BIG DATA MINING AND ANALYTICS
ISSN 2096-0654 06/06 pp279–297
Volume 4, Number 4, December 2021
DOI: 10.26599/BDMA.2021.9020012

Intelligent and Adaptive Web Data Extraction System Using Convolutional and Long Short-Term Memory Deep Learning Networks

Sudhir Kumar Patnaik*, C. Narendra Babu, and Mukul Bhave

Abstract: Data are crucial to the growth of e-commerce in today's world of highly demanding hyper-personalized consumer experiences, which are collected using advanced web scraping technologies. However, core data extraction engines fail because they cannot adapt to the dynamic changes in website content. This study investigates an intelligent and adaptive web data extraction system with convolutional and Long Short-Term Memory (LSTM) networks to enable automated web page detection using the You only look once (Yolo) algorithm and Tesseract LSTM to extract product details, which are detected as images from web pages. This state-of-the-art system does not need a core data extraction engine, and thus can adapt to dynamic changes in website layout. Experiments conducted on real-world retail cases demonstrate an image detection (precision) and character extraction accuracy (precision) of 97% and 99%, respectively. In addition, a mean average precision of 74%, with an input dataset of 45 objects or images, is obtained.

Key words: adaptive web scraping; deep learning; Long Short-Term Memory (LSTM); Web data extraction; You only look once (Yolo)

1 Introduction

Recent advancements in machine learning and Artificial Intelligence (AI) have unfolded new opportunities, even in extensively studied research programs in numerous domains, including medical imaging (e.g., image recognition), transportation (feature extraction in self-driving cars)[1,2], and traffic scenarios (e.g., object detection)[3,4]. These advancements also encourage the extraction of relevant information from documents (pdf, doc, or txt files), websites, and images that use Optical Character Recognition (OCR)[5], subsequently inspiring the development of automated web data extraction systems through leading edge technology solutions[6,7]. The application of deep learning in web data extraction[8,9] is still in its nascent stage; in addition to extracting data from documents or web pages, this application involves navigating different websites and storing data for analytics and visualization purposes.

Although web application development is making rapid technological change in terms of rendering content dynamically and embedding websites within restricted browser environments, maintaining core data extraction engines still remains a serious challenge[6], which requires frequent and manual intervention. Manually fixing extraction engine scripts for hundreds of websites is cumbersome and inefficient, thus urging the development of adaptive web data extraction systems that can adapt to dynamic changes in websites[10]. Some of the available web data extraction techniques,

• Sudhir Kumar Patnaik is with the Department of Computer Science and Engineering, M. S. Ramaiah University of Applied Sciences, Bangalore 560054, India, and also with Gibraltar India Solutions LLP, Bangalore 560103, India. E-mail: skpatnaik9@gmail.com.
• C. Narendra Babu is with the Department of Computer Science and Engineering, M. S. Ramaiah University of Applied Sciences, Bangalore 560054, India. E-mail: narendrababu.c@gmail.com.
• Mukul Bhave is with Gibraltar India Solutions LLP, Bangalore 560103, India. E-mail: mukulbhave@gmail.com.
* To whom correspondence should be addressed.

Manuscript received: 2021-04-18; revised: 2021-05-30; accepted: 2021-06-28

that are used in such adaptive systems are discussed below. Web data extraction is explored using repetitive blocks[11], with their respective attributes obtained from classification-based approaches. This data extraction technique demonstrates good accuracy and adaptability to layout changes in websites. However, it assumes websites to have repetitive blocks and such an assumption is not always possible. Another approach that uses weighted tree matching clustering algorithms has been developed to extract records from websites; this approach can be applied to structured (product descriptions) and unstructured records (e.g., blogs)[12].

The voluminous growth of web data proportionally adds challenges to the development of adaptive web data extraction techniques. A detailed review[13] summarized deep web data extraction technologies and discussed alternate data extraction methods for structured and unstructured data. The characteristics of the deep web (e.g., wider data coverage, higher quality, and strong structure) make the development of deep web data extraction techniques challenging[14]. Moreover, the large amount of data hidden in web pages with noisy content (in the form of advertisements, page settings, and navigation buttons) institute further challenges[15–17]. In this regard, techniques based on AI and machine learning[5], including deep learning, can potentially accelerate the development of adaptive web data extraction techniques. One way to extract such data is to use the fast Region-based Convolutional Neural Network (R-CNN) model to filter these images by using layers in artificial neural network model, extract hidden data from web pages, and then store them in a database[18]. Similarly, data extraction techniques with web wrappers have continued to mature with the evolution of machine and deep learning techniques. Contemporary web wrappers adapt to specific web templates before initiating extraction process. Such web wrappers use fast R-CNN to learn from previous templates[19], leading to a site independent web wrapper, which however, requires their results to be empirically validated. Some web wrappers can not check tree structure similarity accurately in the deep web due to the use of the Document Object Model (DOM) tree of records[20]. In such cases, ontological techniques are used, so that wrappers can reduce data regions and improve extraction accuracy. Empowered with script edit based machine learning techniques, wrappers can also adapt to dynamic changes in website layout[21] by using an extraction model to train new wrappers that automatically extract the target data with improved accuracy. Classical object detection problems in computer vision and image processing have also evolved by using deep learning methods, especially those with Convolutional Neural Network (CNN), but not applied for web data extraction[20]. The application of CNN is typically used for accurate object detection[13], semantic segmentation[22] (using selective search algorithm to propose possible regions of interest)[16,23] and object classification.

The You only look once (Yolo)[24] deep learning model was conceptualized in 2015. It works based on (1) image classification and (2) object localization. In image classification, an image is assigned to numberous categories, such as "electronics" and "book", with only one category assigned to it. This method quickly evolved into fast R-CNN[25] by using the region of interest pooling technique. Subsequently faster R-CNN[26] and Mask R-CNN[17] models emerged as the first fully differentiable models. Furthermore, Yolo was introduced to enable real time image or object detection[27,28]. Table 1 shows the evolution of image or object recognition technologies, models, and algorithms in recent years. Yolo uses classification and regression algorithms for image classification and localization. After selecting the regions, Yolo uses the idea of anchor boxes, but instead of selecting the anchor boxes manually, finding the best anchor boxes makes it easier

Table 1 Deep learning models and techniques.

Model	Algorithm/Technique
R-CNN	Dataset: ImageNet Classification: Binary SVM
Fast R-CNN	RoI pooling
Faster R-CNN	RPN
Mask R-CNN	RoI align (pixel level segmentation)
RetinaNet	ResNet, Feature Pyramid Network (FPN)
Single Shot Detector (SSD)	Single deep neural network, feed forward convolutional network
Histogram of Oriented Gradients (HOG)	Detection window, RoI
Region-Fully Convolutional Network (R-FCN)	Convolutional + RoI
Spatial Pyramid Pooling (SPP)	Pyramid pooling
Yolo	Classification and regression, Deformable Parts Model (DPM) and R-CNN

for the network to learn as shown in Table 1 with an aim to predict class and bounding box specifying object location.

Table 2 compares traditional, machine learning, and deep learning tools in manual and automated web data extraction. It reveals that much of the literature focused on automated web data extraction using machine learning[39] and neural network techniques for deep web data extraction[20]. The above discussion also suggests that the above-mentioned approaches did not use image-based recognition as a technology for data extraction. Furthermore, traditional and machine learning techniques focused on data accuracy, rather than building intelligent and adaptive web data extraction systems, with the exception of Refs. [10, 21], where some structuralism of the website is assumed for developing adaptive web scrapers.

Today, web data extraction is performed using contemporary and machine learning techniques, as shown in Table 2 with 99% data extraction accuracy. The techniques followed are DOM-based by using automated wrapper generation algorithms[8], which makes the system susceptible to website layout changes, despite progress in development of automated information extraction systems[10, 12, 21].

The proposed system in this study primarily addresses this gap in the contemporary and machine learning techniques by using deep learning object detection technology, such as Yolo, and eliminates DOM or wrapper techniques for data extraction. The novelty in the proposed system is based on object detection technique, and therefore lends itself to detecting the object irrespective of its location in the website. Once the object or product page in a website is detected, LSTM is used for extracting the product detail from the object. This approach to detect the object or product page makes the proposed web data extraction system adaptable to dynamic changes in the website. The proposed web data extraction system using Yolo and Tesseract LSTM deep learning networks will demonstrate two key concepts to prove that:

- web data extraction system is intelligent by handling the image identification, detection and extraction of data from web page,
- proposed system is adaptive, and extracts data from the new location or layout of the web page due to change in website layout, and
- proposed system is domain agnostic and can handle data extraction in nonretail domain.

The contribution of this study is to build a state-of-the-art intelligent and adaptive web data extraction system to

- extract data with a deep learning based object detection technique using Yolo and Tesseract Long Short-Term Memory (LSTM) networks,
- eliminate traditional or machine learning based core data extraction engines, and
- build self-correction capability in a web data extraction system with an adaptive capability to handle dynamic changes in the website layout[10, 21].

To the best of the authors' knowledge, the proposed approach is one of its kinds in the literature. The remaining part of this paper are structured as follows: Section 2 defines the problem statement outlining the issues in the web data extraction system due to changes in website layouts. Design and architecture of self-correction capability with Yolo[24] and LSTM neural networks[39], including input data modeling, are discussed in Section 3. Section 4 presents the experiment setup process, input data modeling, and results from seven real-world cases conducted with the

Table 2 Comparison of traditional, machine learning, and deep learning based web data extraction tools.

Tool	Extraction rule	Technique	Precision (%)	Self-healing
TSIMMIS[29]	Wrapper-based	Traditional/statistical	Not available	No
WebOQL[30]	Tag tree	Traditional/statistical	Not available	No
WHISK[31]	Regular expression	Supervised learning	69	No
RAPIER[32]	Logic rules	Supervised learning	89	No
SRV[33]	Logic rules	Supervised learning	58	No
SoftMealy[34]	Regular expression	Supervised learning	58	No
DEPTA[35]	Tag tree	Un-supervised learning	98	No
Trinity[36]	Regular expression	Un-supervised learning	96	No
DeLA[37]	Regular expression	Un-supervised learning	80	No
OLERA[38]	Regular expression	Semi-supervised learning	99	No
Proposed system	Object detection	Deep learning	To be determined	Yes

proposed system. Section 5 highlights the critical review, limitations, and future scope. Concluding remarks are summarized in Section 6.

2 Problem Definition

Data extraction is accomplished through web scraping techniques, such as open source, commercial crawlers, and application program interface. "Extraction Engine" is one of the core subcomponents in traditional, machine learning, and deep learning based automated end-to-end web data extraction systems, as shown in Fig. 1[40]. It is implemented using various techniques, such as automated wrapper generation[21], tree-based DOM[41], pattern matching[12], and HyperText Markup Language (HTML) tags, which assume some form of data structuralism in the websites. Figure 2 shows traditional, machine learning, and deep learning based web data extraction systems with core data extraction engine, and their subcomponents are explained as follows:

• **Crawler**: A web crawler downloads and indexes content from all over the Internet. It follows certain policies that make it more selective about which pages to crawl, in what order to crawl them, and how often they should crawl them again to check for content updates. Web crawlers start from a list of known URLs and then

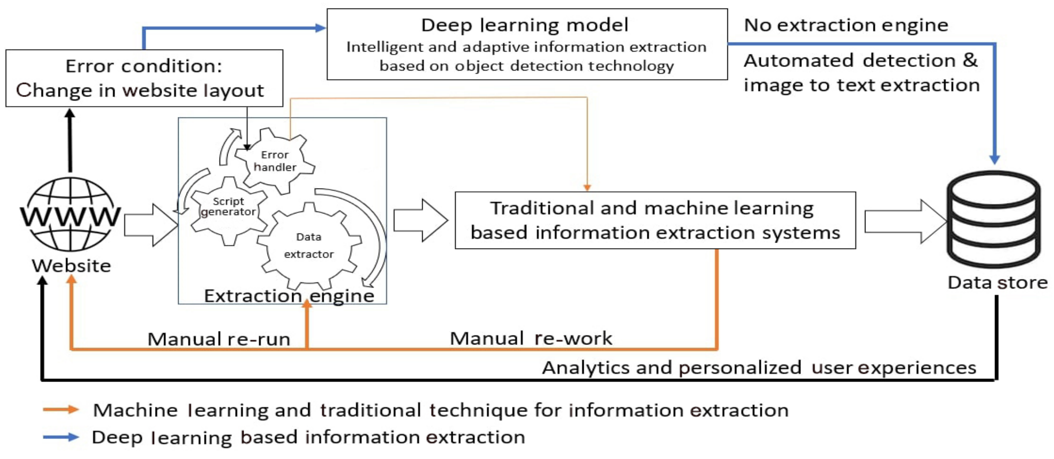

Deep learning model
Intelligent and adaptive information extraction based on object detection technology
No extraction engine
Automated detection & image to text extraction
Error condition: Change in website layout
Error handler
Script generator
Data extractor
Extraction engine
WWW
Website
Traditional and machine learning based information extraction systems
Data store
Manual re-run
Manual re-work
Analytics and personalized user experiences
Machine learning and traditional technique for information extraction
Deep learning based information extraction

Fig. 1 Traditional, machine learning, and deep learning based web data extraction system.

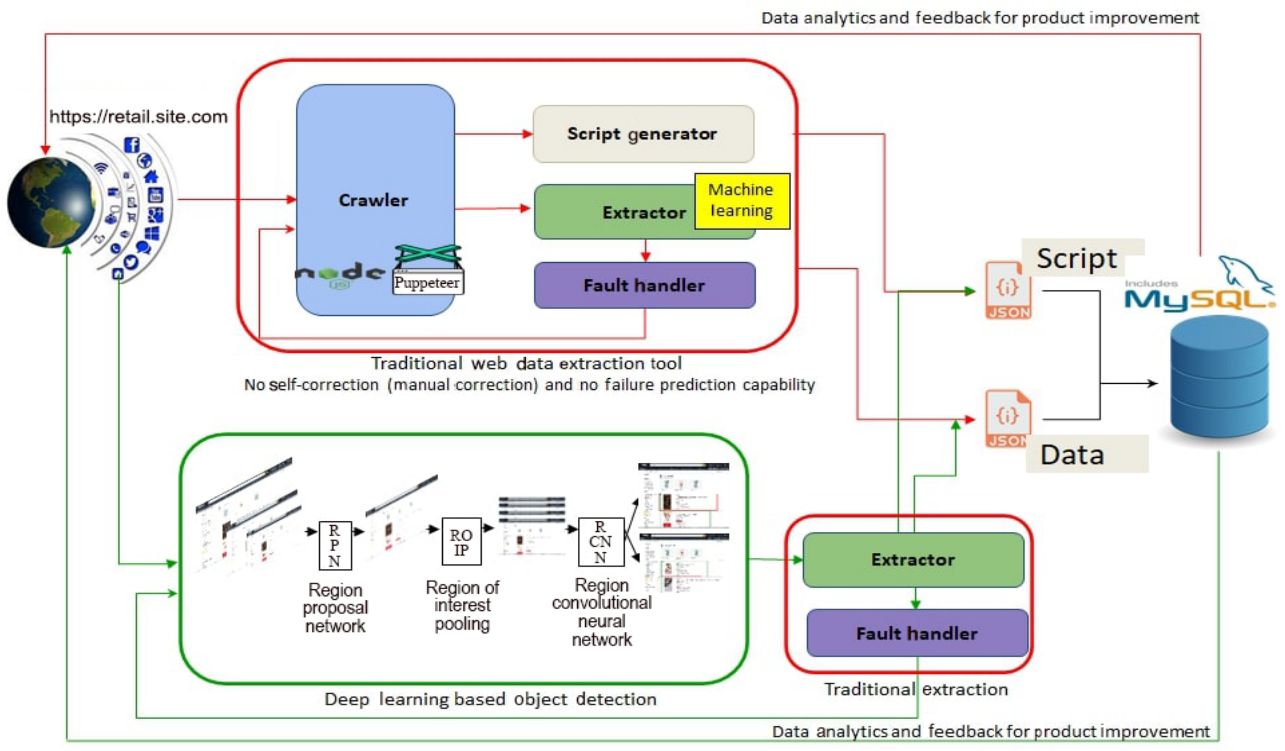

Data analytics and feedback for product improvement
https://retail.site.com
Crawler
Puppeteer
Script generator
Extractor
Machine learning
Fault handler
Traditional web data extraction tool
No self-correction (manual correction) and no failure prediction capability
Script
Data
MySQL
RPN
Region proposal network
ROIP
Region of interest pooling
RCNN
Region convolutional neural network
Deep learning based object detection
Extractor
Fault handler
Traditional extraction
Data analytics and feedback for product improvement

Fig. 2 Traditional, machine learning, and deep learning based web data extraction system with core data extraction engine.

crawl the web pages at those URLs first. As they crawl those webpages, they will find hyperlinks to other URLs, and they add those to the list of pages to crawl next.

• **Script generator**: These generators are selenium-based record and playback tools that generate Python code that can be rerun for navigation.

• **Data extractor**: Extraction engines can extract specific data, images, and files from any website. It contains crawling rules and an extraction pattern providing efficient and accurate data extraction. The extractor engine automatically scans the provided URLs and scrapes all the information that meets the specified template.

The process involved in manual and automated data extraction is explained as follows:

• **Navigation and categorization**: Navigation is performed to identify the product pages, thereby obtaining the URL in an e-commerce website and the product is categorized.

• **Extraction**: A core extraction engine is used to identify the sections of the page, thereby obtaining the x-path from where the data of interest is extracted.

The introduction of machine learning and deep learning techniques in the data extraction engine enhances data extraction accuracy and enables automated web data extraction, by following the process as outlined below, while still using a core extraction engine for crawling the data from the webpage, as shown in Fig. 2:

• Selecting parameter and hyper-parameters for input data modeling and training.

• Training the machine learning model with input images and minimizing loss function for improved data accuracy.

• Extracting data from the specific product page and storing in database.

• Gaining insights from the web data extraction process and continuously training the machine learning model for improving data extraction accuracy.

• Analyzing stored data and feeding them back to the website for improving personalized user experience.

2.1 Limitations of traditional, machine learning, and deep learning based web data extraction systems due to changes in website layout

Given the dynamic changes in websites, errors, such as (1) website not found (404 error), (2) change in location of product page, and (3) nonexistent product page, occur for various reasons and break the "data extractor and script generator", as shown in Fig. 3. Data extraction is performed from web pages by (1) using extraction engines with contemporary, machine learning, and deep

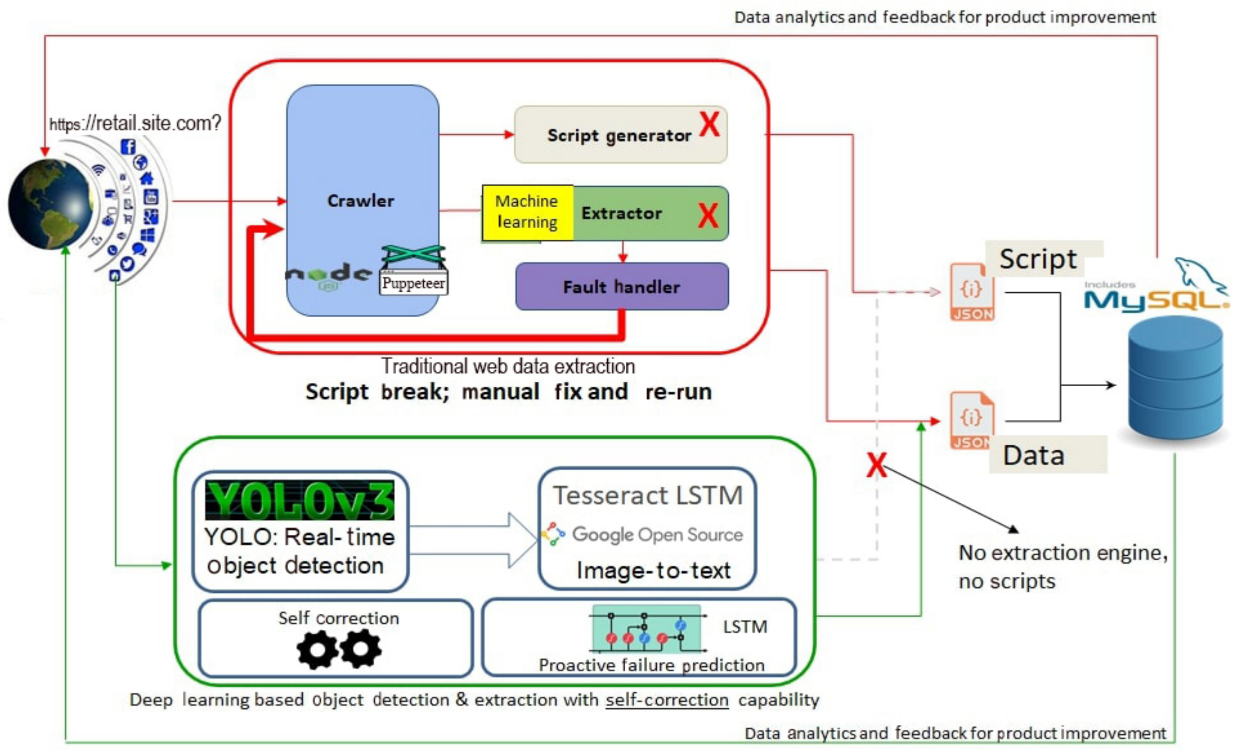

Data analytics and feedback for product improvement
https://retail.site.com?
Crawler
Puppeteer
Script generator X
Machine learning
Extractor X
Fault handler
Traditional web data extraction
Script break; manual fix and re-run
Script
Data
MySQL
YOLOv3
YOLO: Real-time object detection
Tesseract LSTM
Google Open Source
Image-to-text
Self correction
LSTM
Proactive failure prediction
X
No extraction engine, no scripts
Deep learning based object detection & extraction with self-correction capability
Data analytics and feedback for product improvement

Fig. 3 Failure in core data extraction engine using traditional, machine learning, and deep learning technique for automated web data extraction.

learning methods, or (2) using open source tools, such as crawlers. In all the cases, the extraction logic is built into the core extraction engine with contemporary, machine learning, or deep learning techniques. In such circumstances, the script developer usually debugs the error, identifies an appropriate solution, fixes the error in "data extractor/script generator", and re-runs the "extraction engine" to extract correct data. This process is manual and labor intensive, and does not scale to extracting data from large websites. Given that traditional, machine learning, and deep learning methods rely on the core "data extraction engine", these three methods fail in the event of changes in website layout or product page location, as shown in Fig. 3.

2.2 Deep learning and LSTM network based automated web data extraction to handle changes in website layout

The proposed system in this study solves the issue highlighted in Section 2 by eliminating the need for a "core extraction engine" making the proposed automated web data extraction intelligent and adaptive to dynamic changes in websites. By addressing this issue, the proposed system redefines the way data extraction will be performed in the future. The proposed system uses a deep learning based Yolo model for object or image detection and Tesseract LSTM deep learning networks for the extraction of data from images or objects. As a result, the "core extraction engine" is eliminated as shown in Fig. 3 and hence referred to as an intelligent web data extraction system. The proposed end-to-end automated web data extraction system is achieved in a three-step process as follows:

• **Step-1: Changes in website layout:** Changes in website layout are introduced by simulating an experiment where the user accesses the specific product web page (1) without login (as a guest user) and (2) with user login. With user login, the layout of the web page changes. In real-world scenarios, this change is a definite possibility due to other business needs, such as promotions and discounts.

• **Step-2: Auto-navigation:** The image detection feature of the Yolo model enables auto-navigation to the product page (also called "records" [product] and "regions" [product detail], and demands extensive training of the model for high accuracy in locating records and regions.

• **Step-3: Self-correction and automated web data extraction:** Using object detection based deep learning techniques, such as Yolo and image-to-text extraction using Tesseract LSTM, eliminates the core extraction engine; thus, the proposed system demonstrates self-correction and adaptability to dynamic changes in website layout. This capability is demonstrated in Fig. 3, where the "X" mark indicates the core data extraction engine is not needed in the proposed web data extraction system.

2.3 Yolo object detection and Tesseract LSTM image-to-text extraction techniques for automated web data extraction

R-CNN techniques use regions to localize objects within an image, and do not consider the entire image, but only the parts of the images with a high chance of containing an object. By contrast, the Yolo framework handles object detection by taking an entire image in a single instance and predicts the bounding box coordinates and class probabilities for these boxes[27, 28]. Hence, the speed at which it processes (45 frames per second) coupled with the ability to understand generalized object representation provides the framework a unique advantage over the R-CNN family of object detection techniques as shown in Table 1. Recently, Yolo has evolved to be one of the best algorithms for object detection with a comparatively similar performance to R-CNN algorithms. Hence, the proposed web data extraction system uses the Yolo architecture and an algorithm for object detection, which is briefly discussed in Section 3. Tesseract LSTM is an OCR technique used for extracting text from images[39]. It uses CNNs to recognize images by using recurrent neural networks[16] and LSTM. The input image is processed in boxes line by line and inserted into the LSTM model; then, the output is generated in the form of a text or Excel file. The proposed web data extraction system uses the Tesseract LSTM architecture and algorithm for image-to-text extraction, which is briefly discussed in Section 3.

3 System Architecture and Proposed Algorithm

3.1 System architecture

The end-to-end automated data architecture, including its subcomponents, is depicted in Fig. 4. However, the Yolo model is a new core component that enables self-correction. It is introduced as a novel method; hence, the proposed architecture discusses Yolo and the self-correction architecture. The Yolo architecture shown

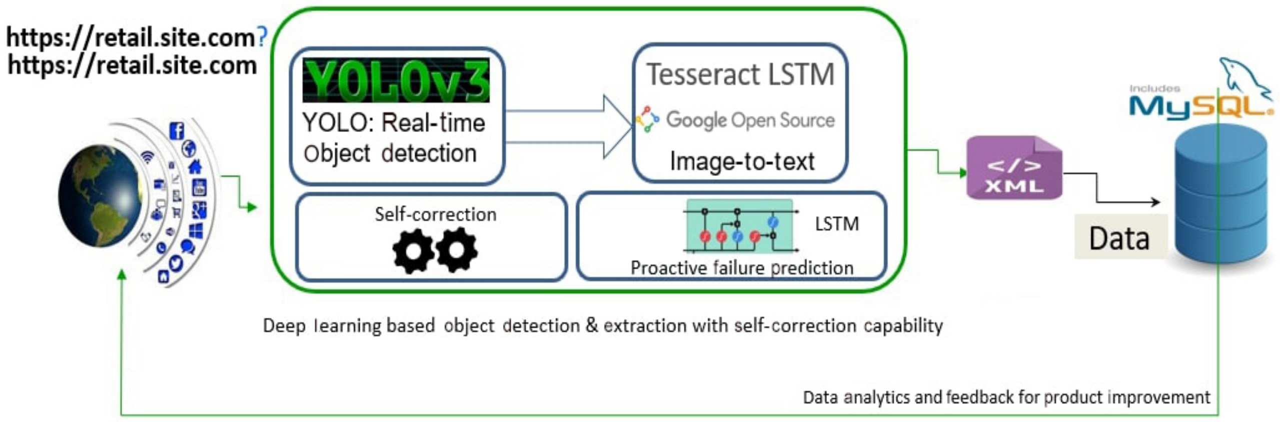

https://retail.site.com?
https://retail.site.com
YOLOv3
YOLO: Real-time Object detection
Tesseract LSTM
Google Open Source
Image-to-text
Self-correction
LSTM
Proactive failure prediction
Deep learning based object detection & extraction with self-correction capability
</> XML
MySQL
Data
Data analytics and feedback for product improvement

Fig. 4 End-to-end automated web data extraction system architecture using Yolo and Tesseract.

in Fig. 4 starts with an image, from which (1) a list of records with bounding box is obtained within each record, (2) a label is assigned to each region with the bounding box, and (3) a probability score is obtained for each label and bounding box. The Yolo architecture has the following subcomponents:

- **Convolutional layer:** In this layer, filters are trained to extract appropriate features from the image. These filters then learn the desired object features. Convolution is computed by sliding the filter all along input images and the result is a two dimension matrix called a feature map. As an example, the input object is a set of image files of a product (e.g., "book"), where the CNN extracts features of different "book" objects; then, applies a filtering process to reach the desired "book" object.
- **MaxPool layer:** The features computed by CNN are used to find a predefined number of regions (bounding boxes), which may contain objects. As an example, once the "book" object is identified, the MaxPool layer then applies the bounding box to the regions (i.e., product detail within the "book" object [e.g., "author", "price", and "title"]). From these regions, data are extracted. The object could have several regions; therefore, classification and bounding box are applied.
- **Classes and bounding boxes prediction:** Regions proposed by MaxPool layer are provided as input to fully connected network. Then they predict object class using the Support Vector Machine (SVM) classification technique and draw bounding boxes around the classified objects. In a specific case of the retail website, the "regions" within the object "book" are classified and predicted using the SVM algorithm and data are extracted from these regions.

The scope of this study aims to understand and apply the deep learning based Yolo model to enable the web data extraction system to perform self-correction in the event of website layout changes. The internal workings of the subcomponents of the Yolo model are already explained in Ref. [26]; hence, an in-depth overview focusing on its application in web data extraction is provided. Once the Yolo is sufficiently trained with product images and the loss function is optimized, the proposed self-correction capability in the end-to-end web data extraction system is achieved.

- **Object detection:** Yolo is used for object or image detection. Twenty-four CNN layers and two fully connected network layers are used to extract features from images and then predict bounding box output coordinates. Yolo CNN, which can identify multiple text objects on a webpage image, is used. To identify text objects, Yolo DarkNet-19 architecture is considered. The DarkNet architecture is selected due to its lower processing requirement compared with other architectures, such as ImageNet and GoogleNet. The structure of DarkNet-19 and its application in the proposed web data extraction system is shown in Fig. 5.
- **Data extraction:** Once, the output image or product detail image is detected, Tesseract is used to extract the text. Tesseract is an optical character recognition open-source system that uses AI for text search and its image recognition; it also uses a two-step approach that initiates adaptive recognition. In turn, Tesseract uses LSTM for text recognition. The Tesseract architecture in the proposed web data extraction system is shown in Fig. 6. To predict text accurately from webpage images, a stepwise strategy is used to identify the text objects on the page using the Yolo model. Once the bounding boxes with text are detected, the

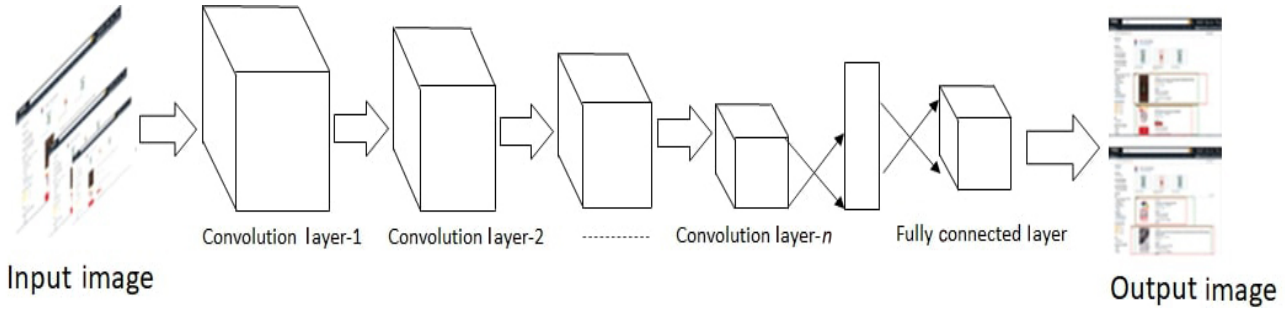
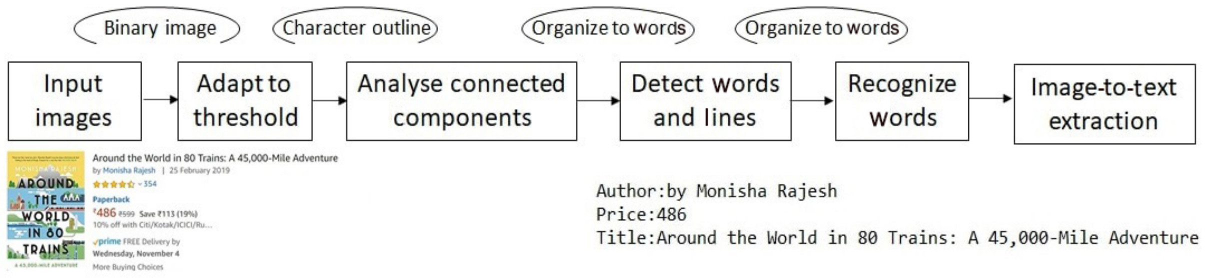


Fig. 5 Yolo architecture for object detection in the proposed web data extraction system.



```
Author:by Monisha Rajesh
Price:486
Title:Around the World in 80 Trains: A 45,000-Mile Adventure
```

Fig. 6 Tesseract LSTM architecture for image-to-text extraction in the proposed web data extraction system.

text is recognized. Several techniques can recognize text. A pretrained Tesseract neural nets LSTM engine, which is an OCR engine, is used. One advantage of the Tesseract engine is its support for many languages. Yolo detects the required text regions and crops them out from the image. Later, these regions are passed one by one to Tesseract, which reads them and returns the corresponding text. In the experiment, sixteen images are used to validate multiple cases. The pipeline predicts different numbers of text objects for retail vs. nonretail wesite, extracting the "book title", "author", and "price" from the products page in retail website and only the "book title" and "author" in the nonretail website. The performance of the pipeline is summarized in Table 3.

The end-to-end automated web data extraction system includes:

ProdDetail = product detail data extracted from output images,

Table 3 Development environment, tools, and technologies.

Tool	Detail
Programming language	Python 3.8
Object detection ML library	Yolo
Text extraction ML library	Tesseract (4.1.1)
Data storage	hdf5
GPU computing capacity	Yolo trained on Intel I7-10750H CPU@2.5 GHz
Installation and package management	Anaconda 4.9.0
Source code repository	GitHub

ObjDetect = object detection using the Yolo deep learning model, and

TxtExtract = image-to-text extraction using the Tesseract LSTM neural network model.

The input image sequence is set as $X = [x_1, x_2, \ldots, x_i]$, where

$x_i = (c, w, h, o, p)$,

c = bounding box center,

w, h = width and height of the bounding box,

o = class of object, and

p = probability of finding the object.

The output of convolutional layers in the Yolo model (*ObjDetect*) is pretrained on the ImageNet 1000 dataset with twenty-four convolutional network layers, followed by ROI pooling and fully connected network layers, using the DarkNet framework for training, inference, and detection. The final network layer predicts bounding box coordinates using an activation function and leaky rectified linear activation function for the remaining layers,

$$\phi(X) = \begin{cases} X, & \text{if } X \geqslant 0; \\ 0.1X, & \text{otherwise} \end{cases} \tag{1}$$

where X = probability of finding an object in the bounding box. After predicting the class probability, it uses a nonmax suppression algorithm to eliminate unnecessary anchor boxes and outputs bounding box detail for each class, as shown in Fig. 4. Text extraction (*TxtExtract*) is performed using the LSTM neural network model for post object identification. The input to LSTM neural networks is defined as $Y = [y_1, y_2, \ldots, y_j]$, where

$y_j = (s_d, t_s, f_t)$,

s_d = sample data point,

t_s = number of steps in single data,

f_t = number of variables for the corresponding true value in X.

3.2 Proposed algorithm

Given the input sequence $Y = (y_1, \ldots, y_i)$, the model is trained to maximize probability distribution $(P(l|Y))$ for the corresponding target and compute the output with the Tesseract algorithm, as shown in Algorithm 1.

3.3 Evaluation metrics

3.3.1 Bounding box output

The bounding box output in XML data format is represented with a few key tags shown (e.g., bounding box coordinates [xmin, ymin, xmax, ymax] and object properties) as follows:

- folder containing images;
- physical file name in the folder;
- image size in width, height, and depth; and
- object details with annotations, such as (1) object name, (2) truncated, indicating that the bounding box specified does not correspond to the full extent of the object, (3) difficult, means the object is difficult to recognize, and (4) bounding box coordinates.

3.3.2 Mean Average Precision error (mAP)

For all the experiments in Section 4, the mAP of a set of objects or images is used to measure the performance of models performing information retrieval and object detection, which is defined in the following:

$$\begin{aligned} \text{AP} &= \textstyle\sum_n (R_n - R_{n-1}) P_n, \\ \text{mAP} &= \frac{1}{n} \textstyle\sum_{i=1}^{n} \text{AP}_i \end{aligned} \tag{2}$$

where n = number of images in the dataset and AP = Average Precision (AP) for a given image. For a given image, AP is calculated and the mean of all the AP scores indicates the model's performance in identifying the images or objects. In Eq. (2), R_n and P_n are the precision and recall at the n-th threshold. mAP is the mean of AP over all the images or objects.

3.3.3 Data extraction accuracy

Data extraction accuracy is measured by precision and recall for image or object detection (e.g., product detail) and character extraction within the product detail or image, precision and recall are calculated in the following:

$$\begin{aligned} \text{Precision} &= (|R_{rl} \cap R_{rt}|)/R_{rl}, \\ \text{Recall} &= (|R_{rl} \cap R_{rt}|)/R_{rt} \end{aligned} \tag{3}$$

where R_{rl} = relevant images or objects and R_{rt} = retrieved images or objects.

3.3.4 Model performance

The third version of Yolo used in the experiment uses a binary cross-entropy loss function for each label, as shown in the following:

$$\text{Loss} = \begin{cases} -\log Obj_{predicted}, & \text{if } Obj_{actual} = 1; \\ -\log(1 - Obj_{predicted}), & \text{otherwise} \end{cases} \tag{4}$$

This reduces the computation complexity and predicts the objectness score for each bounding box by using logistic regression.

If validation loss and total loss converge, then the model is performing an accurate prediction task. Validatoin loss is a cost function value for cross-validation data, whereas total loss is a cost function value for training data.

Evaluation time is measured as an average of training time per input image over n images in the input dataset specified in the following:

$$\text{EvalTime} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{5}$$

where $x_i = [x_1, x_2, \ldots, x_n]$ is the input dataset.

Algorithm 1 Proposed web data extraction system algorithm

```
1:  Define evalProductData(image_names_list):
2:    classes ← load_class_names()
3:    tesseract_model ← init_tesseract_lstm_model()
4:    yolo_model ← init_yolov2_model()
5:    fpred_out = open_file("prediction_output.txt")
6:    prediction_time= { }
7:    FOR image_name in image_names_list:
8:      image ← load_image_array(image_name)
9:      predict_start_time = current_time
10:     predicted_bounding_boxes, predicted_classes = yolo_model.predict (image)
11: image_with_boxes_classes ← draw_boxes_classes(image)
12: save_image_to_disk(image_with_boxes_classes)
13: FOR index,box in enumerate( predicted_bounding_boxes) :
14:     predicted_location ← image.crop(box)
15:     predicted_class_name = classes[predicted_classes[$i$]]
16:     predicted_text ← tesseract.image_to_string(predicted_location)
17: prediction_time[image_name]= current_time - predict_start_time
18: plot_prediction_time(prediction_time)
```

4 Experimentation and Result

The experiment environment and input data modeling are explained in Section 4. Section 4.3 focuses on the results of seven real-world cases in the retail and nonretail domains to demonstrate the unique ability of the proposed system to identify the image of the product specification page by using Yolo, extract data from it by using Tesseract LSTM neural networks, and adapt to dynamic changes in the website layout. Section 5 provides deep insights into the limitation of the proposed web data extraction system and future research scope. The experimental results for 7-different real-world cases are presented in this section.

4.1 Experiment setup

Table 3 shows the tools, technologies, and hardware configuration used in the proposed web data extraction system. The case details in the proposed web data extraction system are shown in Table 4 to prove the system's automated extraction and self-correction capabilities.

4.2 Input data modeling and training

• **Input data modeling:** One of the key steps to building a successful deep learning based web data extraction system is the ability to define the input data model correctly. The remainder of the paper refers to specific product pages as "records" and product attributes as "regions". Each record is at a product category level,whereas each region is at a product detail level. Each record consists of multiple regions. Two parameters can influence automated web data extraction by using deep learning model (1) number of records to the input data model and (2) the loss incurred during the objection detection process. Increasing or decreasing the number of records or tuning the learning parameter in the loss function can remarkably improve the accuracy of the output data. The section below explains the input data setup process, which includes three steps: (1) data gathering, (2) data labeling, and (3) file creation.

– Gathering a high quality dataset: Image files are crucial to training deep learning systems, and considerable amount of time is spent to create 100+ image files across retail and nonretail websites, including single and multiple product specifications. The Yolo model is trained to detect single product (e.g., book) and multiple products (e.g., book and mobile) in the retail domain, and single product (e.g., book1) and multiple products (e.g., book1 and mobile1) in the nonretail domain.

– After data are gathered, bounding boxes around the objects are drawn. Eighty percent of the dataset is then labeled and associated with corresponding object classes and validation; testing phases use 10% of the dataset.

– The DarkNet expects the same dataset as that used in Section 2 for training and uses 80% of the total dataset for training and the remaining 20% for testing.

• **Training:** Training the model starts with pretraining the weights of the DarkNet and fine-tuning the model by changing the weights of the last two layers. This procedure allows maintaining the same feature extraction layers and retraining only the decision part, thus reducing training time. With the scope of research in consideration, the dataset is relatively smaller. The training, validation, and test sets are composed of 1000 images at a ratio of 80:10:10. The dataset is created by capturing images of product pages from different retail websites, and then creating bounding boxes and label for each image in the dataset. Yolo identifies the top K bounding boxes per image based on the intersection over union measure and the confidence score of the bounding box for each text object class. In this research scope, three different classes of text objects (book title, author, and retail price) for identification are chosen. The model is trained for 30 epochs, and the learning rate, weight decay, and momentum values are similar to the values used for training Yolo DarkNet-19 on the Common Objects in COntext (COCO) and Visual Object Challenge (VOC) datasets. Pascal VOC is an XML file, where one file is created for each image in the dataset, whereas COCO is in the form of a JSON file, which has one file for the entire dataset and the testing and validation datasets. The bounding box in Pascal VOC is represented as (x – top left, y–top left, width, height), and the COCO bounding box is represented as (xmin – top left, ymin – top left, xmax – bottom right, ymax – bottom right).

Table 4 Experiment structure.

Experiment	Product	Domain	Error	Self-correction
1	Single	Retail	No	No
2	Single	Retail	Yes	Yes
3	Multiple	Retail	No	No
4	Multiple	Retail	Yes	Yes
5	Single	Nonretail	No	No
6	Single	Nonretail	Yes	Yes
7	Multiple	Nonretail	No	No

4.3 Results

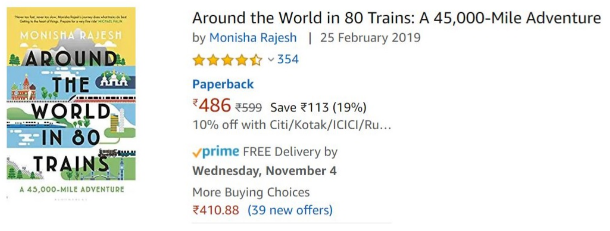
For each experiment the following figures are included:

• Input image with and without user login indicating an error/ no error condition;

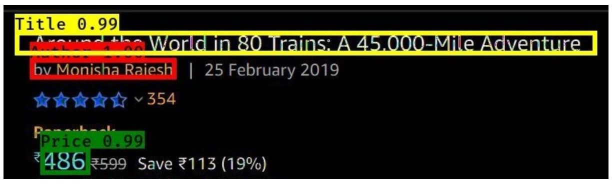
- Output image with a bounding box user login indicating an error/no error condition, and demonstrating the automated image or object detection capability by using the deep learning based Yolo model;
- Output text demonstrating the work of the proposed web data extraction algorithm.

4.3.1 Experiments 1 and 2: Extracting data from single product specification in the Amazon retail site without and with changes in website layout

This experiment uses the proposed web data extraction framework to extract data from one product page (e.g., "book") in the Amazon retail website, without logging into the Amazon website, which represents the normal condition. The next step is to log into the Amazon retail website as a registered user and search for the same book that shows a different URL and layout of the product page. The change in product page layout and associated URL is an indication of the change in website layout and hence there is an error condition. The input data are an image for a single product specification from which data are extracted without login (or as a guest user), and the corresponding output image with a bounding box around single product detail is shown in Fig. 7. Similarly, input and output images with bounding boxes around single product detail and with error conditions (guest user) are shown in Fig. 8. Data extracted from single product specification pages (e.g., product: "book" and product details: "author name" and "price") in the Amazon retail website are shown in Figs. 9 and 10.


Around the World in 80 Trains: A 45,000-Mile Adventure
by Monisha Rajesh | 25 February 2019
354
Paperback
₹486 ~~₹599~~ Save ₹113 (19%)
10% off with Citi/Kotak/ICICI/Ru...
prime FREE Delivery by
Wednesday, November 4
More Buying Choices
₹410.88 (39 new offers)

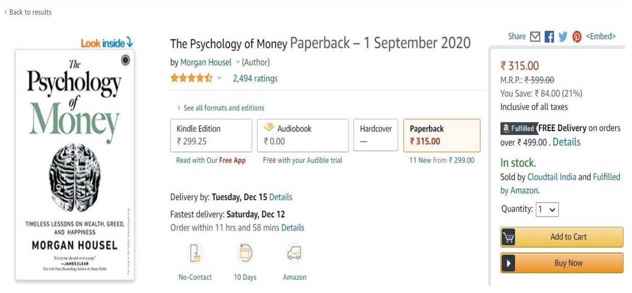
(a) Input: Single product page without changes in website layout or location (URL)



Around the World in 80 Trains: A 45,000-Mile Adventure
by Monisha Rajesh | 25 February 2019
354
₹486 ₹599 Save ₹113 (19%)

(b) Output: Single product page with bounding boxes around product detail (e.g., book) without changes in website layout or location (URL)

Fig. 7 Object detection with bounding boxes around single product detail without changes in the website layout or location (URL) of the product page.

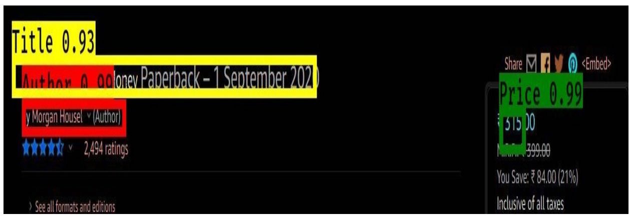

‹ Back to results
The Psychology of Money Paperback – 1 September 2020
by Morgan Housel (Author)
2,494 ratings
› See all formats and editions

Kindle Edition	Audiobook	Hardcover	Paperback
₹ 299.25	₹ 0.00	—	₹ 315.00
Read with Our Free App	Free with your Audible trial		11 New from ₹ 299.00

Delivery by: Tuesday, Dec 15 Details
Fastest delivery: Saturday, Dec 12
Order within 11 hrs and 58 mins Details

₹ 315.00
M.R.P.: ~~₹ 399.00~~
You Save: ₹ 84.00 (21%)
Inclusive of all taxes
Fulfilled FREE Delivery on orders over ₹ 499.00 . Details
In stock.
Sold by Cloudtail India and Fulfilled by Amazon.
Quantity: 1

(a) Input: Single product page with changes in website layout or location (URL)



loney Paperback – 1 September 2020
y Morgan Housel (Author)
2,494 ratings
315.00
399.00
You Save: ₹ 84.00 (21%)
Inclusive of all taxes
› See all formats and editions

(b) Output: Single product page with bounding boxes around product detail (e.g., book) with changes in website layout or location (URL)

Fig. 8 Object detection with bounding boxes around single product detail with changes in the website layout or location (URL) of the product page.

```
-<object>                -<object>              <name>Title</name>
<name>Author</name>      <name>Price</name>     -<bndbox>
-<bndbox>                -<bndbox>              <xmin>60</xmin>
<xmin>94</xmin>          <xmin>206</xmin>       <ymin>370</ymin>
<ymin>389</ymin>         <ymin>402</ymin>       <xmax>90</xmax>
<xmax>120</xmax>         <xmax>239</xmax>       <ymax>1099</ymax>
<ymax>572</ymax>         <ymax>464</ymax>       </bndbox>
</bndbox>                -<object>              </object>
</object>
```

(a) Output: Coordinates of the bounding box around product detail without changes in the website layout or location (URL) of the product page

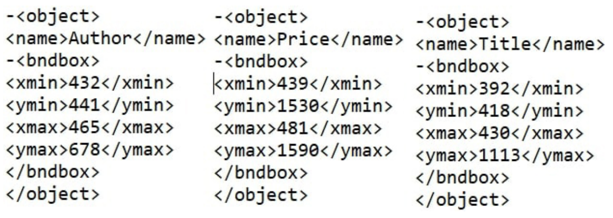
```
Author:by Monisha Rajesh
Price:486
Title:Around the World in 80 Trains: A 45,000-Mile Adventure
```

(b) Output: Product detail extracted from the single product page (e.g., book) of the Amazon retail website

Fig. 9 Object detection with bounding boxes around single product detail and data extracted without changes in the website layout or location (URL) of the product page.

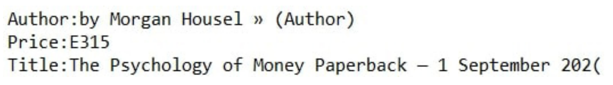
4.3.2 Experiments 3 and 4: Extracting data from multiple product specifications in the Amazon retail site without and with changes in website layout

The input data as an image for multiple product specifications from which data need to be extracted without error conditions(no login) and the corresponding output image with a bounding box, as shown in Fig. 11. Similarly, input and output images with bounding boxes around product detail with error condition (as a guest user) are shown in Fig. 12. Data extracted from the product specification page (e.g., product: "book" and product detail: "author name" and "price") in



```
-<object>                -<object>                -<object>
<name>Author</name>      <name>Price</name>       <name>Title</name>
-<bndbox>                -<bndbox>                -<bndbox>
<xmin>432</xmin>         <xmin>439</xmin>         <xmin>392</xmin>
<ymin>441</ymin>         <ymin>1530</ymin>        <ymin>418</ymin>
<xmax>465</xmax>         <xmax>481</xmax>         <xmax>430</xmax>
<ymax>678</ymax>         <ymax>1590</ymax>        <ymax>1113</ymax>
</bndbox>                </bndbox>                </bndbox>
</object>                </object>                </object>
```

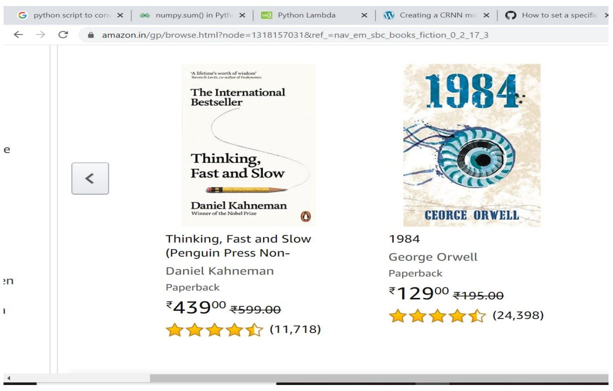
(a) Output: Coordinates of the bounding box around single product detail with changes in the website layout or location (URL) of the product page



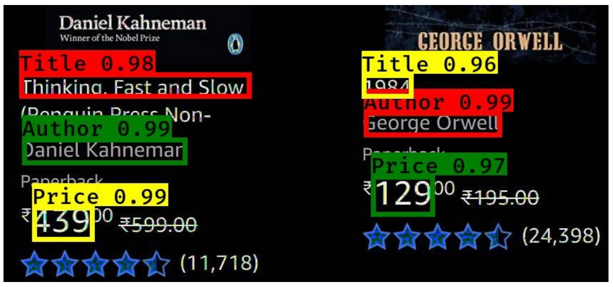
```
Author:by Morgan Housel » (Author)
Price:E315
Title:The Psychology of Money Paperback – 1 September 202(
```

(b) Output: Product detail extracted from the single product page (e.g., book) in the Amazon retail website

Fig. 10 Object detection with bounding boxes around single product detail and data extracted with changes in the website layout or Location (URL) of the product page.


amazon.in/gp/browse.html?node=1318157031&ref_=nav_em_sbc_books_fiction_0_2_17_3
Thinking, Fast and Slow (Penguin Press Non-
Daniel Kahneman
Paperback
₹439⁰⁰ ₹599.00
(11,718)
1984
George Orwell
Paperback
₹129⁰⁰ ₹195.00
(24,398)

(a) Input: Multiple products with no changes in website layout location (URL)



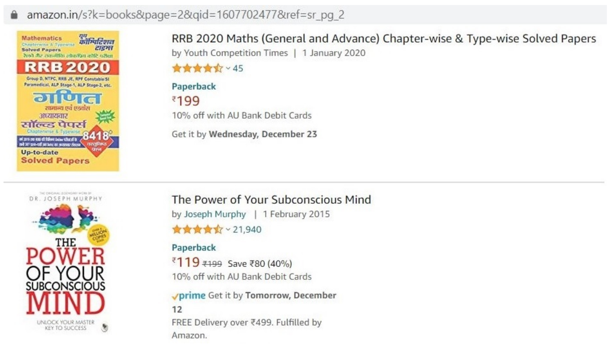
(b) Output: Multiple products with bounding boxes around product detail (e.g., book) without changes in website layout or location (URL)

Fig. 11 Object detection with bounding boxes around multiple product detail without changes in the website layout or location (URL) of the product page.

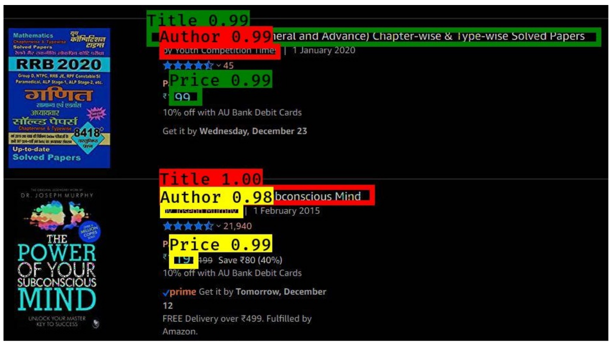
the Amazon website by using Tesseract LSTM neural networks are shown in Figs. 13 and 14.

4.3.3 Experiments 5 and 6: Extracting data from single product specification in nonretail site without and with changes in website layout

This experiment uses the proposed web data extraction framework to extract data from one product page (e.g.,

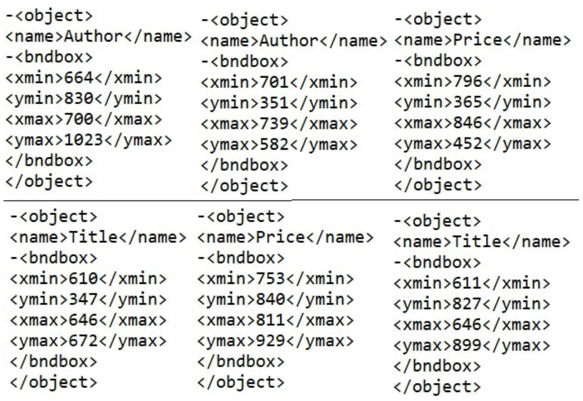

amazon.in/s?k=books&page=2&qid=1607702477&ref=sr_pg_2
RRB 2020 Maths (General and Advance) Chapter-wise & Type-wise Solved Papers
by Youth Competition Times | 1 January 2020
Paperback
₹199
10% off with AU Bank Debit Cards
Get it by Wednesday, December 23
The Power of Your Subconscious Mind
by Joseph Murphy | 1 February 2015
Paperback
₹119 ₹199 Save ₹80 (40%)
10% off with AU Bank Debit Cards
prime Get it by Tomorrow, December 12
FREE Delivery over ₹499. Fulfilled by Amazon.

(a) Input: Multiple products with changes in website layout or location (URL)



(b) Output: Multiple products with bounding boxes around product detail (e.g., book) with change in website layout or location (URL)

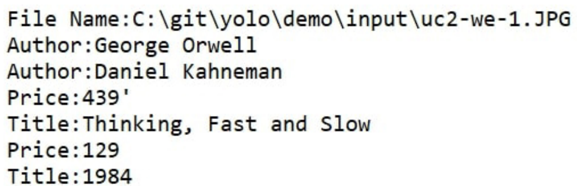
Fig. 12 Object detection with bounding boxes around multiple product detail with changes in website layout or location (URL) of the product page.



```
-<object>                -<object>                -<object>
<name>Author</name>      <name>Author</name>      <name>Price</name>
-<bndbox>                -<bndbox>                -<bndbox>
<xmin>664</xmin>         <xmin>701</xmin>         <xmin>796</xmin>
<ymin>830</ymin>         <ymin>351</ymin>         <ymin>365</ymin>
<xmax>700</xmax>         <xmax>739</xmax>         <xmax>846</xmax>
<ymax>1023</ymax>        <ymax>582</ymax>         <ymax>452</ymax>
</bndbox>                </bndbox>                </bndbox>
</object>                </object>                </object>

-<object>                -<object>                -<object>
<name>Title</name>       <name>Price</name>       <name>Title</name>
-<bndbox>                -<bndbox>                -<bndbox>
<xmin>610</xmin>         <xmin>753</xmin>         <xmin>611</xmin>
<ymin>347</ymin>         <ymin>840</ymin>         <ymin>827</ymin>
<xmax>646</xmax>         <xmax>811</xmax>         <xmax>646</xmax>
<ymax>672</ymax>         <ymax>929</ymax>         <ymax>899</ymax>
</bndbox>                </bndbox>                </bndbox>
</object>                </object>                </object>
```

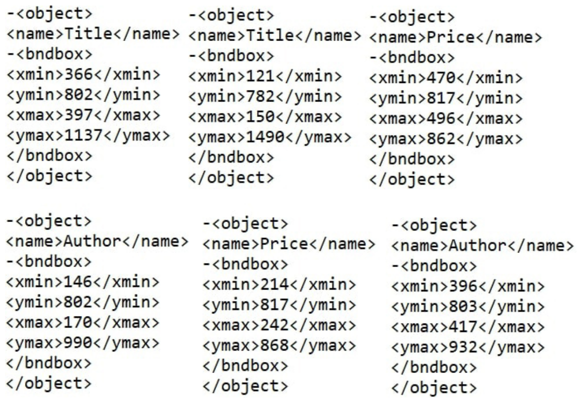
(a) Output: Coordinates of the bounding box around multiple product detail without changes in website layout or location (URL) of the product page



```
File Name:C:\git\yolo\demo\input\uc2-we-1.JPG
Author:George Orwell
Author:Daniel Kahneman
Price:439'
Title:Thinking, Fast and Slow
Price:129
Title:1984
```

(b) Output: Product detail extracted from multiple products (e.g., two books) in the Amazon retail website

Fig. 13 Object detection with bounding boxes around multiple product detail and data extracted without changes in website layout or location (URL) of the product page.



```
-<object>
<name>Title</name>
-<bndbox>
<xmin>366</xmin>
<ymin>802</ymin>
<xmax>397</xmax>
<ymax>1137</ymax>
</bndbox>
</object>
```

```
-<object>
<name>Title</name>
-<bndbox>
<xmin>121</xmin>
<ymin>782</ymin>
<xmax>150</xmax>
<ymax>1490</ymax>
</bndbox>
</object>
```

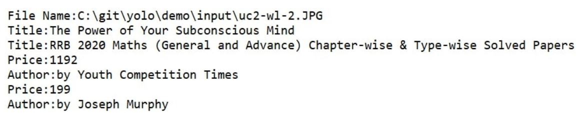
```
-<object>
<name>Price</name>
-<bndbox>
<xmin>470</xmin>
<ymin>817</ymin>
<xmax>496</xmax>
<ymax>862</ymax>
</bndbox>
</object>
```

```
-<object>
<name>Author</name>
-<bndbox>
<xmin>146</xmin>
<ymin>802</ymin>
<xmax>170</xmax>
<ymax>990</ymax>
</bndbox>
</object>
```

```
-<object>
<name>Price</name>
-<bndbox>
<xmin>214</xmin>
<ymin>817</ymin>
<xmax>242</xmax>
<ymax>868</ymax>
</bndbox>
</object>
```

```
-<object>
<name>Author</name>
-<bndbox>
<xmin>396</xmin>
<ymin>803</ymin>
<xmax>417</xmax>
<ymax>932</ymax>
</bndbox>
</object>
```

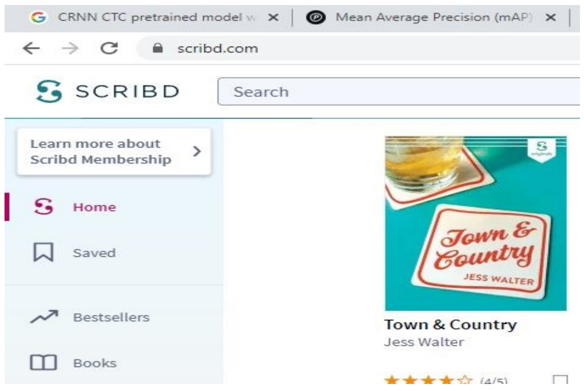
(a) Output: Coordinates of bounding box around product detail with changes in website layout or location (URL) of the product page



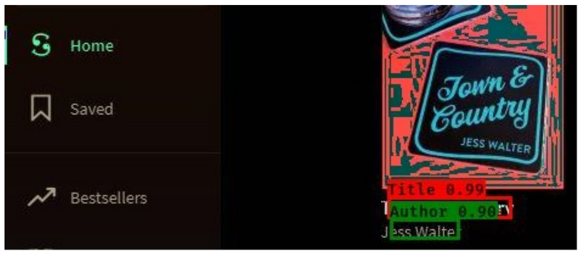
```
File Name:C:\git\yolo\demo\input\uc2-wl-2.JPG
Title:The Power of Your Subconscious Mind
Title:RRB 2020 Maths (General and Advance) Chapter-wise & Type-wise Solved Papers
Price:1192
Author:by Youth Competition Times
Price:199
Author:by Joseph Murphy
```

(b) Output: Product detail extracted from single product page (e.g., book) in Amazon retail website

Fig. 14 Object detection with bounding boxes around multiple product detail and data extracted with changes in website layout or location (URL) of the product page.



(a) Input: Single product page in nonretail website without changes in website layout or location (URL)



(b) Output: Single product page in a nonretail website with bounding boxes around product detail (e.g., book) without changes in website layout or location (URL)

Fig. 15 Object detection with bounding boxes around single product detail in a nonretail website without changes in the website layout or location (URL) of the product page.

book) in the nonretail website without logging in to the website, which is the normal condition. The next step is to log into the nonretail website as a registered user and search for the same book that shows a different URL and layout of the product page. The change in product page layout and associated URL is an indication of the change in website layout and hence an error condition. The input data as an image for a single product specification from which data need to be extracted without error conditions (no login) and the corresponding output image with a bounding box is shown in Fig. 15. Similarly, input and output images with bounding boxes around product detail with error conditions (as guest user) are shown in Fig. 16. Data extracted from product specification page (e.g., product: "book" and product detail: "author name" and "price") of the Amazon website using Tesseract LSTM neural networks are shown in Figs. 17 and 18.

4.3.4 Experiment 7: Extracting data from multiple product specification in nonretail site without changes in website layout

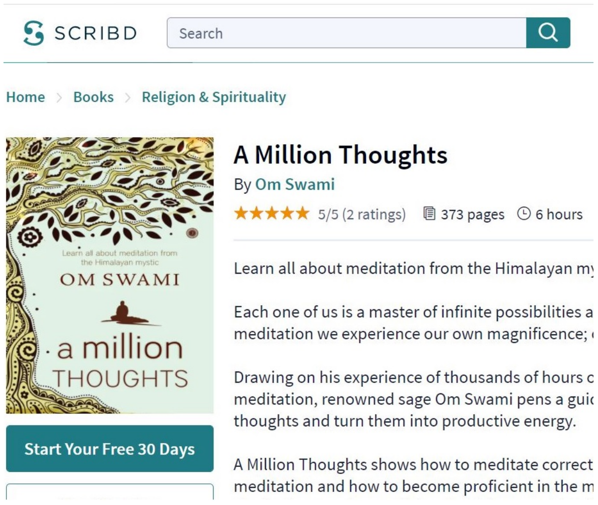
This experiment uses the proposed web data extraction framework to extract data from multiple products (e.g., two book) in the Amazon retail website, without user login to the nonretail website. The input data for multiple product specifications from which data are extracted without login (or as a guest user); the corresponding output image with a bounding box around product detail is shown in Fig. 19. Data extracted from the multiple product specifications page (e.g., product: "book" and product detail: "author name" and "price") of a nonretail domain are shown in Fig. 20.

5 Critical Review, Limitation, and Future Scope

The performance parameters outlined below demonstrate the intelligent extraction and self-correction capability of the proposed web data extraction system, that is, (1) data correctness; (2) data or character extraction accuracy; (3) image detection accuracy; (4) no code, no rework for data extraction, resulting in developer productivity; and (5) adaptability to dynamic changes in websites due to self-correction (enabling automated extraction). Two key parameters constitute the performance of the proposed system: (1) data quality and (2) performance.

5.1 Data quality and performance

In object detection technology, mAP is used as an evaluation metric to measure the model performance and data quality of web data extraction systems. First, data are assessed whether they have been extracted


SCRIBD Search

Home > Books > Religion & Spirituality

A Million Thoughts
By Om Swami
★★★★★ 5/5 (2 ratings) 373 pages 6 hours

Learn all about meditation from the Himalayan my

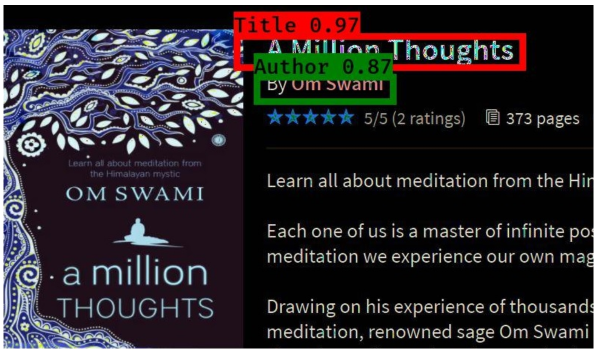
Each one of us is a master of infinite possibilities a
meditation we experience our own magnificence;

Drawing on his experience of thousands of hours c
meditation, renowned sage Om Swami pens a guic
thoughts and turn them into productive energy.

A Million Thoughts shows how to meditate correct
meditation and how to become proficient in the m

Start Your Free 30 Days

(a) Input: Single products in a nonretail website with changes in website layout or location (URL)


Title 0.97
A Million Thoughts
Author 0.87
By Om Swami
★★★★★ 5/5 (2 ratings) 373 pages
Learn all about meditation from the Hir
Each one of us is a master of infinite pos
meditation we experience our own mag
Drawing on his experience of thousands
meditation, renowned sage Om Swami
Learn all about meditation from the Himalayan mystic
OM SWAMI
a million
THOUGHTS

(b) Output: Single product page in a nonretail website with bounding boxes around product detail (e.g., book) with changes in website layout or location (URL)

Fig. 16 Object detection with bounding boxes around single product detail in a nonretail website with changes in the website layout or location (URL) of the product page.

```
-<object>                -<object>
<name>Title</name>       <name>Author</name>
-<bndbox>                -<bndbox>
<xmin>353</xmin>         <xmin>371</xmin>
<ymin>317</ymin>         <ymin>319</ymin>
<xmax>371</xmax>         <xmax>388</xmax>
<ymax>419</ymax>         <ymax>376</ymax>
</bndbox>                </bndbox>
</object>                </object>
```

(a) Output: Coordinates of the bounding box around the product detail without changes in the website layout or location (URL) of the product page in a nonretail website

```
Title:own & Country
Author:ass Walter
```

(b) Output: Single product page in a nonretail website with bounding boxes around product detail (e.g., book) without changes in website layout or location (URL)

Fig. 17 Object detection with bounding boxes around product detail in a nonretail website without changes in the website layout or location (URL) of the product page.

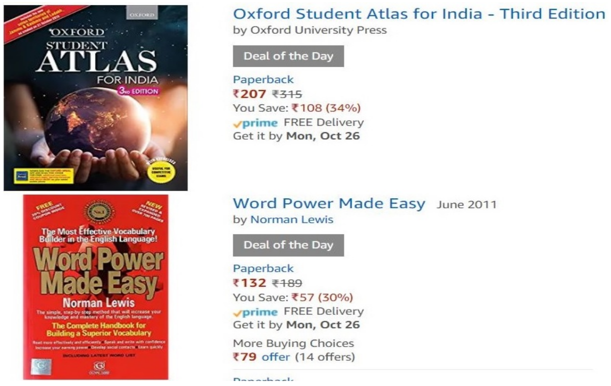
```
-<object>                -<object>
<name>Title</name>       <name>Author</name>
-<bndbox>                -<bndbox>
<xmin>307</xmin>         <xmin>362</xmin>
<ymin>728</ymin>         <ymin>755</ymin>
<xmax>357</xmax>         <xmax>403</xmax>
<ymax>1122</ymax>        <ymax>947</ymax>
</bndbox>                </bndbox>
</object>                </object>
```

(a) Output: Coordinates of the bounding box around product detail with changes in the website layout or location (URL) of the product page of a nonretail website

```
Title:z AMillion Thoughts
Author:By Om Swami
```

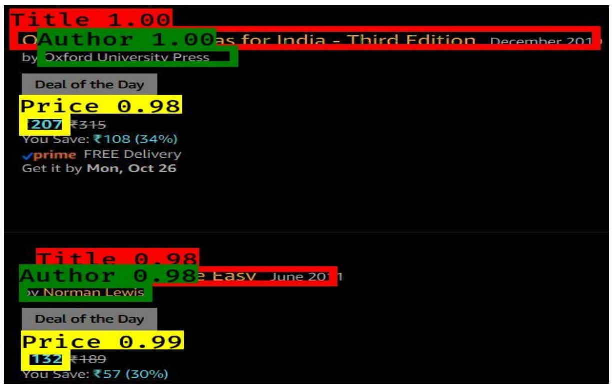
(b) Output:Product detail extracted from the single product page (e.g., book) of a nonretail website

Fig. 18 Object detection with bounding boxes around product detail and data extracted from a nonretail website with changes in the website layout or location (URL) of the product page.


Oxford Student Atlas for India - Third Edition
by Oxford University Press
Deal of the Day
Paperback
₹207 ~~₹315~~
You Save: ₹108 (34%)
✓prime FREE Delivery
Get it by Mon, Oct 26

Word Power Made Easy June 2011
by Norman Lewis
Deal of the Day
Paperback
₹132 ~~₹189~~
You Save: ₹57 (30%)
✓prime FREE Delivery
Get it by Mon, Oct 26
More Buying Choices
₹79 offer (14 offers)

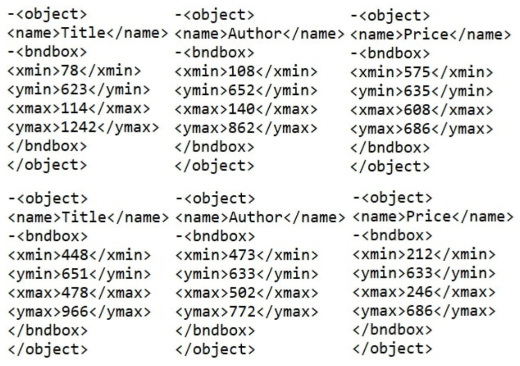
(a) Input: Multiple product page in a nonretail website without changes in website layout or location (URL)



(b) Output: Bounding boxes around multiple product detail (e.g., two book) without changes in the website layout or location (URL) of the product page

Fig. 19 Object detection with bounding boxes around multiple product detail in a nonretail website without changes in the website layout or location (URL) of the product page.

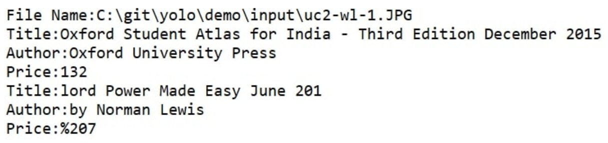
or measured by mAP. Second, the extracted data are assessed whether they are correct (also called data accuracy or data correctness). Experiments 1–7 have established that the extracted data are correct as per product detail, with a focus on measuring whether all the data records in a product specification have been extracted, as shown in Table 5.



```
-<object>          -<object>          -<object>
<name>Title</name> <name>Author</name> <name>Price</name>
-<bndbox>          -<bndbox>          -<bndbox>
<xmin>78</xmin>    <xmin>108</xmin>   <xmin>575</xmin>
<ymin>623</ymin>   <ymin>652</ymin>   <ymin>635</ymin>
<xmax>114</xmax>   <xmax>140</xmax>   <xmax>608</xmax>
<ymax>1242</ymax>  <ymax>862</ymax>   <ymax>686</ymax>
</bndbox>          </bndbox>          </bndbox>
</object>          </object>          </object>

-<object>          -<object>          -<object>
<name>Title</name> <name>Author</name> <name>Price</name>
-<bndbox>          -<bndbox>          -<bndbox>
<xmin>448</xmin>   <xmin>473</xmin>   <xmin>212</xmin>
<ymin>651</ymin>   <ymin>633</ymin>   <ymin>633</ymin>
<xmax>478</xmax>   <xmax>502</xmax>   <xmax>246</xmax>
<ymax>966</ymax>   <ymax>772</ymax>   <ymax>686</ymax>
</bndbox>          </bndbox>          </bndbox>
</object>          </object>          </object>
```

(a) Output: Coordinates of the bounding box around multiple product detail with changes in the website layout or location (URL) of the product page of a nonretail website



```
File Name:C:\git\yolo\demo\input\uc2-wl-1.JPG
Title:Oxford Student Atlas for India - Third Edition December 2015
Author:Oxford University Press
Price:132
Title:lord Power Made Easy June 201
Author:by Norman Lewis
Price:%207
```

(b) Output: Product detail extracted from multiple products (e.g., two book) of a nonretail website

Fig. 20 Object detection with bounding boxes around multiple product detail and data extracted from a nonretail website with changes in the website layout or location (URL) of the product page.

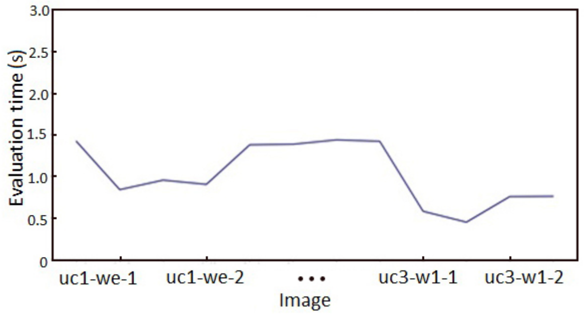
Table 5 Performance metrics of the proposed web data extraction system.

Parameter	Object/character
mAP	74%
Object extraction accuracy (precision and recall)	97% and 48.8%
Character extraction accuracy (precision and recall)	99% and 49.5%
Evaluation time	1.02 s (Avg.)
Total loss	6% (Avg.)
Validation loss	6% (Avg.)

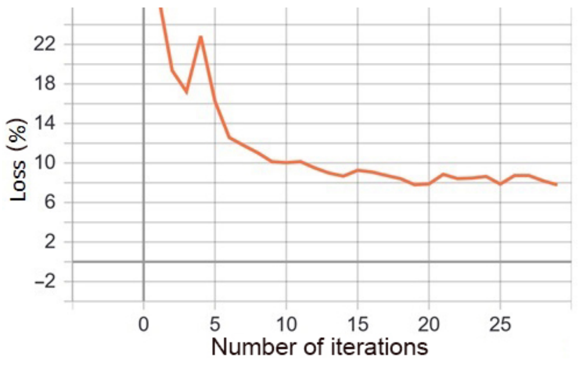
Through experimentation with the seven cases, 18 "author" records, 12 "price" records, and 15 "title" records were successfully extracted. At the record level, the total number of objects to be detected was 45, of which 44 were extracted and only 1 object was not detected, with a precision of 97% and recall of 48.8%. At the character level, 990 characters out of 1000 characters were detected and 10 characters were not detected or detected wrongly, with a precision of 99% and recall of 49.5%. The mAP was 74%.

The performance of the proposed system, such as evaluation time, is shown in Fig. 21a, that includes execution time from initiation to completion process of the end-to-end extraction. Figures 21b and 22 show overall loss and validation loss for all the experiments conducted as per Eq. (5). Figure 21a shows the time consumed by the pipeline when extrating text data from the webpage, including the time consumed when saving images with bounding boxes and extracting text in the disk or output file.

Yolo and LSTM model execution time is higher in the initial run, but decreases in subsequent cycles, thus reducing the overall execution time. The model training time is a function of the system GPU because it involves graphic image processing, which is computation intensive. The execution time starts from the initiation of the experiment (i.e., locating the product specification page) up to the extraction of data, as shown in Fig. 21a. The evaluation time is high in the first iteration



(a) Evaluation time of the proposed model (Avg.: 1.02 s)



(b) Total loss of the proposed model

Fig. 21 Evaluation time and total loss of the proposed web data extraction system.

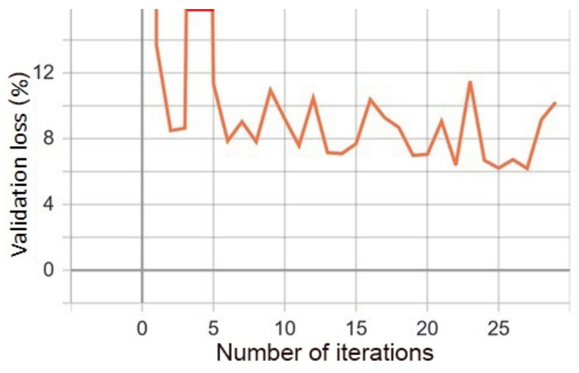


Fig. 22 Validation loss of the proposed web data extraction model.

due to the initial model training time, but decreases in subsequent iterations.

Evaluation time, total loss, and validation loss exhibit a marginal difference for the experiments involving data extraction from single and multiple product specifications for retail and nonretail websites. This result reinforces the proposed system's scaling capability to extract data from multiple product specifications without much performance degradation. Therefore, the proposed system can perform self-correction in the event of website layout changes.

5.2 Limitations and future scope

Many articles on web data extraction using heuristics techniques, traditional statistical models, and machine learning algorithms are available. Research on web data extraction includes webpages with unstructured, semistructured, and structured data, in addition to data rendering with static HTML or dynamic rendering using JavaScript. The following presents some challenges faced by the research community in automated web data extraction and the future scope of research.

5.2.1 Deep learning based web data extraction systems

• Deep learning networks using image-based recognition techniques are transforming the auto navigation and extraction process in the end-to-end web data extraction systems; however, such transformation requires extensive training of the Yolo and Tesseract LSTM neural network models for a precise object or image detection and image-to-text extraction[1–3].

• Deep learning based automated web data extraction results in 99% data accuracy (precision), as demonstrated with seven real-world cases, including the detection and extraction of 18 "author" records, 12 "price" records, and 15 "title" records, the detection of 44 out of 45 objects, and the extraction of 990 out of 1000 characters.

• Model loss and evaluation time data reveal no significant degradation of the proposed web data extraction system by using Yolo and LSTM models, although further experimentation with additional datasets is recommended to prove the scalability, adaptability, and domain agnostic capability of the proposed system. However, extracting multiple product information across web pages requires numerous images to be captured and trained, thus increasing the model training time and potentially slowing down the execution time. Optimizing the model training and execution time will be a future scope of research.

• Deep learning based automated and adaptive web data extraction systems are progressing rapidly[3, 8, 41] with high data accuracy rates (precision and recall 100%)[6]; however, the adaptation of such systems to dynamic changes in website layout still remains a challenge. The proactive detection of faults in websites using anomaly detection techniques[42, 43] for improved self-correction capability is another area of future research on web data extraction.

• Deep learning based object detection techniques are highly computation intensive, thus requiring numerous GPUs because of the need to preprocess and train several web pages as images. Web data extraction systems handling data extraction from numerous websites and nested web pages will limit the ability of the proposed web data extraction system to fetch data with optimal performance. These systems will have to rely on cloud-based computing systems, such as Google Colaboratory, for data extraction efficiency.

• Rotating bounding boxes as a technique is not considered due to its implementation complexity, but in a real-world scenario where the text is rotated, the proposed approach will not work.

• Text extraction OCR cannot recognize the rupee symbol in the book "price". Additionally, the price listed on the Amazon retail website has the decimal part in superscript form and hence not recognized. Nevertheless, this problem can be handled by retraining Tesseract to recognize such text elements.

5.2.2 Machine learning based web data extraction systems

• Scheduling the extraction of the same data multiple times causes errors, due to the inability to find the interval for extraction. It assumes websites to expose JSON data in DOM. Future work is required to integrate DOM extraction techniques to develop a general purpose web data extraction engine.

• In recent years, research on automatic extraction without manual labeling has been conducted. Many of these automatic extraction systems are less accurate because users need to label the training pages. Additionally, a manual post-processing step is involved, that is, users must identify what he/she is interested in.

• Data manipulation, extensibility, and execution are key considerations when extracting data from restricted browser environments. JavaScript has become an essential part of the majority of webpage development and the cornerstone of rich Internet applications.

Browser extension to improve browsing experience has been another recent development that has posed challenges to extracting data, examples include pop-up data.

• Commonly used position or structure-based web scraping tools must be manually reconfigured as soon as the structure of the webpage changes. Wrappers are specific to a given website and are tightly linked to the markup and structure of provider pages.

• With the development of web applications using cascading style sheets and JavaScript enabling dynamic data rendering, modern-day approaches use visualization-based techniques for data extraction. However, such approaches are highly computation intensive, and it slows down the extraction process (e.g., deep learning).

5.2.3 Adaptive web data extraction systems

• Contemporary and machine learning based extraction systems usually rely on extraction rules or wrappers tailored to a particular data source.

• Most automatic data extraction systems can only cope with a limited set of document formats and do not adapt well to changes in the document structure, HTML documents containing interesting data must be located.

• Data of interest must be located within the web page, and rules that can be used to extract data must be created.

• A mechanism used to create data extraction rules must either be sufficiently general or be easy to implement, so that data can be extracted from the wide variety of page formats available on the web.

• Data extraction system must be able to cope with changes to webpage structure, because web data providers frequently change the configuration and data of their pages.

6 Conclusion

This study proposes an adaptive and intelligent automated web data extraction system that uses YOLO and Tesseract LSTM neural networks, which not only adapts to dynamic changes in website layout, but also automatically extracts data. The proposed web data extraction system eliminates the extraction engine subcomponents, which are core to traditional and machine learning techniques for efficient data extraction; therefore, the system has the potential to transforming the process of automated data extraction from websites in the future. This study completes experiments with real-world examples from the retail and nonretail domains to extract product detail from single and multiple web pages, and demonstrates intelligence and adaptability. Future research will be driven by advancements in deep learning networks to reimagine the end-to-end automated web data extraction process.

References

[1] Y. B. Zhang, Image feature extraction algorithm in big data environment, *Journal of Intelligent and Fuzzy Systems*, vol. 39, no. 4, pp. 5109–5118, 2020.

[2] L. Xie, J. L. Tao, Q. N. Zhang, and H. Y. Zhou, CNN and KPCA-based automated feature extraction for real time driving pattern recognition, *IEEE Access*, vol. 7, pp. 123765–123775, 2019.

[3] J. Tao, H. B. Wang, X. Y. Zhang, X. Y. Li, and H. W. Yang, An object detection system based on YOLO in traffic scene, in *Proc. of 2017 6th Int. Conf. Computer Science and Network Technology* (*ICCSNT*), Dalian, China, 2017, pp. 315–319.

[4] F. Ali, A. Ali, M. Imran, R. A. Naqvi, M. H. Siddiqi, and K. S. Kwak, Traffic accident detection and condition analysis based on social networking data, *Accident Analysis & Prevention*, vol. 151, p. 105973, 2021.

[5] N. Islam, Z. Islam, and N. Noor, A survey on optical character recognition system, *Journal of Information & Communication Technology-JICT*, vol. 10, no. 2, pp. 1–4, 2016.

[6] H. Rao and D. R. M. Sashikumar, A survey on automated web data extraction techniques for product specification from e-commerce web sites, *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 6, no. 8, pp. 310–316, 2016.

[7] E. Uzun, A novel web scraping approach using the additional information obtained from web pages, *IEEE Access*, vol. 8, pp. 61726–61740, 2020.

[8] M. Salah, B. Al Okush, and M. Al Rifaee, A comparison of web data extraction techniques, in *Proc. of 2019 IEEE Jordan Int. Joint Conf. Electrical Engineering and Information Technology* (*JEEIT*), Amman, Jordan, 2019, pp. 785–789.

[9] S. L. Li, C. Chen, K. W. Luo, and B. Song, Review of deep web data extraction, in *Proc. of 2019 IEEE Symp. Series on Computational Intelligence* (*SSCI*), Xiamen, China, 2019, pp. 1068–1070.

[10] W. Nadee and K. Prutsachainimmit, Towards data extraction of dynamic content from JavaScript web applications, in *Proc. of 2018 Int. Conf. Information Networking* (*ICOIN*), Chiang Mai, Thailand, 2018, pp. 750–754.

[11] B. V. S. Ujwal, B. Gaind, A. Kundu, A. Holla, and M. Rungta, Classification-based adaptive web scraper, in *Proc. of 16th IEEE Int. Conf. Machine Learning and Applications*, Cancun, Mexico, 2017, pp. 125–132.

[12] J. Park and D. Barbosa, Adaptive record extraction from web pages, in *Proc. of WWW 2007*, Banff, Canada, 2007,

pp. 1335–1336.

[13] C. J. Liu, Y. F. Tao, J. W. Liang, K. Li, and Y. H. Chen, Object detection based on YOLO network, in *Proc. of 2018 IEEE 4th Information Technology and Mechatronics Engineering Conf.* (*ITOEC*), Chongqing, China, 2018, pp. 799–803.

[14] J. L. Hong, Deep web data extraction, in *Proc. of 2010 IEEE Int. Conf. Systems, Man and Cybernetics*, Istanbul, Turkey, 2010, pp. 3420–3427.

[15] F. Ali, P. Khan, K. Riaz, D. Kwak, T. Abuhmed, D. Park, and K. S. Kwak, A fuzzy ontology and SVM-based web content classification system, *IEEE Access*, vol. 5, pp. 25781–25797, 2017.

[16] W. Li, W. Shao, S. X. Ji, and E. Cambria, BiERU: Bidirectional emotional recurrent unit for conversational sentiment analysis, arXiv preprint arXiv: 2006.00492, 2021.

[17] K. M. He, G. Gkioxari, P. Dollár, and R. Girshick, Mask R-CNN, in *Proc. of 2017 IEEE Int. Conf. Computer Vision* (*ICCV*), Venice, Italy, 2017, pp. 2980–2988.

[18] S. Nagarajan and K. Perumal, A deep neural network for information extraction from web pages, in *Proc. of 2017 IEEE Int. Conf. Power, Control, Signals and Instrumentation Engineering* (*ICPCSI*), Chennai, India, 2017, pp. 918–922.

[19] T. Gogar, O. Hubacek, and J. Sedivy, Deep neural networks for web page information extraction, in *Artificial Intelligence Applications and Innovations. IFIP Advances in Information and Communication Technology, vol. 475*, L. Iliadis and I. Maglogiannis, eds. Thessaloniki, Greece: Springer, 2016, pp. 154–163.

[20] R. Baumgartner, M. Ceresna, and G. Ledermuller, DeepWeb navigation in web data extraction, in *Proc. of Int. Conf. Computational Intelligence for Modelling, Control and Automation and Int. Conf. Intelligent Agents, Web Technologies and Internet Commerce* (*CIMCA-IAWTIC'06*), Vienna, Austria, 2005, pp. 698–703.

[21] D. Liu, L. Ma, and X. Liu, Research on adaptive wrapper in deep web data extraction, in *Internet of Vehicles-Safe and Intelligent Mobility. IOV 2015. Lecture Notes in Computer Science, vol. 9502*, C. H. Hsu, F. Xia, X. Liu, and S. Wang, eds. Chengdu, China: Springer, 2015, pp. 409–423.

[22] R. Girshick, J. Donahue, T. Darrell, and J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, arXiv preprint arXiv: 1311.2524v5, 2014.

[23] M. E. Basiri, S. Nemati, M. Abdar, E. Cambria, and U. R. Acharya, ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis, *Future Generation Computer Systems*, vol. 115, pp. 279–294, 2021.

[24] J. Redmon and A. Farhadi, YOLO9000: Better, faster, stronger, in *Proc. of 2017 IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, HI, USA, 2017, pp. 6517–6525.

[25] R. Girshick, Fast R-CNN, in *Proc. of 2015 IEEE Int. Conf. Computer Vision* (*ICCV*), Santiago, Chile, 2015, pp. 1440–1448.

[26] S. Q. Ren, K. M. He, R. Girshick, and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, arXiv preprint arXiv: 1506.01497v3, 2016.

[27] R. Huang, J. Pedoeem, and C. X. Chen, YOLO-LITE: A real-time object detection algorithm optimized for Non-GPU computers, in *Proc. of 2018 IEEE Int. Conf. Big Data* (*Big Data*), Seattle, WA, USA, 2018, pp. 2503–2510.

[28] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, You only look once: Unified, real-time object detection, in *Proc. of 2016 IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, NV, USA, 2016, pp. 779–788.

[29] J. Hammer, J. McHugh, and H. Garcia-Molina, Semistructured data: The TSIMMIS experience, in *Proc. of 1st East-European Symp. Advances in Databases and Information Systems* (*ADBIS*), St. Petersburg, Russia, 1997, pp. 1–13.

[30] G. O. Arocena and A. O. Mendelzon, WebOQL: Restructuring documents, databases and webs, in *Proc. of 14th IEEE Int. Conf. Data Engineering*, Orlando, FL, USA, 1998, pp. 24–33.

[31] S. Soderland, Learning information extraction rules for semi-structured and free text, *Machine Language*, vol. 34, nos. 1–3, pp. 233–272, 1999.

[32] M. E. Califf and R. J. Mooney, Bottom-up relational learning of pattern matching rules for information extraction, *The Journal of Machine Learning Research*, vol. 4, pp. 177–210, 2003.

[33] D. Freitag, Information extraction from HTML: Application of a general machine learning approach, in *Proc. of 15th National/Tenth Conf. Artificial Intelligence/Innovative Applications of Artificial Intelligence*, Madison, WI, USA, 1998, pp. 517–523.

[34] C. N. Hsu and M. T. Dung, Generating finite-state transducers for semi-structured data extraction from the web, *Information Systems*, vol. 23, no. 8, pp. 521–538, 1998.

[35] A. Manjaramkar and R. L. Lokhande, DEPTA: An efficient technique for web data extraction and alignment, in *Proc. of Int. Conf. Advances in Computing, Communications and Informatics*, Jaipur, India, 2016, pp. 2307–2310.

[36] H. A. Sleiman and R. Corchuelo, Trinity: On using Trinary trees for unsupervised web data extraction, *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 6, pp. 1544–1556, 2014.

[37] J. Y. Wang and F. H. Lochovsky, Data extraction and label assignment for web databases, in *Proc. of the 12th Int. Conf. World Wide Web*, Budapest, Hungary, 2003, pp. 187–196.

[38] C. H. Chang and S. C. Kuo, OLERA: Semisupervised web-data extraction with visual support, *IEEE Intell. Syst.*, vol. 19, no. 6, pp. 56–64, 2004.

[39] Y. Wang, A new concept using LSTM Neural Networks for dynamic system identification, in *Proc. of 2017 American Control Conf.* (*ACC*), Seattle, WA, USA, 2017, pp. 5324–5329.

[40] E. Ferrara, P. De Meo, G. Fiumara, and R. Baumgartner, Web data extraction, applications and techniques: A survey,

Knowledge-Based Systems, vol. 70, pp. 301–323, 2014.

[41] Y. H. Zhai and B. Liu, Web data extraction based on partial tree alignment, in *Proc. 14th Int. Conf. World Wide Web*, Chiba, Japan, 2005, pp. 76–85.

[42] S. Kuamri and C. N. Babu, Real time analysis of social media data to understand people emotions towards national parties, in *Proc. of 8th Int. Conf. Computing, Communication and Networking Technologies* (*ICCCNT*), Delhi, India, 2017, pp. 1–6.

[43] D. G. Gregg and S. Walczak, Adaptive web information extraction, *Communications of the ACM*, vol. 49, no. 5, pp. 78–84, 2006.

Sudhir Kumar Patnaik received the MEng degree in electronics and communication from National Institute of Technology, Rourkela, India in 1995. He is currently a PhD candidate in computer science (machine learning) at M. S. Ramaiah University of Applied Science, Bangalore, India. He is working as the vice president of engineering and site leader at Gibraltar India Solutions LLP, Bangalore, India. Prior to Gibraltar India, he was the VP of platform engineering at Intuit India, for 13 years. His research interests are in the areas of data extraction, deep learning, and machine learning. He is a member of Industry Advisory Board at International Institute of Information Technology, Bangalore, and a member of the Board of Studies for Computer Science at Vellore Institute of Technology, Andhra Pradesh. He is also a senior member of IEEE, a fellow at Institution of Engineers (India), and a member of CSI, ACM, and ISTE.

C. Narendra Babu received the BEng degree in CSE from Adichunchanagiri Institute of Technology, India in 2000, the MEng degree in CSE from M.S. Ramaiah Institute of Technology, India in 2004. and the PhD degree from Jawaharlal Nehru Technological University Anantapur, India in 2015. He is currently an associate professor at the Department of Computer Science and Engineering, M. S. Ramaiah University of Applied Sciences, Bangalore, India. His research interests include artificial intelligence, machine learning, data analytics, social media analytics, and time series and spatio-temporal data modeling. He is a senior member of IEEE, a member of the IEEE Education Society, and a member of IAENG. He has published a book chapter, over twelve refereed journal papers, and eleven refereed conference proceeding papers.

Mukul Bhave received the MS degree in mathematics from Bundelkhand University, Jhansi, India in 1997, and the MS degree in business administration and management from Pt. Ravishankar Shukla University, Raipur, India in 1999. He is working as a software engineer at Gibraltar India Solutions LLP, Bangalore, India. He was previously employed at Intuit where he worked on the data aggregation platform. With over 16 years of software development experience, he has worked with Digital Insight, SoftwareAG (webMethods), and MindTree. His research interests are building application servers and platforms, deep learning, and web data extraction.