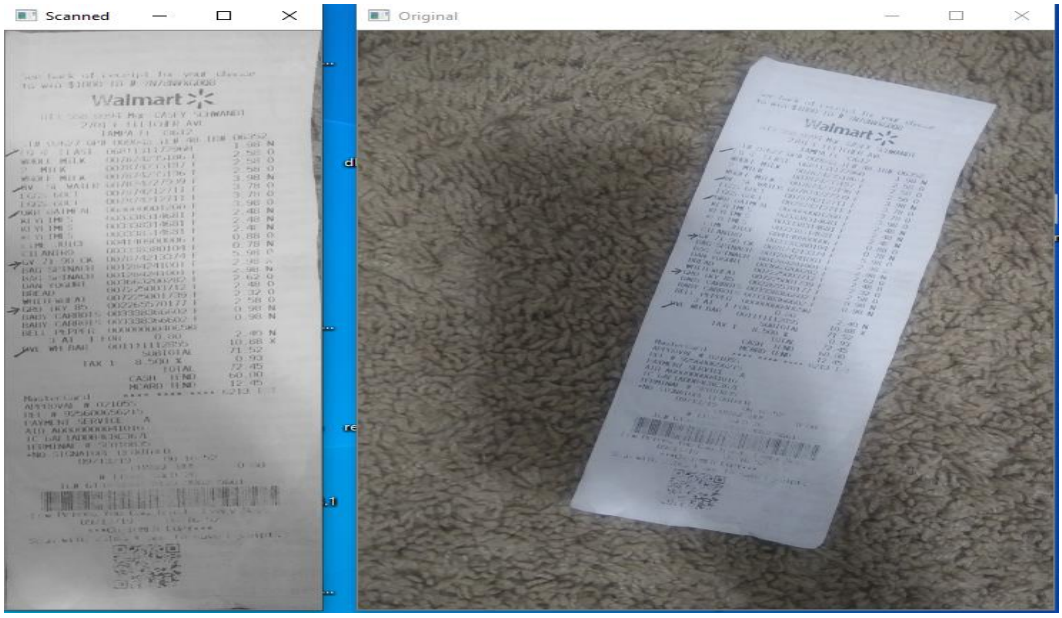
Contours are found by utilizing the edges. For detecting Contours the system assumes the following:

1. Each receipt contains Four edges (Xmin, Xmax,Ymin,Ymax)
2. We assume largest contour in the image is rectangular and with exactly four points

Steps Involved:

- All the contours are found and only the contour with maximum area is kept by looping over them

Cropped Image:



Cropped and original image

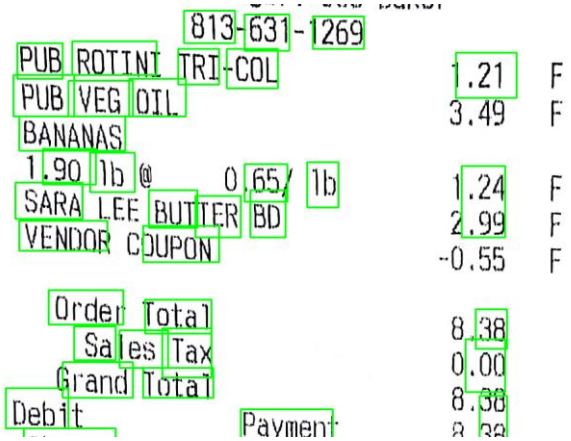
For getting the bird eye view of the image, cv2 getPerspectiveTransform is used for this purpose.Four point transform function takes in the arguments as Image and the contours identified.

Utilized four_point_transform from https://www.pyimagesearch.com/

Data Generation via Augmentation:

We were able to collect 60 Publix receipts out of which 30 were readable and distinct. For overcoming the problem of sparse dataset, data augmentation is leveraged. The receipt is cropped so that it includes the core part of the image which contains the product description. Using image processing library skimage random noise is included to generate 1000 images. Also data augmentation is performed using Keras Datagenerator method which generates noise and created additional images that are stored in the memory so that model can be trained with sufficiently high amount of data.

Bounding Boxes and ROI's:



For detecting Bounding boxes East- (Efficient and accurate scene detection) deep learning model is used, output feature maps are extracted from two layers for probability

and for finding bounding box coordinates. Non max suppression is applied to bounding boxes to suppress weak overlapping boxes

Model:

We used Keras library to build Convolution Neural Network, to classify the images. Data preprocessing is done on more time using Keras ImageDataGenerator.

The following folder structure is maintained for the model to train, test and evaluate images.

- Dataset
    - Test_set
        - Milk
        - Non-Milk
    - training_set
        - Milk
        - Non-Milk

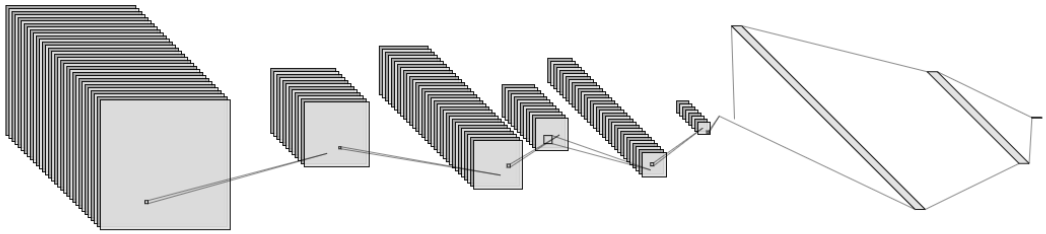
Model fit is performed by selecting number of epochs as 5 and steps_per_epoch as 40 and batch size as 20.

For overcoming the overfitting issue, early stopping is added with patience value as 2.

So if the model accuracy score goes down for a continuous range of 2 epochs

Following structure is used for building CNN

Architecture:



- Three convolution layers and 3 max pooling layers are used
- Convolution layer1 accepts the input format as 128*128 pixels and as RGB format- with number of layers as 3 and filter size of 3*3. It creates 32 feature maps and relu is used as non-linear activation function to bring in the nonlinearity in the feature maps.
- MaxPooling Layer1 uses pool_size of (2*2) for downsampling the feature maps by summarizing the most activated presence of a feature respectively.
- Similarly 2 more sets of Convolution and Pooling Layers are added
- Finally, Dense layers are added where linear operation is performed on 128 input nodes and the activation function ‘relu’ is used.

- For the final layer, sigmoid activation function is used to provide binary output (Either 0 or 1)
- After comparing various optimizers like RMSprop,sgd ,Adam and Adaboost , Adam optimizer is used and learning rate is set to 0.001

Model Evaluation:

Model is evaluated using the Accuracy metric that is shown in the running of Epochs. They contain validation accuracy and training accuracy.

In the current scenario, validation and training accuracy were very close to 1, meaning over fit. Hence the issue of overfit was resolved by adding more data, generated artifically, and by adding Dropouts at a probability of 0.4

Results

Iteration 1 had overfitting problem with training and validation accuracy of 1.0 and prediction accuracy of 50%

By adding a Dropout with probability of 40%, the training and validation accuracy of 83% was received as follows

Epoch 1/5
40/40 [==============================] - 151s 4s/step - loss: 0.7826 - accuracy: **0.4914** - val_loss: 0.6936 - val_accuracy: **0.4850**

Epoch 2/5
40/40 [==============================] - 68s 2s/step - loss: 0.6724 - accuracy: **0.5603** - val_loss: 0.6888 - val_accuracy: **0.6600**

Epoch 3/5
40/40 [==============================] - 63s 2s/step - loss: 0.6149 - accuracy: **0.6584** - val_loss: 0.6237 - val_accuracy: **0.7100**

Epoch 4/5
40/40 [==============================] - 64s 2s/step - loss: 0.4233 - accuracy: **0.8155** - val_loss: 0.5009 - val_accuracy: **0.7450**

Epoch 5/5
40/40 [==============================] - 64s 2s/step - loss: 0.2544 - accuracy: **0.9010** - val_loss: 0.2801 - val_accuracy: **0.8350**

Future work:

Model would work accurately if the Bounding Box coordinates are passed to the CNN model and ROI pooling can be performed instead of MaxPooling. Passing the coordinates for ROI pooling was a challenge while working with the project.

Additionally, along with classification of receipts ,effort was made to extract the data from the images using Recurrent Neural Network which is present in the code section.

Following are some of the other improvements which we plan to consider for future:

Applications like Facebook, Twitter and Instagram are so popular for a reason that, every individual is so curious to know how other thinks about them. In the same way, everyone is curious on how others spend their money, the products they buy and the place where they buy them, brands they buy, how often they buy and the savings they do.

Implications:

Imagine when you place your mobile device on a receipt and it identifies all the items on it and provides you a notification that could list the places where you could buy the same items at a cheaper price. This could be achieved through decentralized network, where everyone scans there receipts on their mobile devices and all the information from the receipt that includes the organisation name, list of items they have bought and their respective prices getting stored in a decentralized database. Now the database have the details of any item and the organization where that could be bought at a cheaper price.

Sharing the receipt details with all the contacts in your mobile can provide you a dashboard on how good or bad are you at your expenses every month. Imagine your application show you a notification that says "You have saved 10$ on dairy products compared to Mrs. XXX this month". This creates a social network of all the expenses a family is making and everyone in the family can track the expenses of other persons.

Videos tutorials for reading receipt
https://www.youtube.com/watch?v=B1d9dpqBDVA
https://www.youtube.com/watch?v=gViXzJl5ef4
https://www.youtube.com/watch?v=w95ktvMyZxg

Receipt samples and code examples
https://gist.github.com/fgrehm/d3612ee6a84fc74e4595e52078040d46
https://dzone.com/articles/using-ocr-for-receipt-recognition
PySpark tutorial to run regression
https://www.youtube.com/watch?v=C49R0o5gE9c&feature=youtu.be

Task	Status
Collect Clean Receipt samples - for CNN large volume is must Receipt images collected personally https://drive.google.com/drive/folders/1dFvgxje_YhGVS79mHf_D9maABR2kQSVc https://github.com/indravz/DSP-Project Following public datasets could not be extracted http://www.cs.cmu.edu/~aharley/rvl-cdip/ https://rrc.cvc.uab.es/?ch=13&com=downloads Challenge SROIE	

To mention in the report: challenge of small number of receipts and how we overcame **1. Data augmentation**	
Collect noise samples Crumpled image samples Blurred text samples	
Load images to S3 or Databricks MLflow	
Call Amazon API and extract Text to .csv file or Databricks API	
A). Compare Sample images and match Text of Store Name and record accuracy by manual labeling - AWS extracted text	
Load .csv to PySpark and Run K-Means Clustering	
Train a Machine Learning algorithm on PySpark to extract text from image, identify Store name	
B). Compare Sample images and match Text of Store name and record accuracy by manual labeling - manually trained ML	
Supportive tasks Identify machine learning algorithms to manually train receipt image text extraction	
Model deployment https://databricks.com/session/how-to-utilize-mlflow-and-kubernetes-to-build-an-enterprise-ml-platform https://databricks.com/blog/2019/10/17/managed-mlflow-now-available-on-databricks-community-edition.html https://pytorch.org/	

References

Resources	Comments
Datasets and APIs http://www.iapr-tc11.org/mediawiki/index.php/Datasets_List	
https://medium.com/capital-one-tech/learning-to-read-computer-vision-	

methods-for-extracting-text-from-images-2ffcdae11594	
https://towardsdatascience.com/all-the-steps-to-build-your-first-image-classifier-with-code-cf244b015799	Step by step image classification
https://stackoverflow.com/questions/31633403/tesseract-receipt-scanning-advice-needed https://stackoverflow.com/questions/45796771/how-to-extract-relevant-information-from-receipt https://stackoverflow.com/questions/44112843/ocr-for-detecting-bills https://www.youtube.com/watch?v=B1d9dpqBDVA **For Preprocessing of image -** https://likegeeks.com/python-image-processing/ https://docparser.com/blog/improve-ocr-accuracy/ Building OCR: https://www.quora.com/What-are-the-steps-for-making-an-optical-character-recognition-OCR-from-scratch https://blogs.dropbox.com/tech/2017/04/creating-a-modern-ocr-pipeline-using-computer-vision-and-deep-learning/ https://nanonets.com/blog/deep-learning-ocr/ https://hackernoon.com/latest-deep-learning-ocr-with-keras-and-supervisely-in-15-minutes-34aecd630ed8	
Convolutional Neural Network https://towardsdatascience.com/from-raw-images-to-real-time-predictions-with-deep-learning-ddbbda1be0e4	
Google Collab to train CNN https://towardsdatascience.com/tensorflow-2-0-create-and-train-a-	

vanilla-cnn-on-google-colab-c7a0ac86d61b	
https://www.analyticsvidhya.com/blog/2018/04/solving-an-image-captioning-task-using-deep-learning/	
https://hackernoon.com/latest-deep-learning-ocr-with-keras-and-supervisely-in-15-minutes-34aecd630ed8	
Image generation/augmentation https://medium.com/@thimblot/data-augmentation-boost-your-image-dataset-with-few-lines-of-python-155c2dc1baec Github code for data augmentation https://gist.github.com/tomahim/9ef72befd43f5c106e592425453cb6ae https://gist.github.com/tomahim/9ef72befd43f5c106e592425453cb6ae	
Paper and code for Scene text recognition https://paperswithcode.com/task/scene-text-recognition https://github.com/Alpaca07/dtr/blob/master/train.py https://arxiv.org/pdf/1908.10998v2.pdf	
Curated list of papers related to scene text recognition https://web.archive.org/web/20170207033948/https://github.com/chongyangtao/Awesome-Scene-Text-Recognition	