Terminator Vision from T2 -1991 Movie

Navigating the Document Object Model in Auto Pilot mode

Deepak Noveen G
Application Security & Solution Engineering

1 article

January 13, 2024

Open Immersive Reader

Have we ever wondered how a popular object detection model like YOLO (You Only Look Once) is capable of doing real-time object detection for autonomous vehicle driving or medical imaging, and why we still don’t have any autonomous web bot which can, not only navigate but understand websites on its own?

While there has been a recent rush in building GPT based browser automation apps and plugins, none of these tools can perform real time web navigation on its own. Either the HTML file or a PDF or an IMG snapshot of the web page needs to be uploaded to the GPT server for the LLM to understand and extract information from a single web screen. There is also the ChatGPT - Browse with Bing option which is available to premium users and its use case is also limited to searching the internet to help answer questions that benefit from recent information.

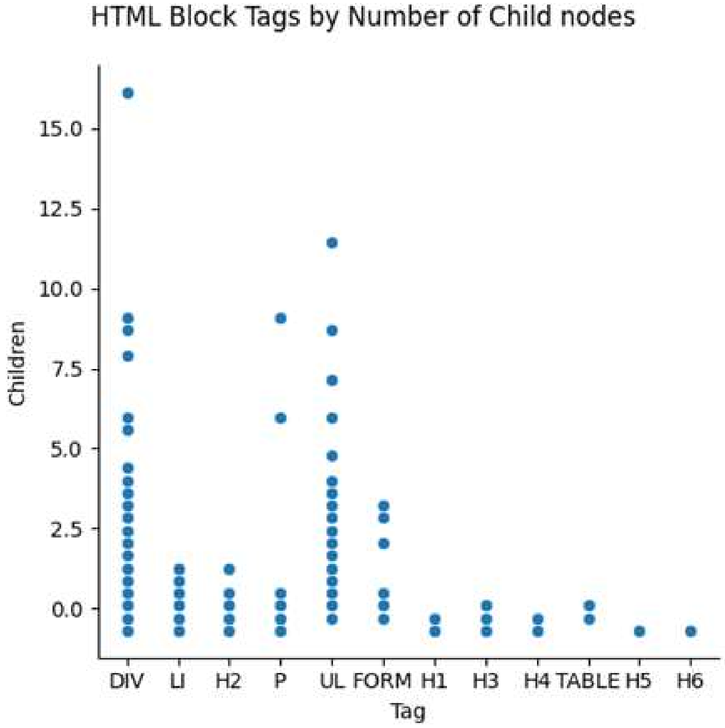
Ideally, a web bot should be able to automate the entire website navigation flow which includes identifying the website's layout and section for menus, form inputs, form submission controls, advertisements, user posted content and the web layout style., These kind off bots may enhance the capabilities of RPAs and Web Application Scanning tools.

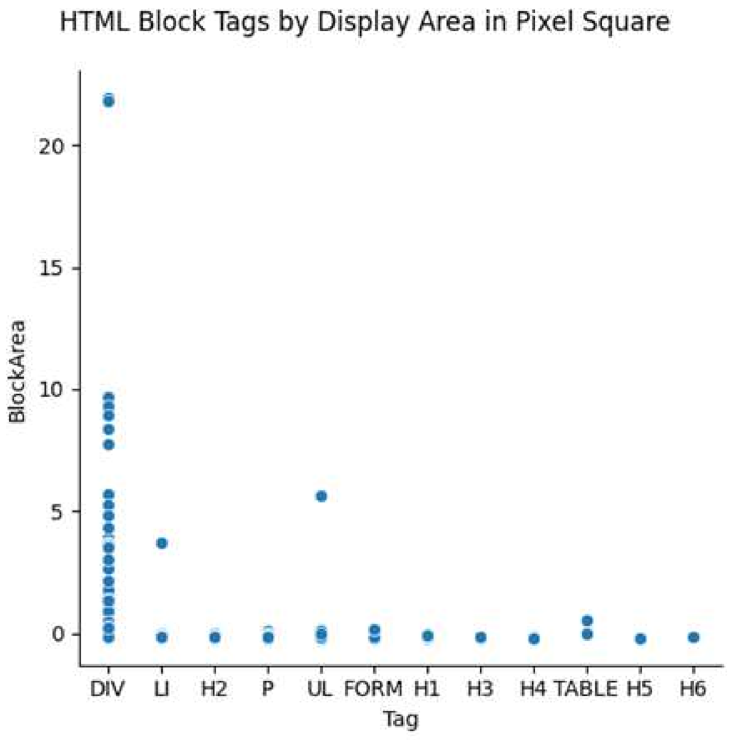
The research paper titled **"Web2Text"** (dated March 2018), proposes a machine learning method using Hidden Markov Model on top of potentials(a mathematical function in a neural network) derived from DOM tree features using Convolutional Neural Networks to perform sequence labeling and then to collectively classify all text blocks in an HTML page as either boilerplate or main content. The HTML DOM Tree structure is parsed and the block level element features along with the information from adjacent neighbors are extracted and fed into the model. While the sample code shared in this paper was effective in identifying static text blocks within a web page, it still relied on deep learning, which uses more resources like computing and memory and it does need more data features (128 for each text block).

Based on the above paper, if we look at other lightweight learning options such as Bayesian networks, Binary Search Trees, and Decision Trees to trial and error and find why these methods cannot do what a Markov model can do. None of these methods can fully address the challenges posed by the complex hierarchical structure of HTML nodes and elements. The Over-nested tags, the Free-form style of HTML grammar, the Context changes of HTML elements and the Dynamically loaded html content are few of the challenges. And these methods will not be able to retain the context or meaning of the information on any website.

This leads to the question of why we cannot use the traditional object detection techniques on a HTML DOM? i.e. Detect a login form either by scanning through the HTML DOM Tree from top to bottom nodes or from bottom to top or just by navigating through the display area from left to right just like how a human would read the content in a web page. The only hurdle in using normal computer vision techniques for HTML is that current techniques process the input as image pixels. Each pixel will represent a neuron in a neural network and then we need to do some deep learning. In a normal computer vision process, the AI component interacts with real world objects through a camera or image feed which is pixel based and the data is unstructured. But the HTML DOM is structured; why we would need deep learning for machine-to-machine interactions? i.e. AI to Web Server. The ideal lightweight approach should scan the structured DOM Tree with full understanding of HTML element's context and the embedded content's meaning.

To experiment with this approach, HTML DOM data from a few websites were captured (Node.js scripts with Puppeteer library for producing the training data) and then few numerical features were extracted. Below we can find a scatter plot created from HTML DOM data extracted from home pages of popular websites. The numerical data points are scaled to fit the graph. We can see how the DIV and UL tags contain more child nodes, but the pixel display area is always more for a DIV block.





An unsupervised learning technique for segmenting these HTML block-level elements and some linear classifier to select the appropriate type of block should be able to help us out here. And I am confident that this effort will yield valuable insights or at least correct my oversight. Stay tuned for updates as I delve deeper into this exciting journey!

#AI #MachineLearning #WebNavigation #RPAs #Bots #WebApplicationScanning #GeminiProAssisted #ComputerVisionForHTML #UnSupervisedLearning #ColabAI

Published by

Deepak Noveen G
Application Security & Solution Engineering
Published • 1w

1 article

My first LinkedIn post on object detection techniques for HTML DOM... 🙄

Like Comment Share