IJARSCT

ISSN (Online) 2581-9429

Impact Factor: 6.252

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

Volume 2, Issue 6, May 2022

Automated HTML Code Generation from Hand Drawn Images using Machine Learning Methods

Gayatri Vitkare[1], Rutuja Jejurkar[2], Sammyaka Kamble[3], Yogeshwari Thakare[4], Prof. P. A Lahare[5]

Student, Department of Information Technology[1,2,3,4]

Professor, Department of Information Technology[5]

Pune Vidyarthi Griha's College of Engineering & S. S. Dhamankar Institute of Management, Nashik, India

Abstract: *The production of individual web page mock-ups, which can be done by hand or with the aid of graphic design and professional mock-up creation tools, is the first stage in the website design process. The mock-ups are then converted into structured HTML or similar mark-up code by software engineers. Typically, this method is performed multiple times until the required template is achieved. The purpose of this review is to make the process of developing code from hand-drawn mock-ups more automated. Hand-drawn mock-ups are processed using computer vision techniques, and the recommended system is built using deep learning techniques. The building of a preliminary drawing of each web page, which can be done using design tools or by hand. After that, corresponding code for the web page draught is written. This procedure is difficult, expensive, and time-consuming. Consequently, the suggested system will automate this operation. A hand-drawn drawing of a form is provided as input, which is analysed, and several components revealed. After the components have been discovered, deep learning CNN techniques are used to crop and recognise them. When the matching component is identified.*

Keywords: Machine learning with convolution neural networks (CNN), Object detection, object recognition, HTML Code generation, HTML (Hypertext markup language)

I. INTRODUCTION

Because of today's technological advancements, the relevance of Internet web pages has expanded significantly. Nowadays, websites represent the personalities of nations, institutions, communities, and individuals. There are websites for practically any subject, from information to social work, games to training, and so on. Companies' websites are brought to the forefront for financial objectives, such as product promotion or advertising. Developers are in charge of creating client-side software based on a designer's mock-up of the Graphical User Interface (GUI). The implementation of GUI code, on the other hand, requires time and diverts engineers' attention away from the product's core functionality and logic. In addition, the computer Furthermore, each target runtime system's computer languages for designing such GUIs are separate, resulting in onerous and repetitive effort when the product is meant to function on several platforms using native technologies. In this research, we present a model that has been trained end-to-end with stochastic capable to reduce to construct variable-length tokens strings from a single GUI picture given input, yet staying technically and informational. An algorithm was created in this work to automatically produce HTML code for a hand-drawn mock-up of a web page. Its goal is to detect the mock-up picture's components and encode them according the web page structure.

II. LITERATURE SURVEY

A large gap occurs between graphic designers' conceptual designs and functional UI code while building the UI code of a mobile application. Programmers often close this gap by rewriting the conceptual drawings in code, which is time-consuming and costly. To fill this need, we pioneered the first method for automatically reverse engineering mobile app user interfaces (REMAUI). REMAUI uses computer vision and OCR methods to identify UI components on a given input image. REMAUI-generated UIs were close to the originals in our tests on 488 screenshots of over 100 major third-party programmes. We intend to expand the export process to more platforms, including iOS and cross-platform JavaScript frameworks. Currently, REMAUI separates each input screen into its own programme. We want to provide a graphical notation that users can use to connect many inputs screen designs, which REMAUI can then utilise to produce a single application with several screens and matching screen transitions. REMAUI will be integrated with tools that produce mobile app functionality using keyword-based code search [44] or high-level models.) We want to index a screenshot corpus by executing REMAUI on it and saving the intermediate results of REMAUI. By exposing this index through a query interface, a user might look for screenshots based on their structure and features.

Building classifiers for picture detection and recognition on multiple datasets using various machine learning had advanced significantly in recent years. Deep learning, in particular, has improved accuracy across a variety of datasets. The below are summaries of some of the works. SVM (support vector machines), KNN (K-nearest Neighbour), and CNN were used to evaluate the performance on MNIST datasets (convolutional neural networks). On that platform, the Multilayer Perceptron didn't function well since it didn't attain the global minimum but rather stayed caught in the local optimum and couldn't reliably distinguish digits 9 and 6.Other classifiers performed well, and it was decided that by building the model on the Keras platform, performance on CNN may be enhanced. On the MNIST dataset, implement DNN (deep neural networks), DBF (deep belief networks), and CNN (convolutional neural networks) and compare the results. According to the research, DNN fared the best, with an accuracy of 98.08 percent, while others had error rates and execution times that differed.

The study was classified into four categories. In the first stage, image processing methods such as erosion, dilation, and contour detection were being used to search for things in the input image. Following that, the identified objects were cropped, and the associated components were labelled using the trained CNN model. Finally, the model's output was transformed to HTML code using the HTML Builder script. Once the input data has already been read, it is transformed to grey scale style. Gaussian Blur was then reapplied to them using a 3x3 rectangle kernel. Following the thresholding step, the contour detection technique was used to create a rectangle in order to identify the objects basis of morphological changes. In this approach, the components of the uploaded picture have been detected. Cropping was used to transfer the identified components to the CNN model. In the phases of morphological features, 8 iteration dilation with a 4x4 rectangle kernel was implemented. With a 3x3 ellipse kernel, the erosion process was used for 4 repetitions, and the dilation process was used for 2 iterations with a 1x10 rectangle shaped kernel. Finally, a 10x1 rectangle shaped kernel was used in the dilatation operation. The model was trained using our component dataset chunks. It is made up of four major types of components, as previously stated: textboxes, drop-down menus, buttons, and checkboxes. After the model was trained, the loss function was trained for 200 epochs using Binary Cross Entropy and RMS Prop methodologies with a batch size of 64. After that, the component recognition procedure used the cropped elements from the preceding stage as input. For feature extraction, we utilized several convolution layers with 4x4 kernels, similar to our CNN Model, and thereafter max pooling operations using 2x2 kernels. We apply the BiLSTM layer to catch the relationship of the collected features after the feature vectorization procedure. After all, we utilised a 20% ratio of Full Connected Layers to Dropout Layers to achieve the classification aim. Recognized components were successfully translated into HTML code via the bootstrap framework. It was performed with the help of the coordinates from the result of the contour finding algorithms.

III. PROPOSED METHODOLOGY

The technology is used to automate the web page creation process. The system is given a hand-drawn image of a web form as input, and corresponding HTML code is to be created.There are four key phases to any process of developing robotic HTML code. The first step is object detection, which involves scanning the picture and using image processing algorithms such as LSTM, DSL, and so on. After then, the objects which have been identified are snipped. Eventually, the outcome of the model is converted to HTML code

3.1 Pix2Code

The method evaluates a snapshot of a graphical user interface (i.e., the layout of a web application interface) and determines what the underlying code looks like through an iterative process. Making computer code from a GUI screenshot is analogous to writing English written explanations from scene images. In both situations, we wish to generate variable-length token strings from pixel values. As a result, our problem may be divided into three pieces. To begin, understanding a given scene (in this example, the GUI display) and inferring the things there is a computer vision challenge, their IDs, two positions, and poses (i.e., buttons, labels, element containers). Secondly, reading and writing syntax correct language (in this case, computer code). It's a difficulty to represent language using conceptually correct examples. Lastly, and used the latent variables identified via scene translation, combine the responses including both sub-problems and provide corresponding textual descriptions (i.e., computer code rather than English) of the things represented by all these variables. The following parts make up Pix2code:

- LSTM (Long-Short Term Memory)
- CNN (Convolutional Neural Network)
- DSL (Domain specific Language)

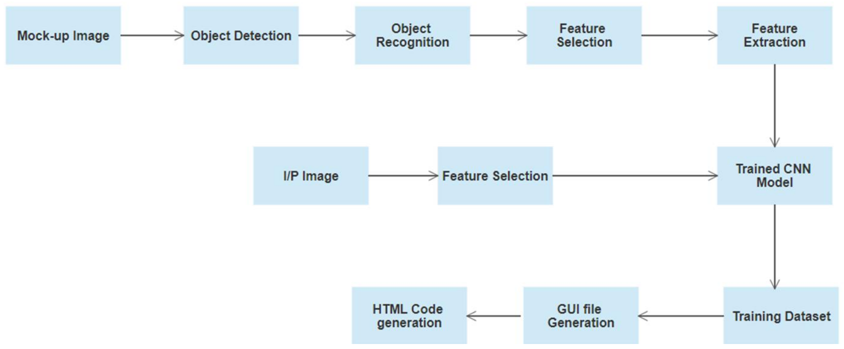


Figure: Proposed System Architecture

Detection of Objects

Textboxes, buttons, and labels are among the components of Dataset [5.] Contour detection and morphological transformation are used to detect these elements in a picture. Outside the component, contour detection is employed to draw boundaries. Using morphological erosion, little things are eliminated, leaving only substantial objects. Objects are made more apparent by filling gaps in them using morphological dilatation. These strategies improve object visibility and clarity.

Object Recognition

Convolutional Neural Network has used to train the dataset. Input image of web form then is given to this training set which recognises components inside an image. The image from the online form is then sent to this trained model, which recognises image components. Object recognition is a challenging computer vision problem that necessitates estimating both of the position and kind of entities seen in a picture.

Training data set

A dataset is a grouping of images with textboxes, buttons, and labels. A dataset then used train a machine learning technique is referred to as a training model.The training model is used to run the algorithm on the input data and compare the processed result to the predicted outcome.The model was derived based upon the results with this connection.

CNN

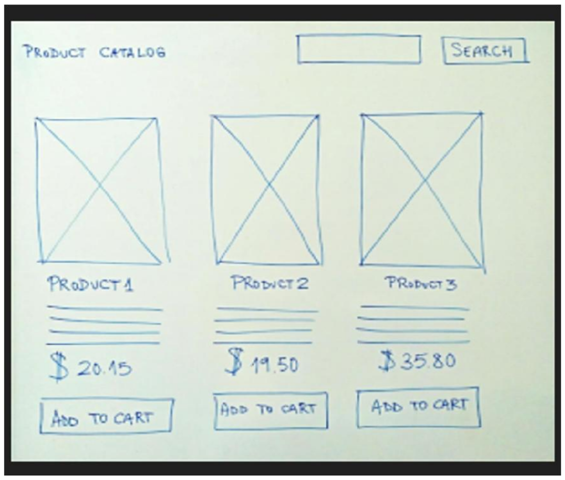
The elements were labelled using the trained CNN model after the recognised objects were clipped. Before even being fed through into CNN, the input pictures are scaled to 256*256 pixels (the aspect ratio is not kept) and the pixel values are balanced. There will be no further processing. We simply utilised 3*3 receptive fields convolved with stride 1 to encode each input image to a fixed-size output vector. To descending utilizing max-pooling, these processes are done again.

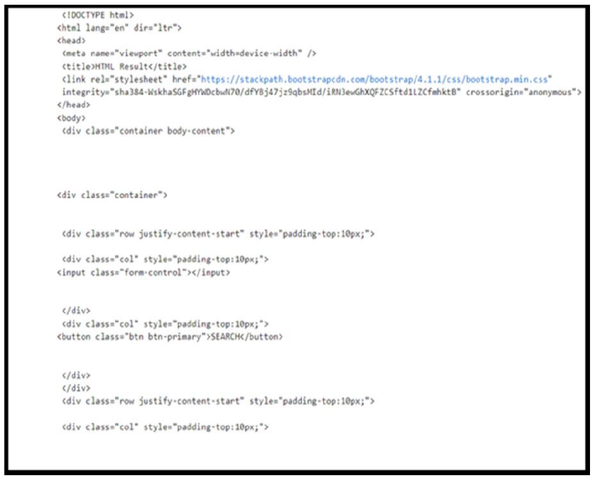
HTML Code Generation

For the detected components, HTML code is generated using the. Gui file created in the previous phase. The goal is to encode the web page's recognised components. First, header and footer templates are generated. Lastly the system creates web page file after generating the HTML code.

IV. RESULTS

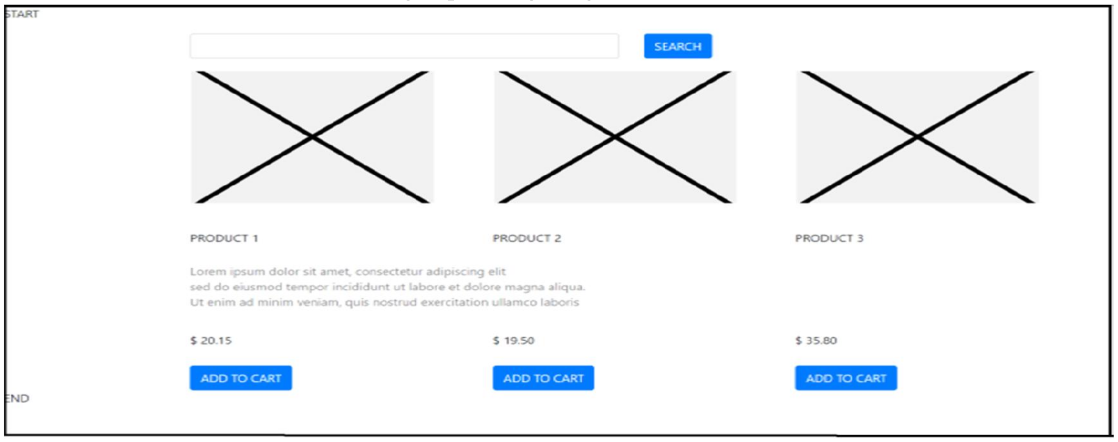
The goal of this research has been accomplished. We were able to generate HTML code from such a mock-up image with the addition of color buttons. All of the approaches are the foundational factors that have led to the current and planned system.





```
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
 <meta name="viewport" content="width=device-width" />
 <title>HTML Result</title>
 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css"
 integrity="sha384-WskhaSGFgHYWDcbwN70/dfYBj47jz9qbsMId/iRN3ewGhXQFZCSftd1LZCfmhktB" crossorigin="anonymous">
</head>
<body>
 <div class="container body-content">




<div class="container">


 <div class="row justify-content-start" style="padding-top:10px;">

 <div class="col" style="padding-top:10px;">
<input class="form-control"></input>


 </div>
 <div class="col" style="padding-top:10px;">
<button class="btn btn-primary">SEARCH</button>


 </div>
 </div>
 <div class="row justify-content-start" style="padding-top:10px;">

 <div class="col" style="padding-top:10px;">
```

Fig. Input Image Fig.HTML Code



Lorem ipsum dolor sit amet, consectetur adipiscing elit
sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris

Fig. Output

V. FUTURE WORK

- Expanding system adaptability by allowing users to generate front-end designs with a range of image types.
- Improving the system so that more appealing designs can be created.
- Adding a new function that allows users to customise the CSS for the website's front end to improve system quality.

VI. CONCLUSION

A critical aspect has been converting website mock-ups into mark-up code with less time and development expense. We built a technique in this work that receives web page mock-ups, processes them, and generates structured HTML code. A collection of images was used, which included several mock-ups of web page architectures. Future research and expertise are needed to improve the code generation's efficiency. Such apps are needed in the industry and in our daily lives since every firm insists on making their laborious task easier and more efficient. The above study presents a system that turns picture data into an HTML webpage automatically.

REFERENCES

[1]. Automatic HTML Code Generation from Mock-up Images Using Machine Learning Techniques978-1-7281-1013-4/19/$31.00 © 2019 IEEE

[2]. Convolutional Neural Network (CNN) for Image Detection and Recognition 978-1-5386-6373-8/18/$31.00 ©2018 IEEE

[3]. Reverse Engineering Mobile Application User Interfaces with REMAUI 978-1-5090-0025-8/15 $31.00 © 2015 IEEE DOI 10.1109/ASE.2015.32

[4]. pix2code: Generating Code from a Graphical User Interface Screenshot arXiv:1705.07962v2 [cs. LG] 19 Sep 2017

[5]. Norhidayu binti Abdul Hamid, Nilam Nur Binti Amir Sjarif, "Handwritten Recognition Using SVM, KNN and Neural Network", www.arxiv.org/ftp/arxiv/papers/1702/1702.00723

[6]. Mahmoud M. Abu Ghosh; Ashraf Y. Maghari, "A Comparative Study on Handwriting Digit Recognition Using Neural Networks", IEEE, 2017

[7]. T. A. Nguyen and C. Csallner, "Reverse Engineering Mobile Application User Interfaces with REMAUI (T)," in 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, nov 2015, pp. 248–259. [Online]. Available: http: //ieeexplore.ieee.org/document/7372013/

[8]. S. Natarajan and C. Csallner, "P2A: A Tool for Converting Pixels to Animated Mobile Application User Interfaces," Proceedings of the 5th International Conference on Mobile Software Engineering and Systems - MOBILESoft '18, pp. 224–235, 2018. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3197231.3197249.

BIOGRAPHY

1. Gayatri Vitkare,Under Graduate Student,PVGCOE & SSDIOM,Nashik, Maharashtra, India
2. Rutuja Jejurkar,Under Graduate Student,PVGCOE & SSDIOM,Nashik, Maharashtra, India
3. Yogeshwari Thakare,Under Graduate Student,PVGCOE & SSDIOM,Nashik, Maharashtra, India
4. Sammyaka Kamble,Under Graduate Student,PVGCOE & SSDIOM,Nashik, Maharashtra, India
5. P.A.Lahare, Professor,PVGCOE & SSDIOM,Nashik, Maharashtra, India