Electronic Invoicing using Image Processing

Team Name   : Good Team

Institute Name: IISc, Bangalore

Team members details

Team Name	Good Team		
Institute Name	Indian Institute of Science, Bangalore		
Team Members >	1 (Leader)	2	3
Name	Jayant Priyadarshi	Deepesh Hada	Danish Shaikh
Batch	2019 - 21	2019 - 21	2019 - 21

Functionalities of Product

- **USP:** Buyers and sellers, who still rely on manual reconciliation of invoices and bills, make up the target audience. The product aims to provide effective reconciliation between the duo by removing manual intervention completely (just an image upload would do the task!), thereby ensuring speed along with reliability. This is fruitful, especially during this pandemic, where there must be minimal contact and when one cannot muster enough human resources.
  What sets the solution apart is that it is based on some of the state-of-the-art segmentation techniques and doesn't use any paid proprietary software!
  The product can hence be used as an alternative to relieve the error-prone manual efforts with a few clicks, making it beneficial for Flipkart.

- **Technology Used:** We're using instance segmentation techniques to solve table detection, table structure recognition (for borderless tables) and text region identification problems in the input documents. Once the key text regions are segmented, we use an image-to-text OCR-based algorithm to convert these regions to text to gain information from it. We'll **NOT** be using any paid licensed products to achieve this; however, we'll be using Tesseract OCR, a free open source OCR tool for image-to-text utilities.

- **Scaling the product:** The solution is highly scalable and can be generalized to new templates as it achieved very accurate results (and has been trained) on the diverse TableBank, ICDAR-2013, ICDAR-2019 and some custom-made datasets.

Product Specifications

Technical Specifications:

- Python v3.7
- PyTorch v1.4
- mmdetection v1.2
- OpenCV v4.3.0

Physical Specifications:

- Google Colaboratory platform (with runtime set to GPU)
- Sufficiently high RAM (8-16 GB)
- GPU with around 8-16 GB GPU memory (for mmdetection toolbox)

Product Limitations

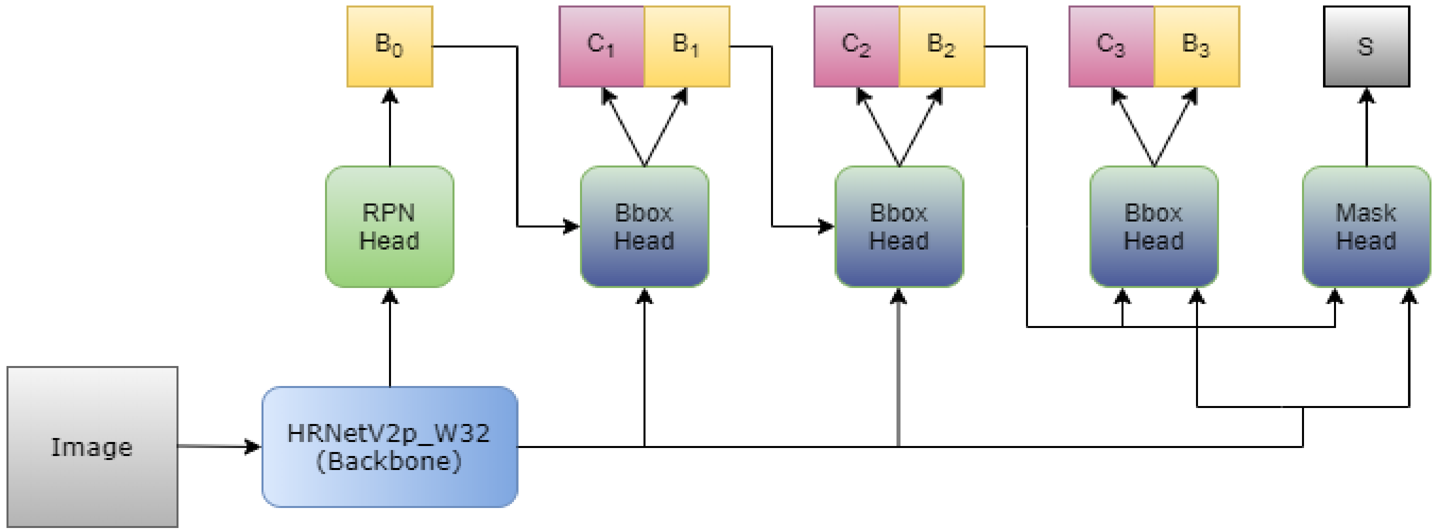
- The product may fail when the invoice has **NO** tabular structure, although, even if there are borderless table structures in the invoice, they’ll get detected.

- When the table gets spanned across multiple pages, then our model could not detect it as an overall single table, it might get split into two tables.

- As we are using OCR to convert contents of cells of the table to raw text, the output of OCR might be noisy as it is highly dependent on the font family, font size as well as on the quality of the image/pdf that we are giving as an input.

- Model fails to detect the information hidden in the acronyms and abbreviations.

Architecture



- Invoices usually have structured information present in a tabular format. Hence, we have used table detection in the input followed by extracting text through an OCR.
- We have followed these papers: CascadeTabNet, Cascade Mask R-CNN and HRNet.
- The invoice is sent as an input image to the backbone CNN.

Architecture

- In our case, the backbone is High-Resolution Network's v2 for semantic segmentation (HRNet).
- HRNet is a very recent technique to obtain high resolution features using interconnected parallel convolutions, which is best suited for tasks which demand good localization, making it the best backbone choice for our use-case.
- On the features produced by HRNetv2, we use a Cascade Mask R-CNN for instance segmentation, which helps to recognize the various instances of tables and the corresponding cells in the invoice.
- This feature map is scanned by a Region Proposal Network (RPN) to identify the regions of interest (region proposals).
- These proposals are then sent to RoI (Region of Interest) Align layer followed by 3-step cascade classification and bounding-box regression layers. The classifiers are trained to identify the tables and the regression is performed to refine the boxes that contain tables.
- In the last step of Cascade R-CNN, an instance segmentation mask is predicted which helps to differentiate between different tables in the invoice at pixel-level.

Architecture

- In the second iteration, the model is again fine-tuned on a smaller dataset to accomplish a more specific task of classifying tables as **bordered** or **borderless** and predicting the cell masks in **borderless** tables.

- Cell masks in bordered tables are predicted using simple edge-detection techniques.

- From these segmented cell masks, we detect and recognize the respective text through an existing OCR (Optical Character Recognition) SaaS software.

- We plan to use **Tesseract** OCR, which is a deep learning-based OCR and is significantly more accurate. Tesseract is freely available and has a seamless integration with Python through the **pytesseract** package.

Brief on Programming Module

Programming Language: Python 3.7 with PyTorch, mmdetection and OpenCV.

Software Modules:

1) mmdetection is an open source object detection toolbox based on PyTorch which contains HRNetV2 and Cascade Mask R-CNN models trained on ImageNet.

2) We will use HRNetV2 and Cascade Mask R-CNN models from this toolbox and then fine-tune them over Table Detection datasets that we have considered (mentioned in an earlier slide).

3) We will be using Tesseract OCR, free software released under Apache license, to recognize characters of the contents of the tabular cell.

4) After the text is recognized through OCR, we’ll then be using RegEx/Levenshtein distance-based matching of column names of the extracted text and relevant terms required in the output file to store the retrieved information at appropriate locations in the output file.

Brief on Programming Module

Integration with existing SaaS softwares:

- The only SaaS software that we’re using is the Tesseract OCR engine.

- Each of the segmented cells from the output of the detection model will be cropped (through OpenCV) and passed on to the OCR engine through the **pytesseract** Python package.

Execution Plan

Dataset Preparation and Augmentation:

- We plan to combine data from multiple publicly available sources, namely, ICDAR-13, ICDAR-19, Marmot and a custom dataset, as indicated in the CascadeTabNet paper. This combined dataset will then be used as the training set for table detection and instance segmentation.
- Also, as CascadeTabNet suggests, we'll further augment more images to the training set using the Smudge and Dilation Transformations.

Model Construction and Fine-Tuning:

- The **mmdetection** toolbox contains the Cascade Mask R-CNN and HRNet models, pretrained on ImageNet.
- We plan to combine these models, as explained in the ***Architecture*** Section, and fine-tune them on the augmented dataset.

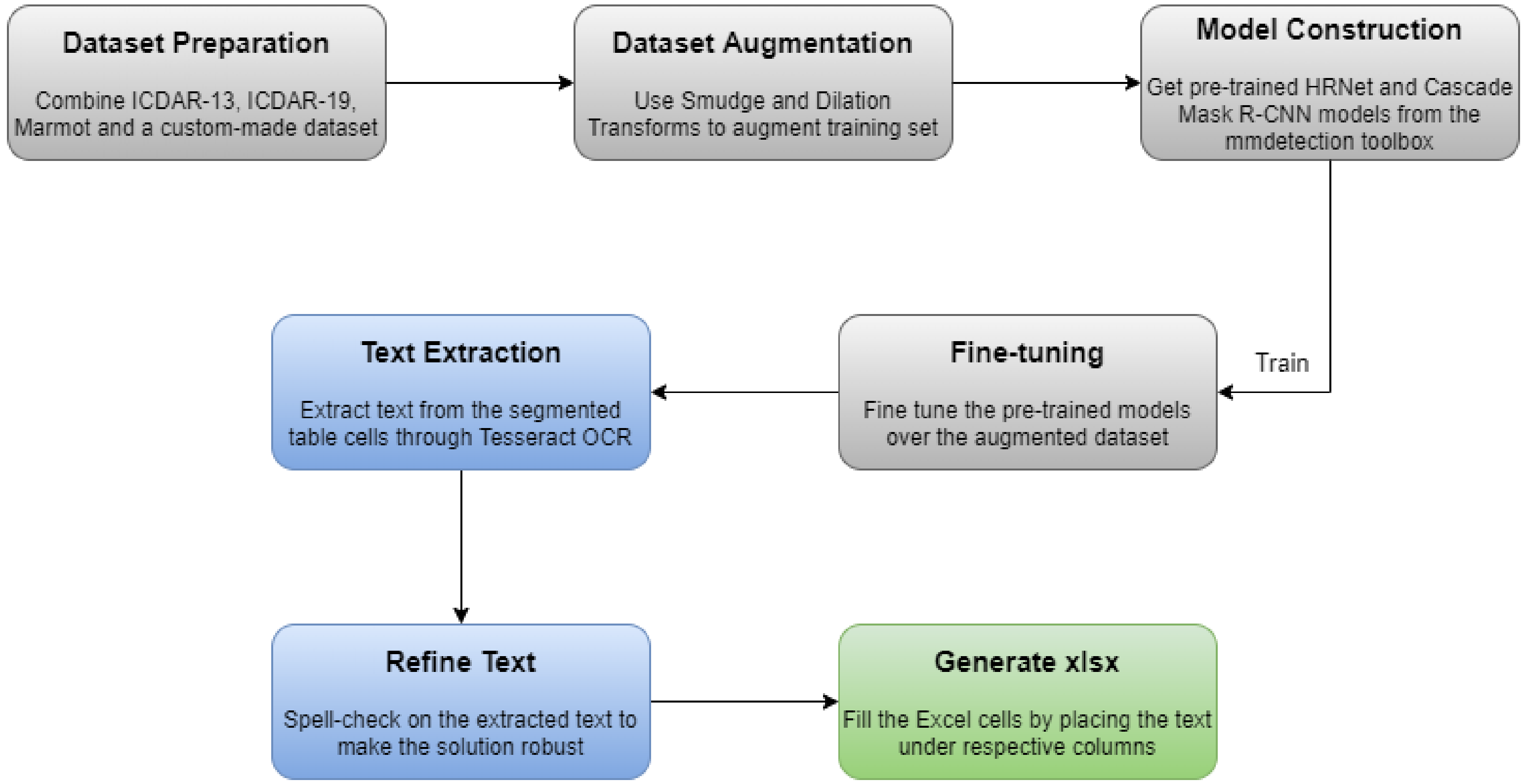
Text Extraction and Refinement:

- We would then extract the text from segmented regions through Tesseract OCR.
- Since the OCR outputs could be noisy, we'll refine them further using algorithms like Minimum Edit Distance.

Generating output file:

- The filtered text will be placed in the respective columns of the output Excel file through some RegEx and Levenshtein distance matching of columns.

Execution Plan



Dataset Preparation
Combine ICDAR-13, ICDAR-19, Marmot and a custom-made dataset
Dataset Augmentation
Use Smudge and Dilation Transforms to augment training set
Model Construction
Get pre-trained HRNet and Cascade Mask R-CNN models from the mmdetection toolbox
Train
Fine-tuning
Fine tune the pre-trained models over the augmented dataset
Text Extraction
Extract text from the segmented table cells through Tesseract OCR
Refine Text
Spell-check on the extracted text to make the solution robust
Generate xlsx
Fill the Excel cells by placing the text under respective columns

Flipkart

GRiD 2.0