Detecting Faces in an Image

1. Introduction to the Watson Visual Recognition API 2

2. Acquiring the API Key 3

3. Getting Started with the Lab Material 4

4. Introduction to the Node.js Code 5

5. Introduction to the Code 6

6. The /detectfaces Route 7

7. Appendix 8

8. First Time with Node.js? 8

9. JSON 8

# Introduction to the Watson Visual Recognition API

The IBM Watson™ Visual Recognition service uses deep learning algorithms to identify scenes, objects, and celebrity faces in images you upload to the service. You can create and train a custom classifier to identify subjects that suit your needs.

Source: [https://www.ibm.com/watson/developercloud/visual-recognition/api/v3/ - introduction](https://www.ibm.com/watson/developercloud/visual-recognition/api/v3/#introduction)

Additional Information: https://www.ibm.com/watson/developercloud/doc/visual-recognition/

# Acquiring the API Key

1. You will first need to have an IBM Bluemix ID to be able to access the Watson APIs. Sign-up and configure your IBM Bluemix ID here  
     
   <https://console.ng.bluemix.net/>
2. Once you have completed that process, login and create a new application
3. Search for “Visual Recognition” and select that service under the Watson section on the Bluemix Console.
4. Confirm or change the service name and credential name. Make sure you start with the free plan and then click create.
5. Select your service on the console in the “All Services” section.
6. Identify and copy your API key from the service to be used through the rest of this lab

# Getting Started with the Lab Material

* Acquire an API key before you start
* Install node + npm <https://nodejs.org/en/download/>
* Download and save the lab code to an empty folder
* From the command line run "node --version"
* From the command line run "npm --version"
* From the command line run the following commands to install Node components to the global (-g) environment needed for the lab

npm install -g express

npm install -g nodemon

npm install -g body-parser

npm install -g url-exists

npm install -g request

npm install -g fs

npm install -g ejs

npm install -g watson-developer-cloud

* From the command line change to the directory where you downloaded the lab code and run the following commands

run "npm init" in the directory where the package.json file is to pull in the necessary Node.js component to the project folder

run "npm start" to launch the application

# Introduction to the Node.js Code

The /classifyimage route takes a URL from a form POST method and calls the Watson Visual Recognition API. The data returned from the API is a JSON object.

[Find details about the JSON object here under the RESPONSE section](https://www.ibm.com/watson/developercloud/visual-recognition/api/v3/#classify_an_image)

The JSON object is nested and the data we are interested can be found here

images: response.images[0].faces

The call to the Watson API is done here where params contains either a pointer to a local file or a URL. This lab will be focusing on URLs but indicates where you will need to specify the local file.

visual\_recognition.detectFaces(params, function(err, response) {

# Introduction to the Code

The URL is being past in from a form (index.ejs) with the POST method and we extract the value from the request body

var imageurl = req.body.imgurl;

Before passing the URL to the Watson API, let verify it actually exists on the Web. The urlExists function from reaches out on the Web and checks whether the URL actually is valid or not.

urlExists(imageurl, function(err, exists) {

Once we have established that the URL is valid, we construct the params object and assign the URL to the url parameter. If you are passing local files to the API, use images\_file

var params = {

// Usage images\_file to pass in a file from the local file system

//images\_file: fs.createReadStream('images\_animals/bws\_5d\_turkey\_061.jpg')

//Use url to pass in an image from the Internet

url: imageurl

};

All data required for the API call have now been gathered and the request can be executed

visual\_recognition.detectFaces(params, function(err, response) {

if (err) {

if ( debug ) { console.log( "There was a problem calling the Watson API" ) }

console.log(err);

} else {

// The Watson API call was successful, process the content

if ( debug ) {

console.log( "The call to the Watson API was successful, let's render the data" )

console.log( response.images[0].faces );

};

res.render('pages/detectfaces',{

mytext: imageurl,

images: response.images[0].faces

});

}

});

# The /detectfaces Route

The full Node.js code below passes the JSON object to a view (classifyimage.ejs) eventually rendering the content returned by the Watson API.

// Detect faces in an image

app.post('/detectfaces', function(req, res) {

if ( debug ) { console.log( "POST request was submitted for detecting faces of an image" ) }

var imageurl = req.body.imgurl;

urlExists(imageurl, function(err, exists) {

if ( exists ) {

if ( debug ) {

console.log( "Entering the detect faces code" );

console.log( "Does the URL actually exist => " + exists );

}

var params = {

// Usage images\_file to pass in a file from the local file system

//images\_file: fs.createReadStream('images\_animals/bws\_5d\_turkey\_061.jpg')

//Use url to pass in an image from the Internet

url: imageurl

};

if ( debug ) { console.log( "Calling the Watson API" ) }

visual\_recognition.detectFaces(params, function(err, response) {

if (err) {

if ( debug ) { console.log( "There was a problem calling the Watson API" ) }

console.log(err);

} else {

// The Watson API call was successful, process the content

if ( debug ) {

console.log( "The call to the Watson API was successful, let's render the data" )

console.log( response.images[0].faces );

};

res.render('pages/detectfaces',{

mytext: imageurl,

images: response.images[0].faces

});

}

});

}

});

});

# Appendix

# First Time with Node.js?

Look here for getting started material

<https://expressjs.com/en/starter/hello-world.html>

<https://www.youtube.com/watch?v=pU9Q6oiQNd0>

<https://www.youtube.com/watch?v=-u-j7uqU7sI&list=PL6gx4Cwl9DGBMdkKFn3HasZnnAqVjzHn_>

<https://nodejs.org/en/docs/>

# JSON

## The JSON Challenge

Use these web sites to look at the returned JSON and understand how to extract the data from the API

[www.jsonlint.org](http://www.jsonlint.org/)

<http://jsonmate.com/>

Use this Link to understand how to access the various bits inside the JSON object

<http://www.w3schools.com/js/js_json_arrays.asp>

// app.get('/', function(req, res) {

//

//     var myObj = {

//      "custom\_classes": 0,

//      "images": [{

//           "classifiers": [{

//                "classes": [{

//                     "class": "Manx cat",

//                     "score": 0.832,

//                     "type\_hierarchy": "/animal/mammal/carnivore/feline/domestic cat/Manx cat"

//                }, {

//                     "class": "domestic cat",

//                     "score": 0.948

//                }, {

//                     "class": "animal",

//                     "score": 0.968

//                }, {

//                     "class": "tomcat",

//                     "score": 0.702,

//                     "type\_hierarchy": "/animal/mammal/carnivore/feline/domestic cat/tomcat"

//                }, {

//                     "class": "tiger cat",

//                     "score": 0.554,

//                     "type\_hierarchy": "/animal/mammal/carnivore/feline/domestic cat/tiger cat"

//                }, {

//                     "class": "tan color",

//                     "score": 0.557

//                }],

//                "classifier\_id": "default",

//                "name": "default"

//           }],

//           "image": "bws\_5d\_turkey\_061.jpg"

//      }],

//      "images\_processed": 1

// }

//

// console.log('Custom Classes: ' + myObj.custom\_classes);

// console.log('Images Processed Classes: ' + myObj.images\_processed);

// console.log('=========');

// console.log(myObj.images[0].classifiers[0]);

// console.log('=========');

// console.log(myObj.images[0].classifiers[0].classes[0]);

// console.log('=========');

// console.log(myObj.images[0].classifiers[0].classes[1]);

// console.log('=========');

// console.log(typeof(myObj.images[0].classifiers[0].classes[4]));

// console.log('=========');

//

// });