如何在浏览器运行深度神经网络？以人脸口罩识别为例进行讲解

原创 AIZOO元峰 AIZOO 2020-02-28

一般来说，深度学习都是运行在服务器或者以原生应用的方式运行，而谷歌开源的TensorFlow.js库，则可以让深度学习，高效率的跑在用户的浏览器、Node.js环境甚至微信小程序里面。

前几天，AIZOO开源的口罩数据得到了很多粉丝朋友的关注，我们也将Keras框架训练的模型顺利的转成了PyTorch、TensorFlow、MXNet和Caffe模型，并将其对应的推理代码进行了开源。不过，仍然有不少的朋友好奇我们是如何让人脸口罩检测模型跑在本地浏览器里的。今天，元峰就简单的介绍一下。

为此，我们专门建立了一个极简的web demo，里面包含可以在浏览器内部运行人脸口罩检测的全部核心代码，并将其在Github开源了。

Github链接如下：
https://github.com/AIZOOTech/mask-detection-web-demo

首先，大家可以通过一个动图简单看一下效果。

如果要体验完整的网页，建议大家还是进入我们部署在AIZOO.com上的网页，链接如下：

https://aizoo.com/face-mask-detection.html

下面，就让元峰来简单介绍一下，如何在浏览器内运行深度学习模型。

在介绍本文的主角TensorFlow.js之前，我们先简单的介绍一下浏览器的三大核心编程语言HTML、CSS和JavaScript。首先，HTML是超文本标记语言的简称，其主要书写网页的主要内容，比如标题、段落、图片等；CSS是层叠样式表的简称，其主要功能就是给内容增加样式，例如修改字体的大小和颜色、网页背景等样式，让网页更美观；而JavaScript则是一门脚本语言，用于交互，例如添加按钮的事件，与服务器端通信，添加网页动画，可以说JavaScript是让网页动起来的。

JavaScript是一门非常强大的语言，该语言诞生于1995年，由网景公司的Brendan Eich开发而成，这位天才程序员，只用了是10天就完成了初版开发，说该语言集成了Java、C、Perl、Python、Scheme等语言的语法特点，非常简单灵活。JavaScript还有一个标准，叫做EcmaScript（简称ES），ES标准规范了JavaScript的基本语法结构，而且该标准在快速的迭代中，自2015年开始，每年都会更新。而遵循ES标准的，还有大名鼎鼎的Typescript，该语言由微软开发，语法比JavaScript更加严格。而TensorFlow.js，正是使用Typescript开发的。

JavaScript语言不仅可以运行在浏览器，2009年，Ryan Dahl将谷歌的Chrome V8引擎（谷歌开发的JavaScript的解释器）移植到服务器端，从此，JavaScript不仅可以运行在浏览器内部，还可以运行在服务器端，使得该语言可以与Python、PHP等脚本语言平起平坐。

好的，关于前端的基本知识，就铺垫这么多，下面，我们介绍一下本文的核心，TensorFlow.js。

1

TensorFlow.js简介

谷歌开源的TensorFlow框架，不仅开发了Python、Swift和Java等语言的接口，使得TensorFlow可以运行在PC、iOS、安卓等设备，另外，谷歌还为浏览器环境和Node.js环境开发了JavaScript版本，也就是TensorFlow.js。借助TensorFlow.js，我们可以很方便的在浏览器里面进行深度学习的推理，甚至训练。谷歌官方也给出了大量的示例。链接如下：

https://github.com/tensorflow/tfjs-models

在浏览器内使用TensorFlow.js与python比较类似，可以说很多API使用都是非常类似的风格，大家来感受一下：

```
let arr = tf.tensor([[1, 2], [3, 4]])
console.log(arr)
let arr1 = tf.reshape(arr, [-1,1])
console.log(arr1)
```

Edit Run

```
Tensor
    [[1, 2],
     [3, 4]]
Tensor
    [[1],
     [2],
     [3],
     [4]]
```

AIZOO

上图中，我们定义了一个2x2的矩阵，然后将其reshape成4x1维，如果有python使用TensorFlow的经验，可以说非常容易上手TensorFlow.js。在TensorFlow的python

接口中常用的add、concat、div、nonMaxSuppression等函数，JS版本都有同名的函数，用起来也非常方便。

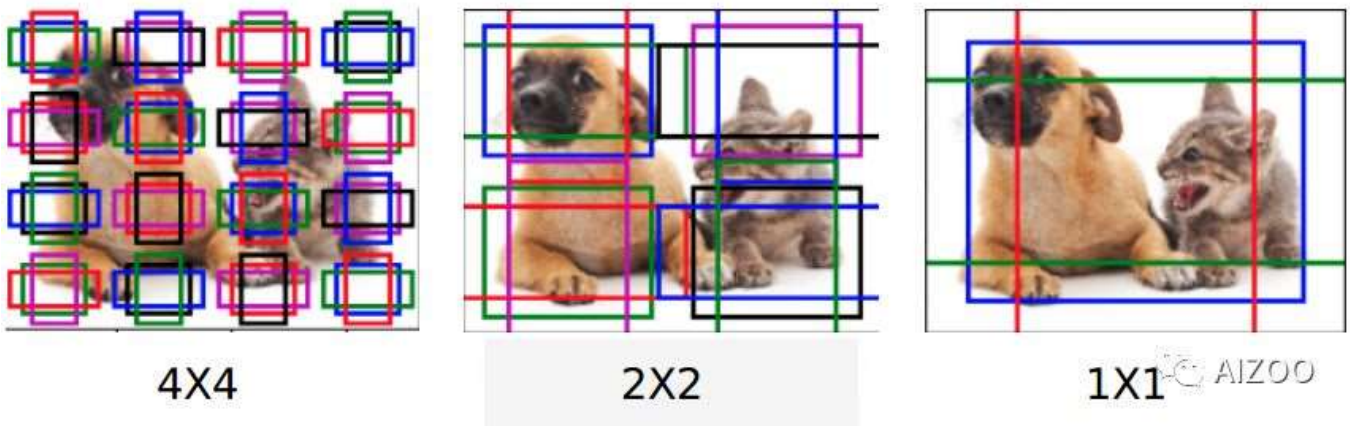
限于篇幅原因，这里不再过多介绍TensorFlow.js，具体的大家可以上TensorFlow官网查看教程。下面，我们说一下如何在浏览器实现人脸口罩检测。

2

目标检测推理的三大核心操作

可能有些朋友并不是很清楚目标检测模型的推理过程，这里，元峰简单的介绍一下。因为我们使用的是SSD模型架构进行人脸口罩检测，这里就以其为例进行介绍。

Faster-RCNN、SSD、Yolo v3等算法的一个共同点，就是他们都是基于锚点（anchor）的。锚点，简单来说，就是在图像上设置的密集的、不同大小和长宽比的参照框，下面是《动手学深度学习》这本书中的示例图。



如上图所示，是在输出大小为4x4、2x2、1x1大小上进行设置的anchor，每个位置有3个不同大小、不同长宽比的anchor。当然，对于一般的网络，anchor是非常密集的，例如，SSD作者是在38x38、19x19、10x10、5x5、3x3、1x1的特征图上设置的anchor。**而神经网路的输出，则是相对于anchor的中心点、长和宽的偏移量，以及该anchor内有何种物体（猫和狗）的置信度。**所以，在SSD模型推理的时候，也就是根据预设的anchor，**解码出真实的物体坐标**，然后做**非最大值抑制**操作即可。

所以，基于SSD检测模型推理，我们只需要实现以下三大操作，写三个函数就OK了。

- 全部anchors生成
- 网络输出值根据生成的anchors解码

- 非最大值抑制

我们在Github上开源的代码中，在detection.js文件中，实现了以上三大函数，分别为：

```
function anchorGenerator()
function decodeBBox()
function nonMaxSuppression()
```

我们先生成anchors，然后得到网络的输出 -> 解码 -> 非最大抑制 -> 得到输出， 最后将网络的结果画到图片上，整个过程就完成了。

等等，说到这里，我们好像没讲怎么在浏览器得到网络输出。下面，我们介绍一下如何在浏览器运行网络。

3

在浏览器运行深度神经网络

我们一般使用TensorFlow或者Keras训练模型，然后保存为pb或者hdf5模型，但是这个模型，并不能直接使用TensorFlow.js加载模型进行前向推理，需要使用工具，转换为TensorFlow.js支持的模型格式。

我们需要先下载相应的转换工具tensorflowjs，然后运行转换tensorflowjs_convert进行模型转换，以keras模型为例：

```
pip install tensorflowjs // 在本地安装tensorflowjs
// 转换模型
tensorflowjs_convert --input_format keras --output_format tfjs_layers_model   /
```

模型的输出是一个 `model.json` 文件和一个或者多个 `bin` 文件，其中前者保存模型的拓扑，后者保存模型的权重。

接下来，我们在HTML页面添加tensorflow.js文件的引入：

```
<script src="tfjs.min.js"></script>
```

我们将检测相关的代码放到detection.js文件中，其中最重要的是要加载模型，我们的模型相对路径为./tfjs-models/model.json，则加载代码为：

```
model = await tf.loadLayersModel('./tfjs-models/model.json');
```

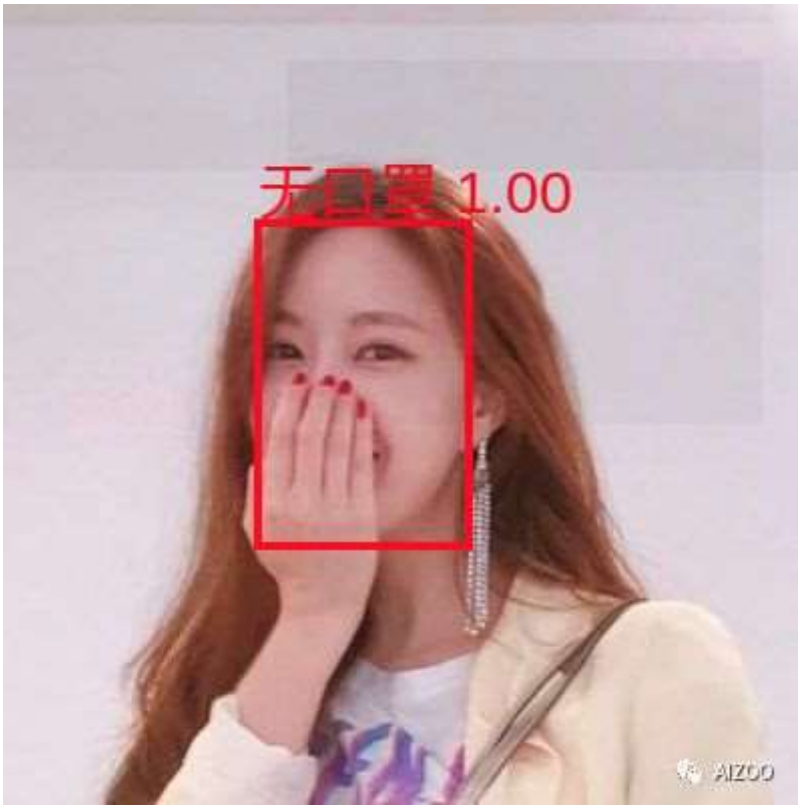
对于一个图像，也就是HTML中的image元素，假设该元素名字为imgToPredict，对于模型预测，要先要将image元素转为矩阵，再归一化既可进行前传了。

```
let img = tf.browser.fromPixels(imgToPredict);
img = tf.image.resizeBilinear(img, [260, 260]);
img = img.expandDims(0).toFloat().div(tf.scalar(255));
const [rawBBoxes, rawConfidences] = model.predict(img);
```

对于网络的输出，进行解码和非最大抑制，就得到了最终的输出：

```
const bboxes = decodeBBox(anchors, tf.squeeze(rawBBoxes));
const Results = nonMaxSuppression(bboxes, tf.squeeze(rawConfidences), 0.5, 0.5,
```

最终的输出，则包含检测到的物体的坐标和类别，以及相应的置信度，也就是[xmin, ymin, xmax, ymax, classID, score]，我们只需要将结果通过HTML的canvas（画板）控件，将结果画到图像上，结果如下图所示，至此，整个项目就完成了。



关于整个项目的代码，我们已经开源到Github上，感兴趣的用户只需要下载该项目，进入该项目内，打开终端，您可以使用以下方式打开一个web server。

Python用户：

```
// python3用户
python -m http.server
// python2用户
python -m SimpleHTTPServer
```

Node.js用户：

```
// 您可以使用serve
npm install serve -g //安装serve
serve // 运行该命令即可打开一个web server
// 您也可以使用http-serve
npm install http-server -g
http-server
```

然后终端会显示本地ip:端口号，例如127.0.0.1:5000，复制该地址到浏览器，就可以打开网页了，你可以点击“打开本地图片”按钮选择本地图片，也可以直接拖动一张图片到网页区域，页面如下。

至此，我们就可以，在浏览器进行口罩检测了。具体代码，大家可以查看我们开源的demo代码。

再次写一下我们开源在Github代码链接：

https://github.com/AIZOOTech/mask-detection-web-demo

本文完～

点击查看往期内容回顾

AIZOO开源人脸口罩检测数据+模型+代码+在线网页体验，通通都开源了

人脸口罩检测现开源PyTorch、TensorFlow、MXNet等全部五大主流深度学习框架模型和代码

2020年代，中国AI创业公司将走向何方

我是元峰，互联网+AI领域的创业者，欢迎扫描下方二维码，或者直接在微信搜索“AIZOO”关注我们的公众号AIZOO。

如果您是有算法需求，例如目标检测、人脸识别、缺陷检测、行人检测的算法需求，欢迎添加我们的微信号AIZOOTech与我们交流，我们团队是一群算法工程师的创业团队，会以高效、稳定、高性价比的产品满足您的需求。

如果您是算法或者开发工程师，也可以添加我们的微信号AIXZOOTech，请备注学校or公司名称-研究方向-昵称，例如“西电-图像算法-元峰”，元峰会拉您进我们的算法交流群，一起交流算法和开发的知识，以及对接项目。

公众号“AIZOO”

扫描二维码
关注我们

AIZOO

小助手微信号“AIZOOTech”

扫描二维码
添加元峰

AIZOOTech

阅读原文

喜欢此内容的人还喜欢

多摄像头实时目标跟踪和计数，使用YOLOv4，Deep SORT和Flask

AIZOO

一句“都挺好”，听到心酸

夜叔

知乎上40个有趣神回复，绝了！

思想新知