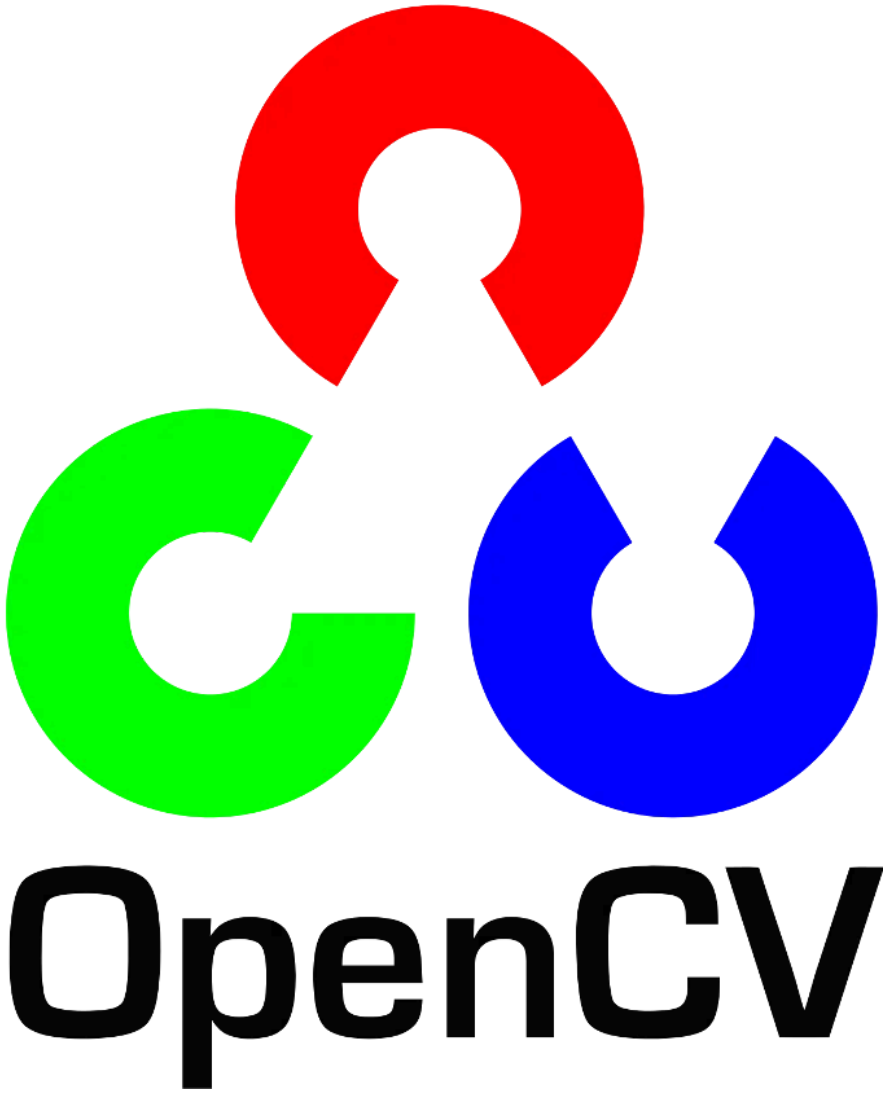

4/15/24, 5:12 PM Using OpenCV.js in Browser based environment | by Aswin Vijayakumar. | Analytics Vidhya | Medium

Using OpenCV.js in Browser based environment

Aswin Vijayakumar. · Follow
Published in Analytics Vidhya · 5 min read · Oct 31, 2019

OpenCV is a powerful tool which does image manipulation and processing at a fast pace. To bring the OpenCV library in the browser the browser needs to support libc++ library which almost all browsers support.

Asm.js is written in C++ that browsers use in their native environment. The **OpenCV.js source** provided here has been compiled using Emscripten (EMSDK) which makes use of Asm.js and libc++ to transpile C++ code to the javascript code. Javascript runs in a single threaded environment, but Emscripten has something which is similar to tasks in C++ (Openmp) called Web Workers. With the WebAssembly library supported by recent browser versions multi-threading is possible. WebAssembly is commonly used in WebGL and benchmarking results are brilliant.

Computer Vision involves Pattern Recognition, Artificial Intelligence and Machine Learning. In a client server model, the server need not own the complete Sequence Diagram model of a given task. It is in some use cases, ideal to split the Sequence Diagram into two contexts: Client and Server contexts.

An OpenCV project that is reliant on Client-side and Server-side

The table shown below is from the post on **Approaching any Machine Learning problem**. It separates the phases of machine learning into :

- Training and validation split,
- Data Transformation
- ML Model Selector
- ML Entity Selector
- Hyperparameter Selector
- Feature Extraction
- Producing output via Optimization

While training the model, the data can be transformed into a variety of ways. A simple method is normalizing the data by subtracting the mean of the data and dividing by standard deviation. OpenCV on the client side can transform the data and extract the features such as in face detection it could be to extract the face location.

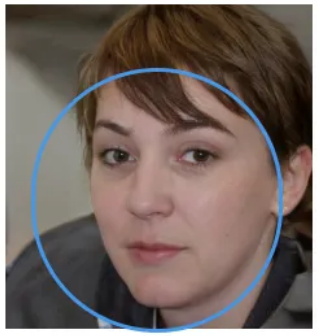

PHOTO 1

Taken using https://aidemos.microsoft.com/face-recognition, @Courtesy https://thispersondoesnotexist.com/

OpenCV has many cascade algorithms such as Viola Jones, HOG (Historical Oriented Gradients). Viola Jones is an unsupervised algorithm that extracts features using proximal pixels. It does image integration of the image similar to what image convolution does but has a restriction that the image sizes need to be specified. Viola Jones is an Unsupervised learning algorithm whereas Convolution Neural networks use Supervised learning techniques unless it is a case of feature extraction using CNNs. They are classified usually as Haar Cascades in OpenCV Library. I believe we need better Service standards for AI, Machine Learning and Computer Vision. The standards that lead us to better performant models.

I’ve got a special affinity towards Design by Contracts in Object-Oriented Methodology. It goes by this technique. If a contract initiator has certain responsibilities, the client has certain obligations towards it. Also if the client has certain responsibilities, the contract initiator has certain obligations towards the client. They are documented in the table below:

	Obligations	Benefits
Client		
Server		

gistfile1.md hosted with ❤ by GitHub view raw

OpenCV.js is best for thresholding on the client side as shown here. Other tasks that client side can do are:

- Erosion
- Dilation
- Convolution
- Laplacian
- Image Gradients
- Region Growing
- Image Segmentation

Image segmentation using region growing and thresholding is especially a crude method, and that requires us to provide parameters to the model in order to get the work done. So this domain is very useful to be executed on the client-side.

Result of Image Thresholding

There are many libraries that make use of parallel computing, such as Emscripten to transpile JS code, **SIMD.js**, Asm.js. From the developers point of view, they make the tasks easy to execute and even do more by localising the computational power.

To setup OpenCV.js in Browser

```
<script type=”text/javascript”>
    var DISABLE_EXCEPTION_CATCHING = 2;
</script>
```

Calling OpenCV methods in browser environment

```
let src = cv.imread(document.getElementById("img_tag_id"));
cv.cvtColor(src, src, cv.COLOR_RGBA2GRAY, 0);

let marker = new cv.Mat(rows,cols,cv.CV_8UC1);

try {
  cv.adaptiveThreshold(src,marker,255,cv.ADAPTIVE_THRESH_GAUSSIAN_C,
```

```
cv.THRESH_BINARY,11,2);
}
catch(e) {
  throw e;
}
```

While coding make sure you write to the same object because that way you can reduce the memory consumption. This is ECMAScript, that works in Chrome but not in every browser. You can use **Webpack** to compile ECMAScript to native JS.

Data formats and Datasets

> A simple example could be that “the arrays are written using Javascript Array notation”

The data formats that any client side algorithm uses must comply with the formats used by the datasets. This could encapsulate the data transformation routines, shapes and sizes of images, classes that one image will map onto, etc. I like Microsoft’s CNTK BrainScript because it is a text-based format and very useful for capturing requirements from the client-side. A simple example could be that “the arrays are written using Javascript Array notation”. I believe CNTK’s BrainScript text readers are an art of invention because they fulfil the best needs that any algorithm wants. A machine learning framework that is mostly reliant on the client side must use data formats that BrainScript produces. This sort of a technique is similar to the React’s technique which uses JSX style syntax to capture almost all functionality that a user wants with almost close native implementation.

Microsoft Azure uses diagramming based models to automate the process of delivering a Machine Learning solution, that is good and provides us the flexibility to design some ML tasks.

Featuretools

Featuretools (Python based) package is a feature synthesis package that creates relationships between data attributes. This is somewhat similar to Data Warehousing because it can generate a fact table, or even a star schema. The data warehousing standards say that we need to split Date into dimensions of Time, Hour, Day, Month, Year, etc. This is super important for featuretools as well because they work with temporal data often. For synthesis and workflow automation, temporal data is a must.

OpenCV and Feature Engineering

This is an excerpt of cascade **haarcascade_smile.xml.** See what metadata is stored in the cascade file. It looks at those Viola Jones proximal pixels and measures.

```
<opencv_storage>
<cascade type_id="opencv-cascade-classifier">
<stageType>BOOST</stageType>
  <featureType>HAAR</featureType>
  <height>18</height>
  <width>36</width>
  <stageParams>
    <maxWeakCount>53</maxWeakCount></stageParams>
  <featureParams>
    <maxCatCount>0</maxCatCount></featureParams>
  <stageNum>20</stageNum>
  <stages>

........
```

```
            <_>
              <internalNodes>
                0 -1 105 -5.1861900836229324e-02</internalNodes>
              <leafValues>
                7.0431172847747803e-01 -2.2143700718879700e-
01</leafValues></_>
            <_>
              <internalNodes>
                0 -1 106 -5.0341628491878510e-02</internalNodes>
              <leafValues>
                -4.6397829055786133e-01 2.8047460317611694e-
01</leafValues></_>
```

Conclusion

I have got to write more on the topic, because the topic on synthetic datasets and front-end excites me. Try writing some threading models in C++ and compile into JS code using openmp. You could see the close to native experience in Javascript. There is ample opportunity in using ONNX.js models in browser based environment. Some of the tools that do the work are:- CoreMLTools (from Apple), ONNX.js (from Microsoft), TensorFlow.js (from Google). The very task of creating a tensor meshgrid is difficult in any of these algorithms. With the client-side processing of vectorized data, there are many use cases of problem solving possible and reduces compute power.

Machine Learning | Opencv | Computer Vision | Programming | Data Science