LIBDS - Library Incorrect Book Detection System using SOTA Deep Learning Object Detection

Kartik Chopra, *Student, Institute of Systems Science, National University of Singapore, Singapore 119615,*
Matthew Chin Heng Chua, *Lecturer, Institute of Systems Science, National University of Singapore, Singapore 119615*

***Abstract*—Libraries are a source of knowledge as well as a warehouse for books. Managing a library is a difficult task as it involves lots of sorting and stocking of books. Often due to human errors, library books get misplaced in the wrong shelves. This not only creates a task for the librarian, who needs to sort the books but also for the next fellow customer who needs to find a book. To solve the problem of misplaced library books, we came up with a solution involving deep learning techniques, that performs object detection and leverages computer vision to identify misplaced books and generate a report for the same.**

***Index Terms*—Object Detection, Computer Vision, Deep Learning**

I. Introduction

LIBRARIES are an asset to all educational institutions as well as to public. They provide a wide variety of knowledge under a single roof. More number of books means more tasks for the person in charge. Tasks such as stocking and sorting take majority of the time since the books need to be placed in order in the assigned shelves. This requires manual labour and the person in charge has to do several rounds of the library, covering each shelf, looking at each row, to identify misplaced books. This is prone to error and is laborious. Intelligent robotic systems can help reduce manual labour and help in the identification of misplaced books. These autonomous systems work on three main principles - Perception, Decision and Actuation. In this paper, we focus on the perception aspect of the intelligent system. We develop a deep learning vision based system capable of identifying book labels, shelf labels. Post detection we use computer vision techniques and text extraction techniques to match the book labels to the correct shelf labels.

Deep learning is a branch of Artificial Intelligence that combines feature extraction and classification into a single framework [1]. They are known to improve state-of-the-art speech recognition, object recognition models among many other domains. We leverage deep learning to perform object detection for our custom dataset. Computer vision is a field of study that enables computers to understand high-level information from images and videos. OpenCV is a library of functions that aims at real-time computer vision. Originally written in C/C++, it has expanded its support to Python and we use OpenCV with Python to perform image enhancement. Optical character Recognition (OCR) is a technique of extracting text from images. We use Pytesseract, to perform OCR to extract text from images. Details of all the techniques are discussed in III.

The remainder of the paper is organized as follows. In II we present an overview of the related works. III shows the proposed approach. In IV we discuss the experimental results and finally in V we offer our conclusions.

II. Related Work

Several works have been done for the use case of detecting books and bookshelves [2]–[6]. In [7], the authors focus on detecting books placed on shelf by first detecting the bookshelf rows using a horizontal line detection method. Then, to detect the spines of the books, they implement a angular line detection method based on the assumption that all books are not placed upright. However, this technique does not extract text from book labels to identify misplaced books. In [8] boundaries of books were taken into account to extract the books and the images were always taken at a fixed orientation. The authors also extracted the entire book title instead of the book label. This approach could be deemed unsuccessful if the image does not comply with the fixed orientation or have multiple rows in a single image. A model-based boundary detection technique is employed in [9]. It estimates the boundary and spine, and the local slant angle of the books and then isolate individual book images. It extracts the title of the book by a vertical projection histogram of its near-horizontal edges. For this work as well book tags are not extracted. In [10] the authors use real-time imaging and a local segmentation algorithm with automatic threshold detection for detecting book tags. Here the authors make a lot of assumptions about the environment being static, all tags are composed of 4 text lines and non-overlapping labels among others. Due to these assumptions, the solution is not robust and the problem statement becomes extremely constrained. In [11] the authors perform book region extraction using high frequency filtering followed by book segmentation. They utilize two techniques, line segment and MSAC (m-estimator sample consensus) based calculation of dominant vertical point [12]. Authors in [13] perform book spine extraction followed by book spine recognition and then use OCR to extract text from the books.

III. Proposed approach

As seen from the previous section, it can be noted that current techniques do not leverage deep learning methodologies for this task. Although, many computer vision techniques have been used but they are not proven to be as accurate as the

deep learning techniques. In our approach, we leverage deep learning algorithms to achieve the best results possible. Earlier works suggests on detection of book spines and extracting text after the detection of the spine purely based on computer vision techniques.

The next few sections explain the major components of the paper beginning with Label Detection Models, ROI Enhancement and Text Extraction and finally Matching and Reporting.

A. Label Detection Model

In III we talked about using deep learning algorithms. We tried 2 deep learning models for object detection customised to our dataset namely, *YoloV3-Tiny* and *YoloV3* which will be explained in detail in section III-A1 and III-A2 respectively. These two algorithms have been trained over 117k images and for over 80 classes. We use the pretrained weights of both the models and implement it to our usecase.

1) YoloV3-Tiny: YoloV3-Tiny is an adaptation of YoloV3 with considerably less number of layers. It has 23 layers as opposed to YoloV3 which has 107 layers in total. YoloV3-Tiny is used to give real-time performance on miniaturized hardware embedded devices which would have been ideal for our use case of deploying the model on a drone [14]. The YoloV3-Tiny model was implemented in PyTorch. A total of 40 training images and 10 test images were used. As stated earlier, YoloV3-Tiny has less layers and is known for its accuracy, it did not perform well on our dataset as it fails to detect small objects. Even after training for over 500 epochs and fine-tuning over another 500 epochs, the results were not satisfactory with the average precision of only 17.39%. Hence we decided to move on to YoloV3 and instead of running the model on drone, we would run the model on the server as it is a much heavier model.

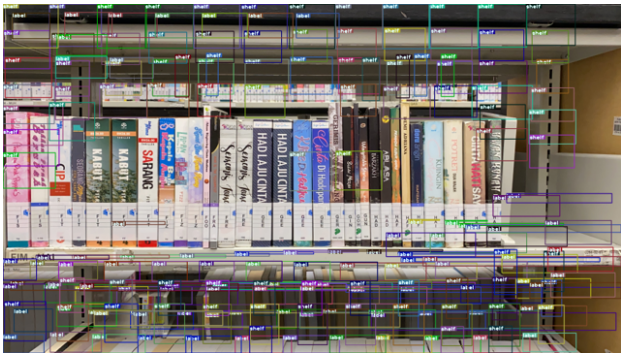

Fig. 1. YoloV3-Tiny predictions fail to detect even a single correct label.

2) YoloV3: Yolo, short for, "You Only Look Once" was first developed by Joseph Redmon et al [15]. YoloV3 is an improvement of Yolo which is a real-time object detection deep learning algorithm. A traditional object detection model first extracts features from input images [16]–[19]. After the features are extracted, the model then proceeds to identify objects based on these extracted features [20]–[23]. However, YoloV3 unifies separate components of object detection into a single network. It divides the image into a S x S grid and if the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.

TABLE I
YOLOV3-TINY ARCHITECTURE

Type	Filters	Size/Stride	Input	Output
Conv	16	3×3/1	416×416×3	416×416×16
Maxpool	2×2/2	416×416×16	208×208×16	
Conv	32	3×3/1	208×208×16	208×208×32
Maxpool	2×2/2	208×208×32	104×104×32	
Conv	64	3×3/1	104×104×32	104×104×64
Maxpool	2×2/2	104×104×64	52×52×64	
Conv	128	3×3/1	52×52×64	52×52×128
Maxpool	2×2/2	52×52×128	26×26×128	
Conv	256	3×3/1	26×26×128	26×26×256
Maxpool	2×2/2	26×26×256	13×13×256	
Conv	512	3×3/1	13×13×256	13×13×512
Maxpool	2×2/1	13×13×512	13×13×512	
Conv	1024	3×3/1	13×13×512	13×13×1024
Conv	256	1×1/1	13×13×1024	13×13×256
Conv	512	3×3/1	13×13×256	13×13×512
Conv	255	1×1/1	13×13×512	13×13×255
YOLO				
Route	13			
Conv	128	1×1/1	13×13×256	13×13×128
Upsampling	2×2/1	13×13×128	26×26×128	
Route	19	8		
Conv	256	3×3/1	13×13×384	13×13×256
Conv	255	1×1/1	13×13×256	13×13×256
YOLO				

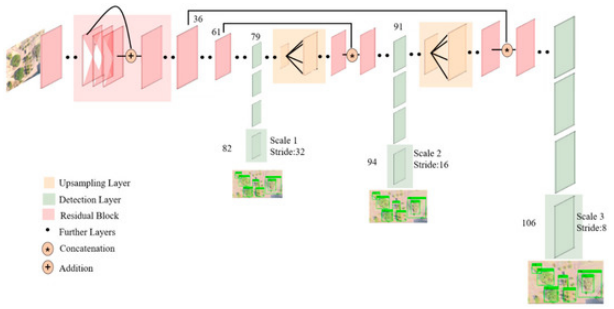

Fig. 2. YOLOv3 architectural summary

Our model comprises of complex layering of multiple residual blocks with initial layers from VGG-16 image classifier. These layers perform the low level feature extraction. The network outputs predictions at three different levels which are then processed individually using anchor boxes. Figure 2 shows a generalized version of architecture [24].

As stated in III-A1, YoloV3-Tiny fails to detect objects for our use case, we move to YoloV3. On our custom dataset, the model performs well enough to solve the problem of correctly detecting the labels. Further experimental settings and results are presented in IV.

B. ROI Enhancement and Text Extraction

After the labels are detected using YoloV3 from III-A2, we use the coordinates of the bounding boxes to extract the region of interest (ROI). Before extracting the ROI, there are possibilities that more than one shelf is detected and we need to match the book labels to the correct shelf. In general, the shelf label is located at bottom left and the book labels are to be matched to that shelf and not the shelf on the top. Hence

Fig. 3. Sample predictions from YoloV3 Model.

the diagonal corners of bounding boxes of book labels are first matched with each shelf label and then the shelf that corresponds to the bottom left of the book labels is selected as the ROI for shelf.

Fig. 4. Matching the books to their corresponding shelves.

We leverage computer vision techniques to enhance the ROI before extracting the text from it. This gives a better text extraction accuracy. We use Python with OpenCV to perform the enhancement. First, we crop the extracted ROI to focus on the center text. Then we use pixel-wise binary thresholding for book labels and truncated thresholding for shelf labels. After performing the thresholding, we use dilation to further enhance the text labels. A dilation of kernel size 3×3 is applied for book labels and a kernel size of 5×5 is applied for shelf labels.

Fig. 5. Enhanced ROI for text extraction

Once the ROIs are enhanced and the text is clearly visible, we use Pytesseract, which is an Optical Character Recognition (OCR) tool for python. This off the shelf tool has various features that help in extraction of text with varied orientations. Since in our use case, the book labels have vertical text, and shelf labels have horizontal text, Pytesseract was a good fit. To further convert the extracted characters into strings, several adjustments were done. These include, removal of non-capitalized alphabets, removal of special symbols, removal of any numbers and limiting the text extracted to 3 characters for book labels and $3-7$ characters for shelf labels.

C. Matching and Reporting

After enhancing the ROIs and extracting the text as discussed in III-B, we perform string comparison to check if the book belongs to the correct shelf or not. We iterate over all the book labels in a particular image and match the text of the book labels to that of the selected book shelf. Book shelves have 2 kinds of labels. Either it can be a range, for example, *ABC-ABE* or they can be a fixed string, *ABC*.

While iterating over each book, we mark the book as incorrect or correct. This gives us the ability to generate a report for each shelf. We generate a table with the book shelf name, misplaced book label name and the image of the shelf. Doing so would help the librarian or the person in charge to easily identify the misplaced books and it can greatly reduce the time that he has to spend looking for misplaced books.

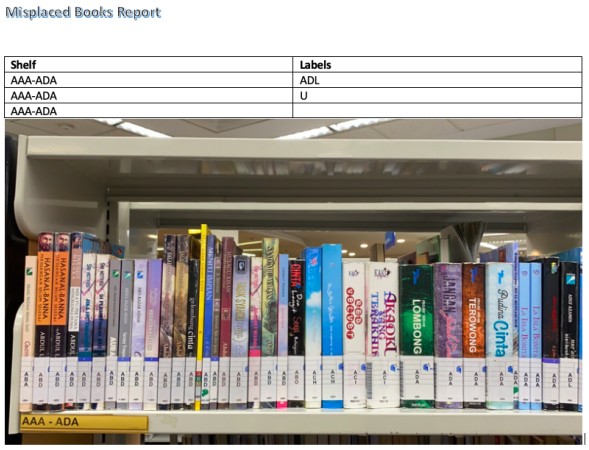

Fig. 6. Sample of report generated.

IV. Experiments and Results

A. Dataset

There are not many readily available datasets of library books that could have been used for our use case. Hence we decided to create our own dataset. For this, we visited the local public libraries in Singapore, Clementi Public Library, and took pictures of the books and shelves. Since our plan for the use of this system is to ultimately mount it on a drone, we took images from the drone point of view by mimicking the movement of the drone using our phone camera. We took images at a resolution of 4032×3024 using iPhone X default camera application. We collected over 120 images but used about 60 images for model training and testing.

Fig. 7. Examples of book and shelves images used in the experiment.

B. Image Annotation and Labeling

As explained in IV-A, we collected our own images to build a dataset, we had to annonate the images ourselves. We used *LabelImg* to annonate and label our images. The data consists of 2 classes, *Label* (book labels) and *Shelf* (shelf labels). Each image had approximately 23-25 book labels and 1-2 shelf labels. This gave us a total of around 1492 book labels and 70 shelf labels.

Fig. 8. Annotations of book and shelves used in the experiment.

C. Experimental Configuration

Our final selected model YoloV3 used a total of 60 images for training and testing, split into 50 images for training and 10 images for testing. A total of 1492 labels for book labels and 70 labels for shelf labels were annotated and fed into the model for training. The training was run for over 6000 epochs with a learning rate α set at 0.001, batch size of 16, height and width of each input as 608 x 608. For predections on the dataset, a minimum confidence was set to 50% while the Non-Maximum Supression (nms) threshold was set to 45%. These values were chosen to have as many detection results as possible, since the book label class has multiple objects placed in very close vicinity.

TABLE II
DATASET CONFIGURATION

	Training	Testing	Total
Images	50	10	60
Book Labels	1256	236	1492
Shelf Labels	56	14	70

TABLE III
EXPERIMENTAL RESULTS OF VARIOUS METRICS.

Class/Metric	Book Label	Shelf Label	Mean
Average Precision	87.34%	100%	93.67%

D. Results

Unlike classification models, object detection models use the metrics of mean Intersection over Unioin (mIOU) and mean Average Precision (mAP) as a measure of goodness of a model as opposed to accuracy. For our model, the results are presented in III. It can be seen that although the shelves have a 100% average precision, the book label class has a lesser mAP due to the books being closer to one another. The mIoU is 65.42% which is lower than mAP.

V. CONCLUSION

In this paper, we present a novel approach to detect misplaced library books and generate a report for the same. We use deep learning model, YoloV3, to perform object detection, book label detection and shelf label detection. After identifying the objects, we enhance the ROI using OpenCV image processing techniques. Once the ROI is enhanced, we use Pytesseract to extract text from the labels. After extracting the text, we perform a text based matching and generate a misplaced book report.

REFERENCES

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] Sam S. Tsai, David Chen, Huizhong Chen, Cheng-Hsin Hsu, Kyu-Han Kim, Jatinder P. Singh, and Bernd Girod, "Combining image and text features: A hybrid approach to mobile book spine recognition," in *Proceedings of the 19th ACM International Conference on Multimedia*, New York, NY, USA, 2011, MM '11, p. 1029–1032, Association for Computing Machinery.

[3] David M Chen, Sam S Tsai, Bernd Girod, Cheng-Hsin Hsu, Kyu-Han Kim, and Jatinder Pal Singh, "Building book inventories using smartphones," in *Proceedings of the 18th ACM international conference on Multimedia*, 2010, pp. 651–654.

[4] Danny Crasto, Amit Kale, and Christopher Jaynes, "The smart bookshelf: A study of camera projector scene augmentation of an everyday environment," in *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05)-Volume 1*. IEEE, 2005, vol. 1, pp. 218–225.

[5] Markus Löchtefeld, Sven Gehring, Johannes Schöning, and Antonio Krüger, "Shelftorchlight: Augmenting a shelf using a camera projector unit," in *UBIProjection 2010 - Workshop on Personal Projection*, 2010.

[6] Kazuhiro Matsushita, Daisuke Iwai, and Kosuke Sato, "Interactive bookshelf surface for in situ book searching and storing support," in *Proceedings of the 2nd Augmented Human International Conference*, New York, NY, USA, 2011, AH '11, Association for Computing Machinery.

[7] M. I. Jubair and P. Banik, "A technique to detect books from library bookshelf image," in *2013 IEEE 9th International Conference on Computational Cybernetics (ICCC)*, 2013, pp. 359–363.

[8] Yukari Akiyama and Minoru Ito, "Book recognition from color images of book shelves," in *Proc. of IAPR Workshop on MVA'98, pp. 106–110*. Citeseer, 1998.

[9] Eiji Taira, Seiichi Uchida, and Hiroaki Sakoe, "Book boundary detection and title extraction for automatic bookshelf inspection," in *10th Korea-Japan Joint Workshop on Frontiers of Computer Vision*, 2004, pp. 232–237.

[10] Mario Prats, Pedro J Sanz, and AR del Pobil, "Model-based tracking and hybrid force/vision control for the uji librarian robot," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2005, pp. 1090–1095.

[11] Nguyen-Huu Quoc and Won-Ho Choi, "A framework for recognition books on bookshelves," in *International Conference on Intelligent Computing*. Springer, 2009, pp. 386–395.

[12] Philip HS Torr and Andrew Zisserman, "Mlesac: A new robust estimator with application to estimating image geometry," *Computer vision and image understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[13] Sam S Tsai, David Chen, Huizhong Chen, Cheng-Hsin Hsu, Kyu-Han Kim, Jatinder P Singh, and Bernd Girod, "Combining image and text features: a hybrid approach to mobile book spine recognition," in *Proceedings of the 19th ACM international conference on Multimedia*, 2011, pp. 1029–1032.

[14] Ji Wan, Dayong Wang, Steven Chu Hong Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, and Jintao Li, "Deep learning for content-based image retrieval: A comprehensive study," in *Proceedings of the 22nd ACM International Conference on Multimedia*, New York, NY, USA, 2014, MM '14, p. 157–166, Association for Computing Machinery.

[15] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[16] Constantine P Papageorgiou, Michael Oren, and Tomaso Poggio, "A general framework for object detection," in *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, 1998, pp. 555–562.

[17] David G Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*. Ieee, 1999, vol. 2, pp. 1150–1157.

[18] Navneet Dalal and Bill Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. IEEE, 2005, vol. 1, pp. 886–893.

[19] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *International conference on machine learning*, 2014, pp. 647–655.

[20] Paul Viola, Michael Jones, et al., "Robust real-time object detection," *International journal of computer vision*, vol. 4, no. 34-47, pp. 4, 2001.

[21] Rainer Lienhart and Jochen Maydt, "An extended set of haar-like features for rapid object detection," in *Proceedings. international conference on image processing*. IEEE, 2002, vol. 1, pp. I–I.

[22] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[23] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2009.

[24] Sabir Hossain and Deok-jin Lee, "Deep learning-based real-time multiple-object detection and tracking from aerial imagery via a flying robot with gpu-based embedded devices," *Sensors*, vol. 19, no. 15, pp. 3371, 2019.