

Task 15

Name - Harsh Shah

Referral id - SIRSS1200

Text extraction from given images

Like many companies, not least financial institutions, Capital One has thousands of documents to process, analyze, and transform in order to carry out day-to-day operations. Examples might include receipts, invoices, forms, statements, contracts, and many more pieces of unstructured data, and it's important to be able to quickly understand the information embedded within unstructured data such as these.

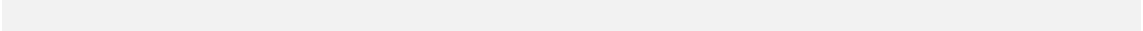
Fortunately, recent advances in computer vision allow us to make great strides in easing the burden of document analysis and understanding. In this post, we'll describe a multi-task convolutional neural network that we developed in order to efficiently and accurately extract text from images of documents.

Optical Character Recognition

The challenge of extracting text from images of documents has traditionally been referred to as [Optical Character Recognition \(OCR\) and has been the focus of much research](#). When documents are clearly laid out and have global structure (for example, a business letter), existing tools for OCR can perform quite well. A popular open source tool for OCR is the Tesseract Project, which was originally developed by Hewlett-Packard but has been under the care and feeding of Google in recent years. Tesseract provides an easy-to-use interface as well as an accompanying Python client library, and tends to be a go-to tool for OCR-related projects. More recently, cloud service providers are rolling out text detection

capabilities alongside their various computer vision offerings. These include [GoogleVision](#), [AWS Textract](#), [Azure OCR](#), and [Dropbox](#), among others. It is an exciting time in the field, as computer vision techniques are becoming widely available to empower many use cases.

There are, however, many use cases in what we might call non-traditional OCR where these existing generic solutions are not quite the right fit. An example might be in detecting arbitrary text from images of natural scenes. Problems of this nature are formalized in the [COCO-Text challenge](#), where the goal is to extract text that might be included in road signs, house numbers, advertisements, and so on. Another area that poses similar challenges is in text extraction from images of complex documents. In contrast to documents with a global layout (such as a letter, a page from a book, a column from a newspaper), many types of documents are relatively unstructured in their layout and have text elements scattered throughout (such as receipts, forms, and invoices). Problems like this have been recently formalized in the [ICDAR DeTEXT Text Extraction From Biomedical Literature Figures challenge](#). These images are characterized by complex arrangements of text bodies scattered throughout a document and surrounded by many “distraction” objects. In these images, a primary challenge lies in properly segmenting objects in an image to identify reasonable text blocks. Example images from COCO-Text and ICDAR-DeTEXT are shown below. These regimes of non-traditional OCR pose unique challenges, including background/object separation, multiple scales of object detection, coloration, text orientation, text length diversity, font diversity, distraction objects, and occlusions.



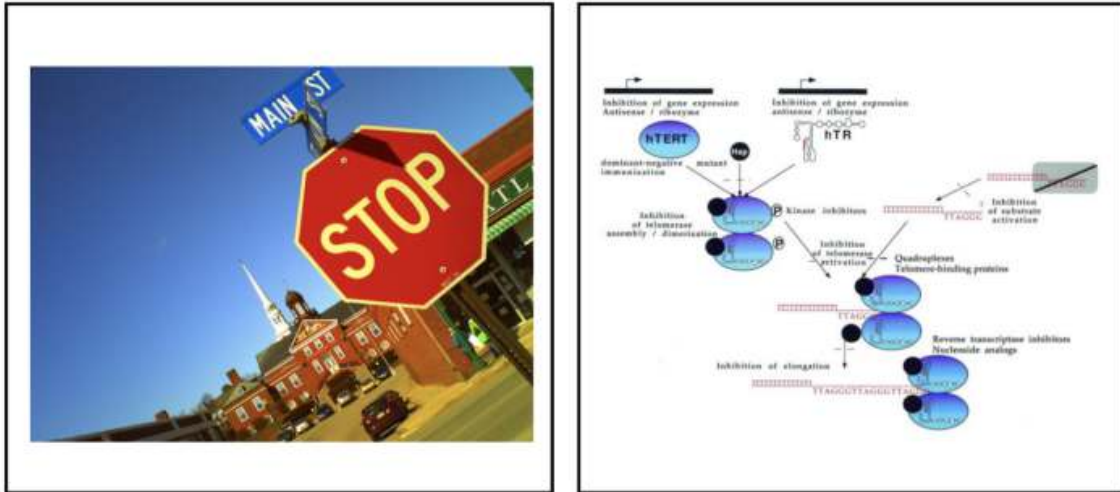


Figure 1. Example images from COCO-Text challenge (left) and ICDAR DeTEXT challenge (right). Notice that OCR in this regime requires the detection of text objects as separate from background pixels and other distractions.

The problems posed in non-traditional OCR can be addressed with recent advances in computer vision, particularly within the field of object detection. As we discuss below, powerful methods from the object detection community can be easily adapted to the special case of OCR.

Object Detection

The field of computer vision aims to extract semantic knowledge from digitized images by tackling challenges such as image classification, object detection, image segmentation, depth estimation, pose estimation, and more. For this discussion, we'll focus on the field of object detection (and related image segmentation) which has seen impressive improvements in recent years. [Early attempts at object detection focused on applying image classification techniques to various pre-identified parts of an image.](#) Many approaches have focused on speeding up the [identification of candidate regions](#) and on using [convolutional mechanisms for feature extraction and classification](#). While there have been many interesting developments in the field, we will focus primarily

on [MaskRCNN](#), a model which is able to very successfully conduct object detection and image segmentation.

An example output from MaskRCNN is shown below. For any input image, this model is trying to accomplish three things: object detection (green boxes), object classification, and segmentation (colorful shaded regions). The green bounding boxes on the image below are the outputs of the model and above each box is a prediction of what kind of object is contained within. We can see that when this scene of a busy street is fed into the model, MaskRCNN is able to successfully identify a large variety of different scene objects including people, cars, and traffic lights. Further, inside each identified bounding box, the colorful shaded region identifies exactly which pixels in an image correspond to the object. This is referred to as segmentation, and each pixel in the image receives a predicted classification label about what kind of object that pixel belongs to (or background).

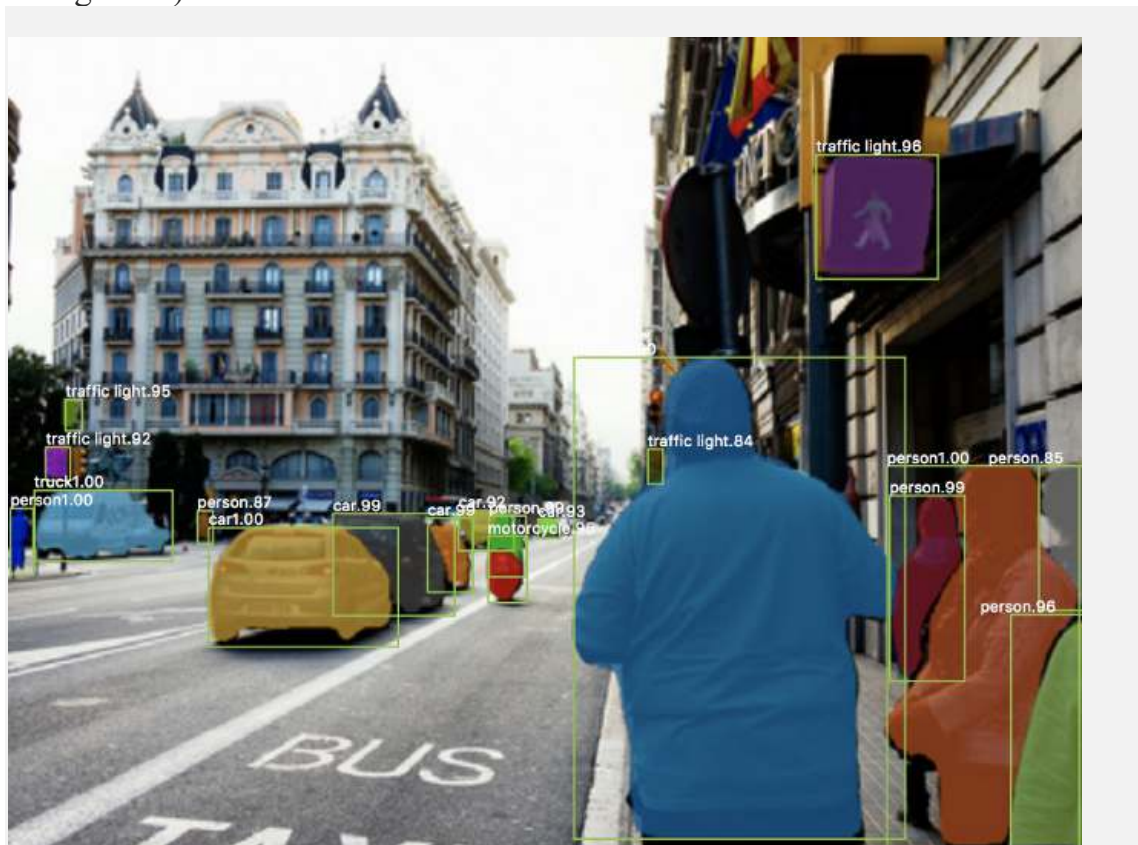


Figure 2: Output of MaskRCNN for an image of a busy street scene. Notice that relevant objects are detected (people, cars, traffic lights) with bounding boxes as well as segmentation regions.

MaskRCNN is an example of a multi-task network: with a single input (image), the model must predict multiple kinds of outputs. Specifically, MaskRCNN is split into three *heads*, where one of the heads is concerned with proposing bounding boxes that likely contain objects of interest, another head is concerned with classifying which type of object is contained within each box, and the final head identifies a pixel-wise bitmask for estimating the segmentation in each box. Importantly, all three of the heads are relying upon a shared representation that is calculated from a deep convolutional backbone model such as [ResNet](#) or similar. This shared representation is important in multi-task learning, and allows each of the heads to back-propagate their respective errors and update this backbone representation. The overall effect of this is that each head actually becomes more accurate than if they were trained as separate models.

Detecting Text Objects

Object detection models such as MaskRCNN and its predecessors provide a very flexible mechanism for identifying regions of interest inside images. As you might guess, we can view non-traditional OCR as closely related to object detection. In this case, we have really only two classes of objects we care about: text objects and then everything else. With this perspective, we can train a model very similar to MaskRCNN to identify regions of interest (RoI) in an image that are highly likely to contain text, a task that is known as *text localization*. An example output of such a model is shown below.

Notice that this input image of a receipt presents some interesting challenges for text extraction. First, the document of interest occurs alongside some background objects (a steering wheel). Second, the text within the document is highly unstructured and therefore it is beneficial to separately identify all the possible text blocks. The output of the model is overlaid on the image above — regions of text are identified with dotted-line bounding boxes and we even have estimated the pixel mask for the text. Above each box is the predicted class and confidence score, which, since we only have one object class of interest, is “Text” in all detected cases. Note that the bounding boxes are all quite tight and encapsulate the text regions fairly accurately. Modifying a model like MaskRCNN and training it with OCR-relevant datasets leads to an effective approach for text localization. It is of interest to note that even though the pixel masks are not inherently required for OCR, we have observed that including this constraint in the multi-task learning forces the localization (bounding box regression) to be even more accurate.

If all we can do is identify RoIs of an image that correspond to text blocks, then this is obviously of limited utility for OCR. But what we need to do next is *read* the text contained in each image region; this is known as *text recognition*. The model described below is a departure from MaskRCNN and is a multi-task network for solving both text localization and text recognition.

Multi-Task Network for Text Extraction

Taking inspiration from models like MaskRCNN, we have designed our own multi-task network for solving both text localization and text recognition. Similar to previous approaches, our model entails a convolutional backbone for extracting image features. We have evaluated [both ResNet and Densely Connected Convolutional Networks \(DenseNet\)](#), for which we find that

DenseNet leads to higher accuracy. Additionally, the output of the convolutional stack is then input into a [Feature Pyramid Network](#) which helps to combine high spatial resolution information from early in the stack with low-resolution but rich semantic detail from deeper in the stack. These form the basis of the convolutional backbone which is then sent to the model heads.

Similar to recent object detection models, the text localization head is comprised of a two-stage mechanism with a Region Proposal Network followed by a Bounding Box Regression network. The output of the latter component is a set of predicted boxes (RoIs) that might contain text. The second head of the model is the classification component which is tasked with estimating the class of object contained inside of each RoI — in this case, a simple binary classification (text vs. background). Finally, we have the text recognition head which takes as input the feature maps from the convolutional backbone and the RoI coordinates generated from the text localization head. This text recognition head must, for each RoI, produce a predicted sequence corresponding to the text inside each box and uses a [CTC loss](#) for training.

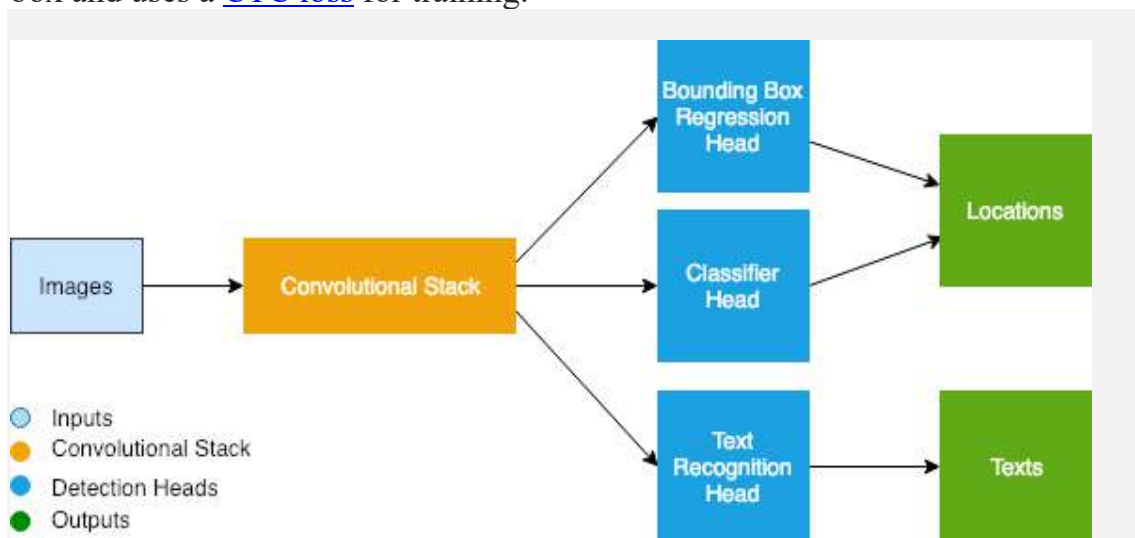


Figure 4: A simplified architecture of our multi-task model. Input images are passed through a convolutional stack which includes ResNet as well as Feature Pyramid Network. The resulting representations are then used by the model's multiple heads in order to identify text locations and sequences.

The text recognition head is a primary point of departure for our model as compared to object detection methods, so some additional detail is worthwhile. In object detection, feature maps are extracted from multiple levels of the convolutional backbone and are pooled to a fixed representation through a mechanism known as RoIPool (or RoIAlign). In OCR, we must retain high spatial resolution information, so we extract features only from early blocks of the convolutional backbone. Additionally, we rely on a new pooling mechanism that allows objects of different aspect ratios to be represented without compressing long sequences or stretching short ones.

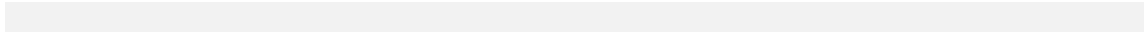
Additionally, one is presented with vast possibilities for how to implement the text recognition head. A simple approach might be to crop the input image as identified by the [RoIs from the Bound Box Regression head and then process this cropped image through an RNN architecture](#) [16]. The limitation of such an approach is that we do not reuse image feature representations between the heads, which would require the recognition head to do more computation on its own. Instead, our multi-task network proceeds by using the identified RoIs and then fetching the relevant representations for each region from the convolutional backbone. Each RoI's feature map is then transformed into a fixed shape as described above and the text recognition can proceed.

A typical approach for sequence classification and sequence labelling tasks would be to use an RNN architecture of some kind where we sequentially process from “left to right” across the RoI's approximately 200 horizontal spatial steps and try to predict an output label for each location in the RoI feature map. However, we find that RNNs perform quite poorly for text recognition here. This is likely due to the fact that we don't really need to account for long-range correlations in this sequence in order to decode the text. Instead, we only need to look at a few “feature columns” at a time in order to get a sense of which

character is being represented. For this reason, we find it most useful to use simple convolutional methods here with short-range kernel widths. At each spatial step, the output of convolutions is used to predict an output letter, and then the overall sequence is collapsed through the CTC layer to output the final sequence for the ROI.

In order to train the model described here, we would need a large number of labeled images. In lieu of tagging and generating these manually, we instead chose to develop our own synthetic training documents. With enough variability in fonts, sizes, colors, distractions objects, and so on, our synthetic data should result in a model which is able to perform well on real-world images. We generated around ten thousand such images which led to strong performance on the real world cases we highlight here.

Example outputs of our model are shown below. On the left is one of the images from the ICDAR DeTEXT challenge and on the right is a screenshot of a receipt. Text segments are identified apart from background pixels and other image objects and are highlighted with dashed lines. The predicted text sequence for each ROI is shown in red above each box. Note that this model is able to accurately identify text object which occur with a large variety of aspect ratios, fonts, font sizes, and colors.



increasing efficiency of document processing pipelines and sheds light on the power of machine learning to transform the future of work.