

Machine Learning for the Web: an Introduction to Tensorflow.js



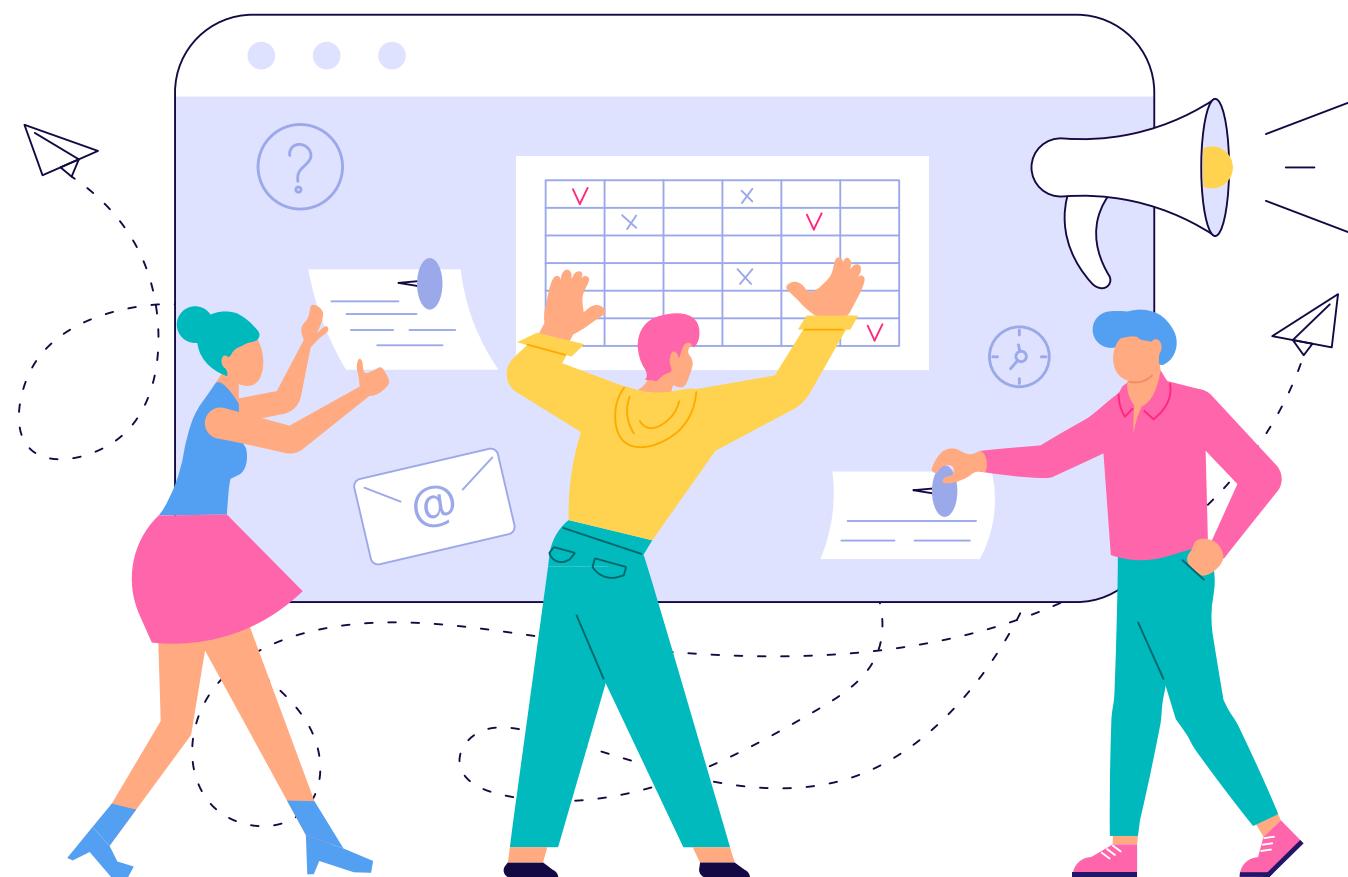
Shivay Lamba

CTO @ DarkHorse
MLH Fellow
GSoC Mentor
Open-source contributor



Hosted by
 Metabob

Agenda



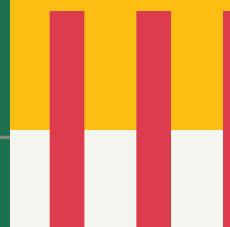
- **10:00-10:10 | Kickoff**
 - Introduction
 - Chit Chat
- **10:10-11:10 | Presentation**
 - Speaker
- **11:10-11:15 | Break**
 - Pre Q&A
- **11:15-11:30 | Q&A**
 - Discussion
- **11:30-11:35 | Wrap Up**
 - Contact Speaker

Hosted by



Axel Lönnfors
Digital Marketing
Specialist | Community
Manager

*Enthusiastic sports fan originally from Finland,
passionate about technologies enhancing employee
productivity*



Metabob

It's the fast, easy, and visual way of
debugging code.

<https://metabob.com>

GOUP

Community driven Open source accelerator!
<https://goupaz.com>

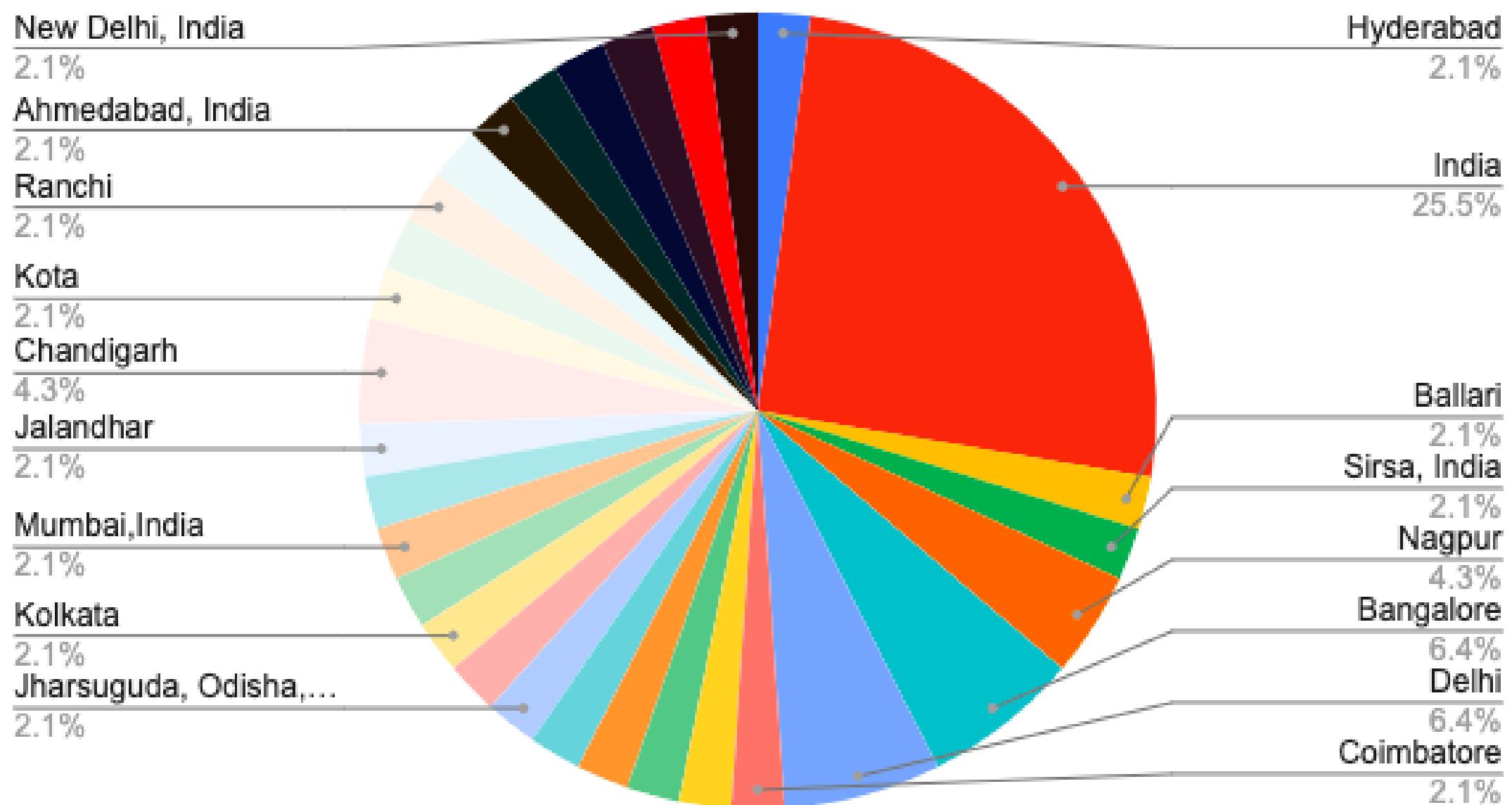
Are you ready?

Let's Begin!

CHITCHAT

Audience

Today's Locations



Code of Conduct

- 1 Learn, benefit, contribute
- 2 No marketing, selling, competing
- 3 Equality despite roles & bg



Photo Shoot Time

Please turn on your camera :D

Presentation

Superpowers For Next Gen Web Apps : ML/AI For Web



Shivay Lamba

TensorFlow.js SIG & Working Group Member
TFUG New Delhi Mentor, Google Code In Mentor for TensorFlow

@howdevelop on Twitter!



Machine Learning in JavaScript

Why?

What is Tensorflow :

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

Use ML anywhere JavaScript can run



Browser



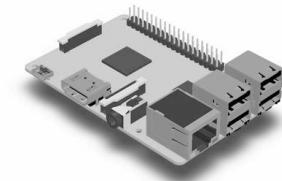
Server



Desktop



Mobile



IoT



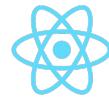
Browser



Server



Mobile



React Native



WeChat

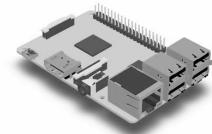
PWA



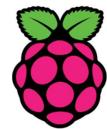
Desktop



Electron



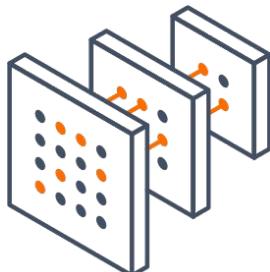
IoT



RaspberryPi
(via Node)

Run, Retrain, Write

Reuse existing models, or create your own



Run existing models

Pre-packaged JavaScript or
Converted from Python



Retrain existing models

With transfer learning



Write models in JS

Train from scratch

For anything you may dream up

Augmented Reality

Gesture-based interaction

Sound recognition

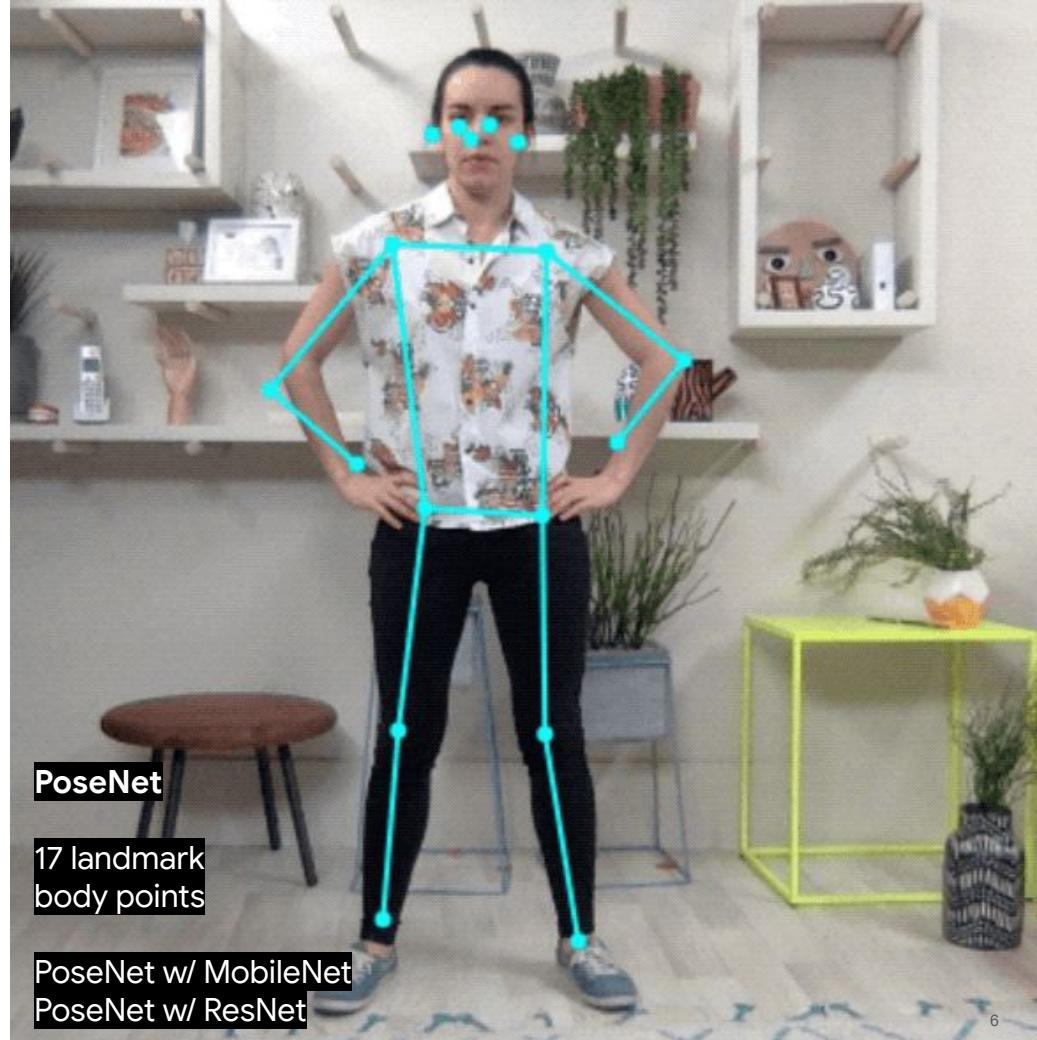
Accessible web apps

Sentiment analysis, abuse detection, NLP

Conversational AI

Web-page optimization

And much more...



Pre trained models

Easy to use JavaScript classes for common use cases

We have several...

And continually expanding our collection.

+ Image classification



+ Object detection



+ Body Segmentation



+ Pose Estimation



New models added:

- + Face Mesh
- + Hand Pose
- + BERT Q&A

+ Text Toxicity



+ Sentence encoding



+ Speech Commands



+ KNN Classifier



Let's check some out...

tensorflow.org/js/models

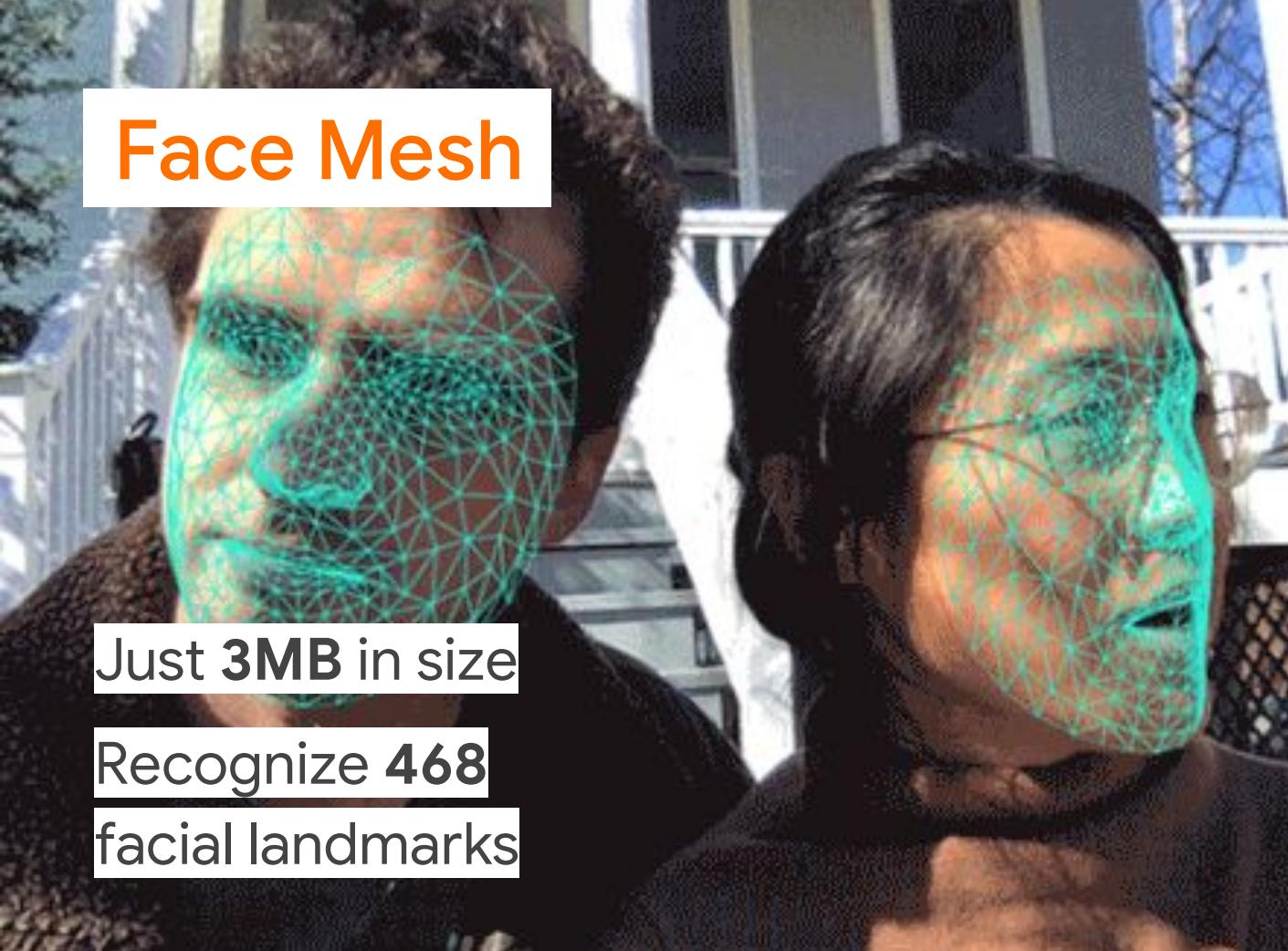
Object Recognition

Using COCO-SSD

Trained on 90 object classes

[Demo](#)

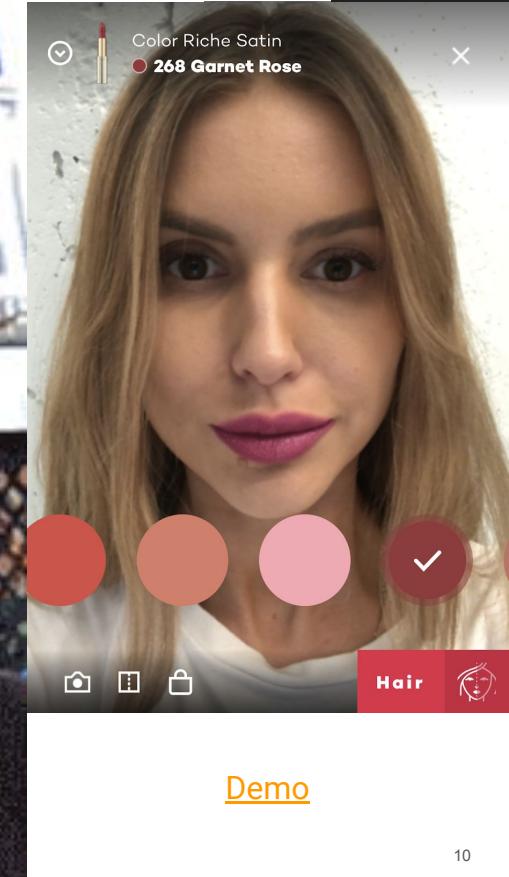




L'ORÉAL

Face Mesh

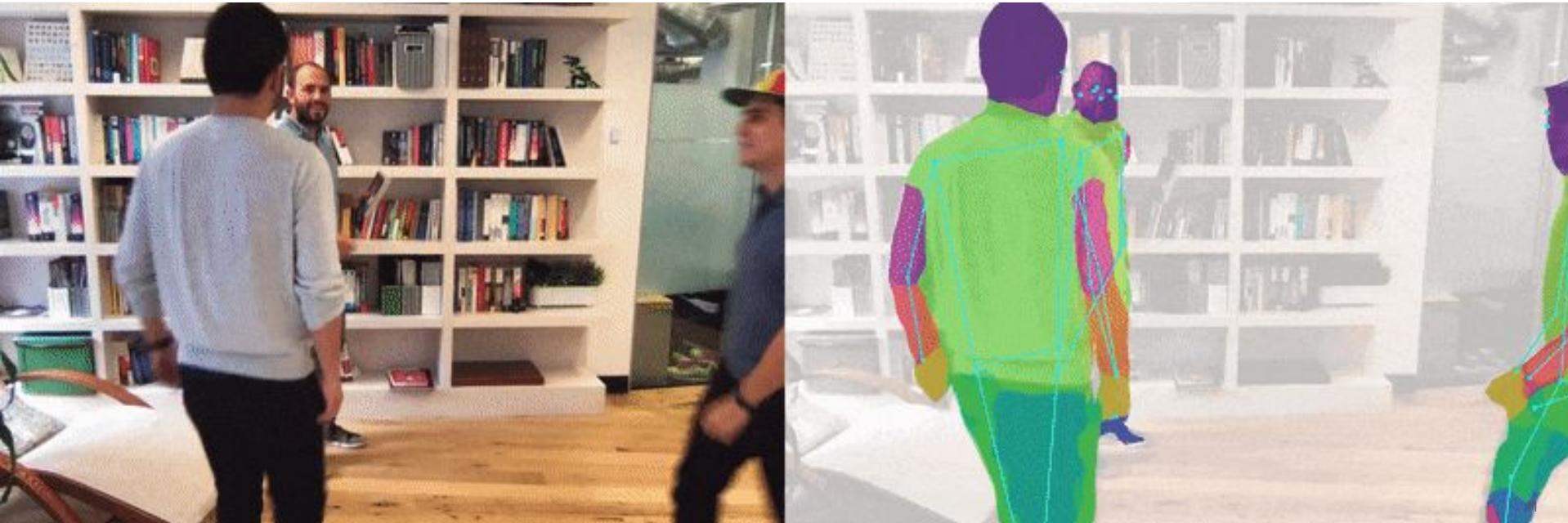
Just 3MB in size
Recognize 468
facial landmarks



MODI
FACE

Body Segmentation

Distinguish **24** body areas, across multiple bodies, real time.



With a little bit of creativity...

We can emulate the superpowers we were promised
in the movies which we should have by now!

Lasers

WebGL Shaders + TensorFlow.js



Teleportation

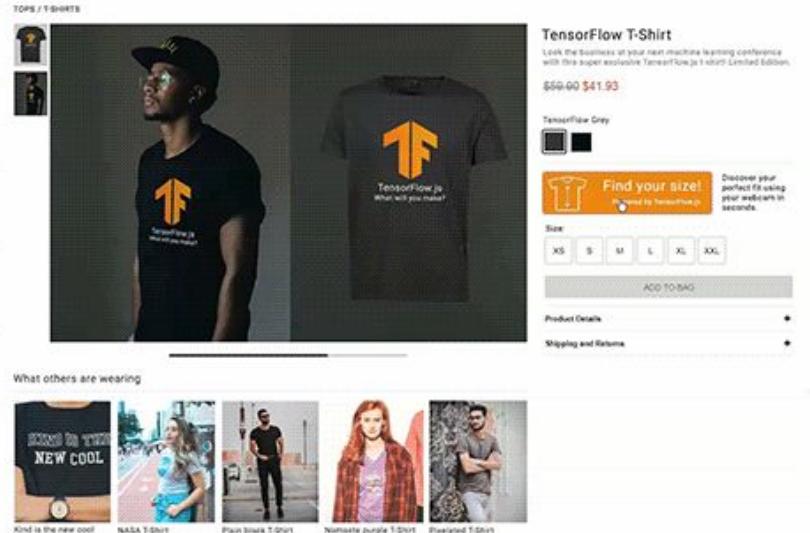
WebRTC + AFrame +
Three.js + TensorFlow.js

Created by Jason Mayes (@jason_mayes)



Or other delightful creations

Such as clothing size estimation



Created by [Jason Mayes \(@jason_mayes\)](#)

Combine with other web tech



Web XR + WebGL
+ TensorFlow.js

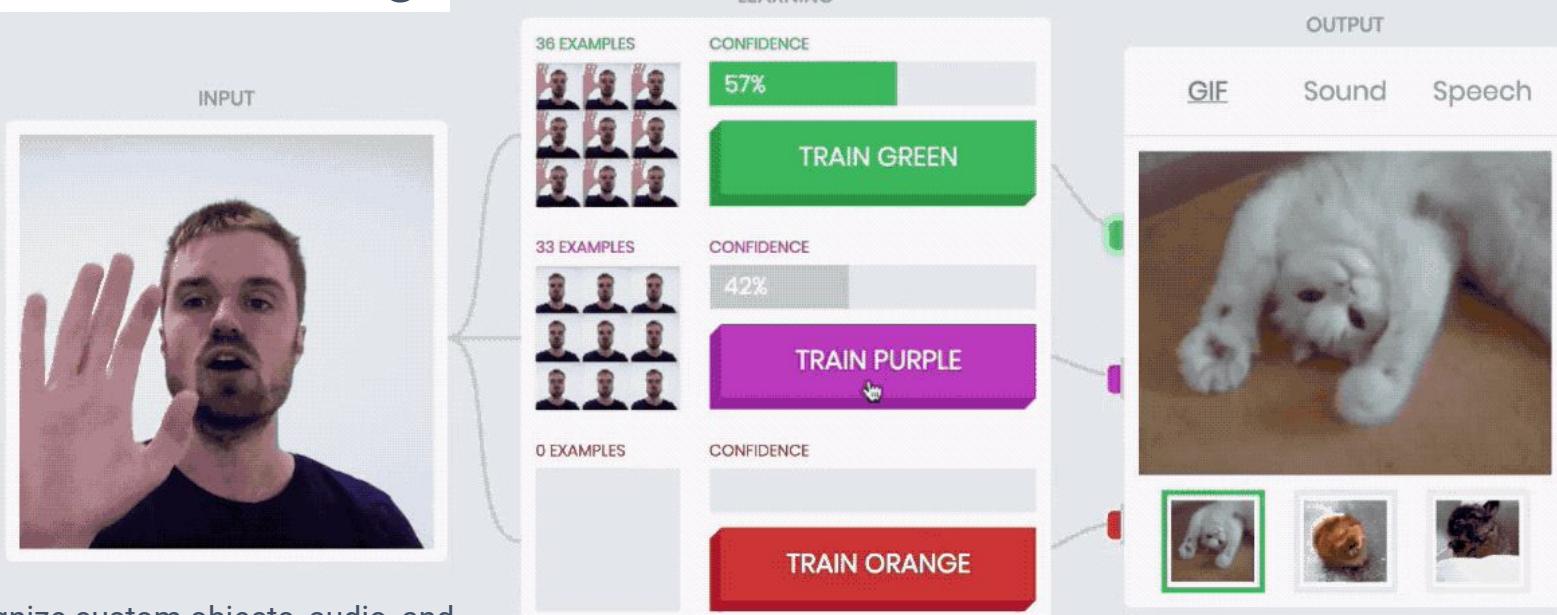
Created by [Alexandre Devaux](#) (@AlexandreDevaux), France

Transfer Learning

Retrain existing models to work with your own data

Teachable Machine

Transfer Learning



Recognize custom objects, audio, and poses in the browser in minutes. [Demo](#).

Cloud Auto ML

Train custom vision models in the cloud + deploy to TF.js

← flowers **BETA** LABEL STATS EXPORT DATA

| IMPORT | IMAGES | TRAIN | EVALUATE | TEST & USE | Single-Label Classification |
|---------------|--------|-------|----------|------------|--|
| All images | 3,667 | | | | Filter images |
| Labeled | 3,667 | | | | tulips(1) dandelion(1) sunflowers(1) dandelion(1) dandelion(1) |
| Unlabeled | 0 | | | | |
| Filter labels | ⋮ | | | | |
| daisy | 633 | | | | tulips(1) dandelion(1) sunflowers(1) dandelion(1) dandelion(1) |
| dandelion | 898 | | | | tulips(1) dandelion(1) tulips(1) daisy(1) sunflowers(1) |
| roses | 640 | | | | |
| sunflowers | 697 | | | | |
| tulips | 799 | | | | |

Cloud Auto ML

The screenshot shows the Google Cloud Platform Vision interface. On the left, there's a sidebar with 'Vision', 'Dashboard', 'Datasets', and 'Models'. The main area has tabs for 'IMPORT', 'IMAGES', 'TRAIN', 'EVALUATE', and 'TEST & USE'. The 'TRAIN' tab is selected, showing a project named 'flowers' (BETA). Below it, a summary card displays 'Average precision 0.995'. Underneath, detailed model information is listed:

| Model ID | ICN7918860300386312098 |
|------------------|--------------------------|
| Created | Sep 1, 2019, 11:55:41 AM |
| Base model | None |
| Data | 3,667 images |
| Model type: | Mobile High Accuracy |
| Budget | 5 compute hours |
| Deployment state | Deployed |

At the bottom, there are buttons for 'SEE FULL EVALUATION' and 'RESUME TRAINING'.

Train new model

① Define your model

② Optimize model for

| Goal | Package size | Accuracy | Latency for Google Pixel 2 |
|--|--------------|----------|----------------------------|
| <input checked="" type="radio"/> Higher accuracy | 2.8 MB | Higher | 360 ms |
| <input type="radio"/> Best trade-off | 2.8 MB | Medium | 150 ms |
| <input type="radio"/> Faster predictions | 2.8 MB | Lower | 56 ms |

Please note that prediction latency estimates are for guidance only. Actual latency will depend on your network connectivity.

[CONTINUE](#)

③ Set a node hour budget

[START TRAINING](#) [CANCEL](#)

Cloud Auto ML

The screenshot shows the Cloud Auto ML interface with the 'TEST & USE' tab selected. At the top, there are tabs for IMPORT, IMAGES, TRAIN, EVALUATE, TEST & USE, LABEL STATS, and EXPORT DATA. Below these, a dropdown menu shows the model name 'untitled_1569280095200'. A note says 'To use online prediction, deploy your model to the cloud. Deployed model charges are per hour and number of machines used.' A 'Pricing guide' link is provided. The main area is titled 'Use your model' and lists several export options:

- TF Lite: Export your model as a TF Lite package to run your model on Android and iOS.
- Container: Export your model as a TensorFlow package to run your model on a Docker container.
- Core ML: Export a .mlmodel file to run your model on iOS and macOS devices.
- Edge devices: Export your model as a TensorFlow package to get faster predictions on your CPU, GPU, or TPU-based edge devices.
- TensorFlow.js**: Export your model as a TensorFlow.js package to run your model in the browser and in Node.js. This option is highlighted with a red circle.
- Coral: Export your model to run on Edge TPU-based devices.
- REST API: Use a REST API to get predictions from this model through Google Cloud.
- Python: Use a Python client to make predictions from the model.

Export TensorFlow.js package

You can use your model in a web browser or on node.js using the TensorFlow.js JavaScript library.

1. Export your model as a TensorFlow.js package.

Destination folder on Cloud Storage

EXPORT

2. After your model finishes exporting, you can copy your package to your computer using this command:

```
$ gsutil cp -r gs://target ./download_dir
```

3. Follow the quickstart to learn how to implement your model. [Web quickstart](#) [Node.js quickstart](#)

Code

```
<script src="//cdn.jsdelivr.net/npm/@tensorflow/tfjs/dist/tf.min.js"></script>
<script src="//cdn.jsdelivr.net/npm/@tensorflow/tfjs-automl/dist/tf-automl.min.js"></script>



<script>
  async function run() {
    const model = await tf.automl.loadImageClassification('model.json');
    const image = document.getElementById('daisy');
    const predictions = await model.classify(image);
  }
  run();
</script>
```

TensorFlow.js: Write your own code

Super powers and performance

TensorFlow.js APIs

Create your own models
with our APIs

Pick your flavour:

- High level **Layers API** (like Keras)
- Low level **Ops API** (mathematical)

Layers

Advanced Activation
`tf.layers.elu`
`tf.layers.leakyReLU`
`tf.layers.prelu`
`tf.layers.relu`
`tf.layers.softmax`
`tf.layers.thresholdedReLU`

Basic

`tf.layers.activation`
`tf.layers.dense`
`tf.layers.dropout`
`tf.layers.embedding`
`tf.layers.flatten`
`tf.layers.permute`
`tf.layers.repeatVector`
`tf.layers.reshape`
`tf.layers.spatialDropout1d`

Convolutional

`tf.layers.conv1d`
`tf.layers.conv2d`
`tf.layers.conv2dTranspose`
`tf.layers.conv3d`
`tf.layers.cropping2d`
`tf.layers.depthwiseConv2d`
`tf.layers.separableConv2d`
`tf.layers.upSampling2d`

Merge

`tf.layers.add`
`tf.layers.average`
`tf.layers.concatenate`
`tf.layers.dot`
`tf.layers.maximum`
`tf.layers.minimum`
`tf.layers.multiply`

Normalization

`tf.layers.batchNormalization`
`tf.layers.layerNormalization`

Pooling

`tf.layers.averagePooling1d`
`tf.layers.averagePooling2d`
`tf.layers.averagePooling3d`
`tf.layers.globalAveragePooling1d`
`tf.layers.globalAveragePooling2d`
`tf.layers.globalMaxPooling1d`
`tf.layers.globalMaxPooling2d`

Tensors

Tensors are the core datastructure of TensorFlow.js They are a generalization of vectors and matrices to potentially higher dimensions.

Tensors / Creation

We have utility functions for common cases like Scalar, 1D, 2D, 3D and 4D tensors, as well as a number of functions to initialize tensors in ways useful for machine learning.

`tf.tensor` (`values`, `shape?`, `dtype?`)

[source](#)

Creates a `tf.Tensor` with the provided `values`, `shape` and `dtype`.

```
// Pass a array of values to create a vector.  
tf.tensor([1, 2, 3, 4]).print();
```

[Edit](#) [Run](#)

```
// Pass a nested array of values to make a matrix or a higher  
// dimensional tensor.  
tf.tensor([[1, 2], [3, 4]]).print();
```

[Edit](#) [Run](#)

```
// Pass a flat array and specify a shape yourself.  
tf.tensor([1, 2, 3, 4], [2, 2]).print();
```

[Edit](#) [Run](#)

Parameters:

`values` (`TypeArray`|`Array`) The values of the tensor. Can be nested array of numbers, or a flat array, or a `TypeArray`. If the values are strings, they will be encoded as utf-8 and kept as `Uint8Array`.

`shape` (`number[]`) The shape of the tensor. Optional. If not provided, it is inferred from `values`. [Optional](#)

`dtype` ('`float32`'|'`int32`'|'`bool`'|'`complex64`'|'`string`') The data type. [Optional](#)

Returns: `tf.Tensor`

`tf.scalar` (`value`, `dtype?`)

[source](#)

Creates rank-0 `tf.Tensor` (scalar) with the provided `value` and `dtype`.

The same functionality can be achieved with `tf.tensor()`, but in general we recommend using `tf.scalar()` as it makes the code more readable.

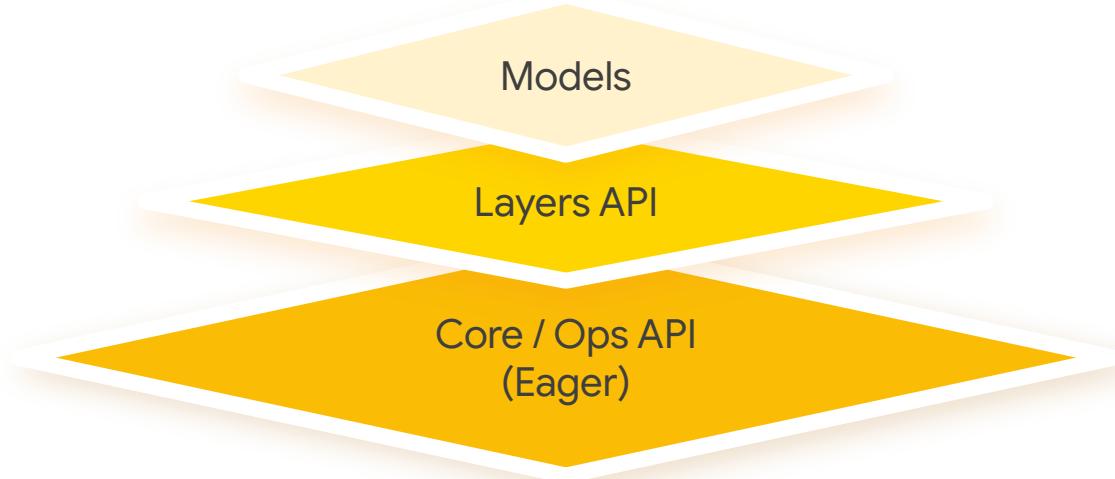
```
tf.scalar(3.14).print();
```

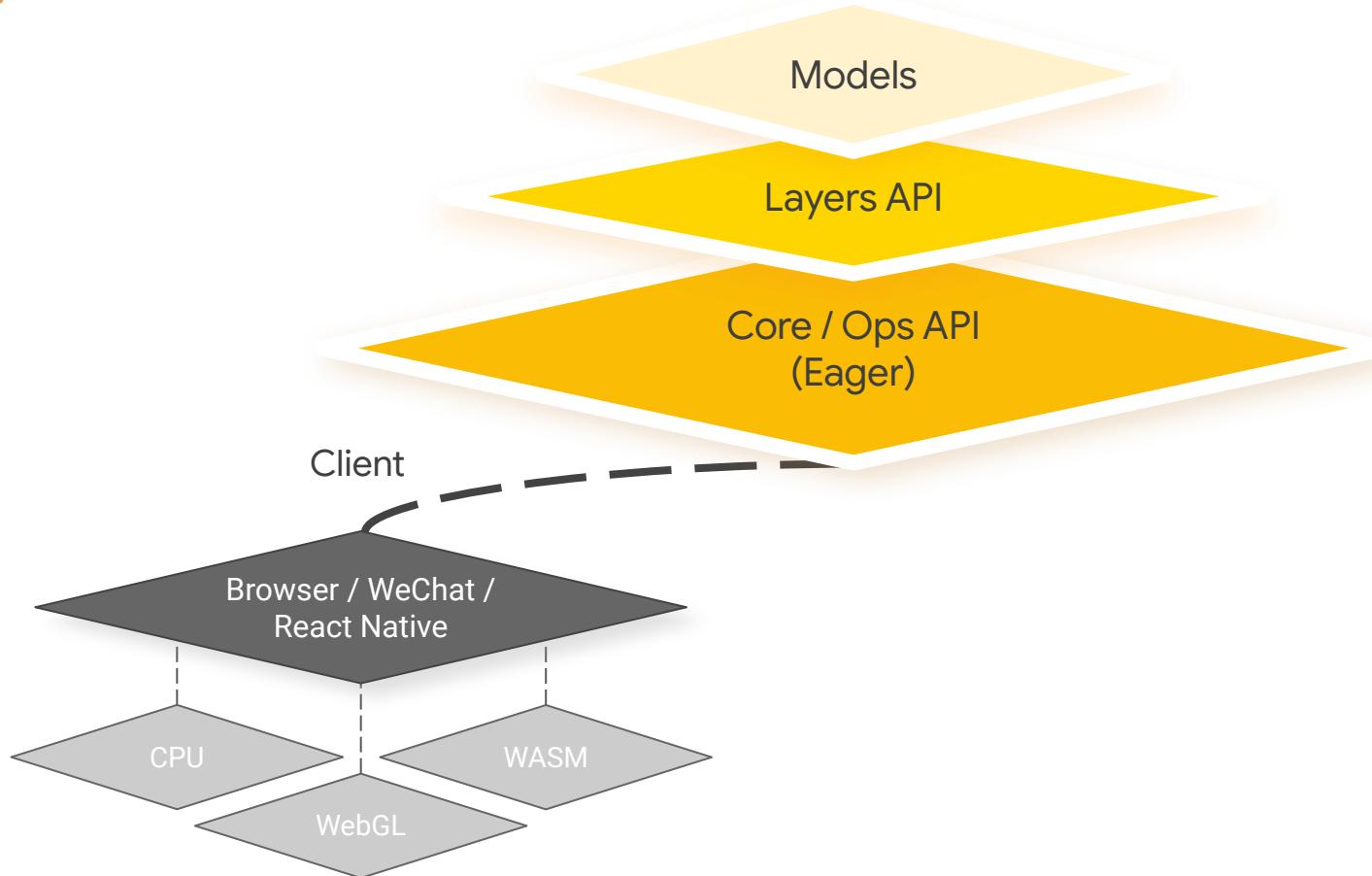
[Edit](#) [Run](#)

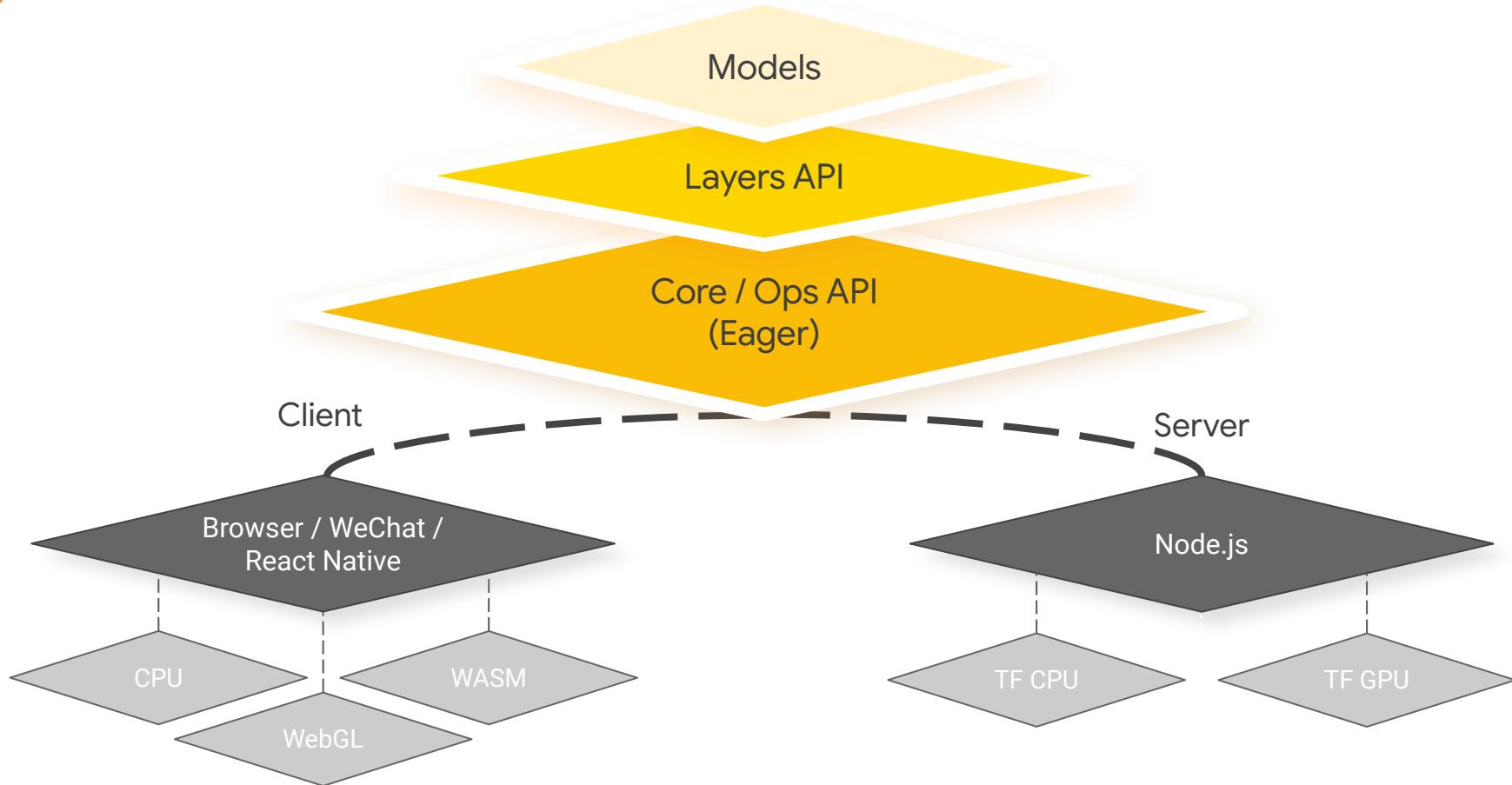
Parameters:

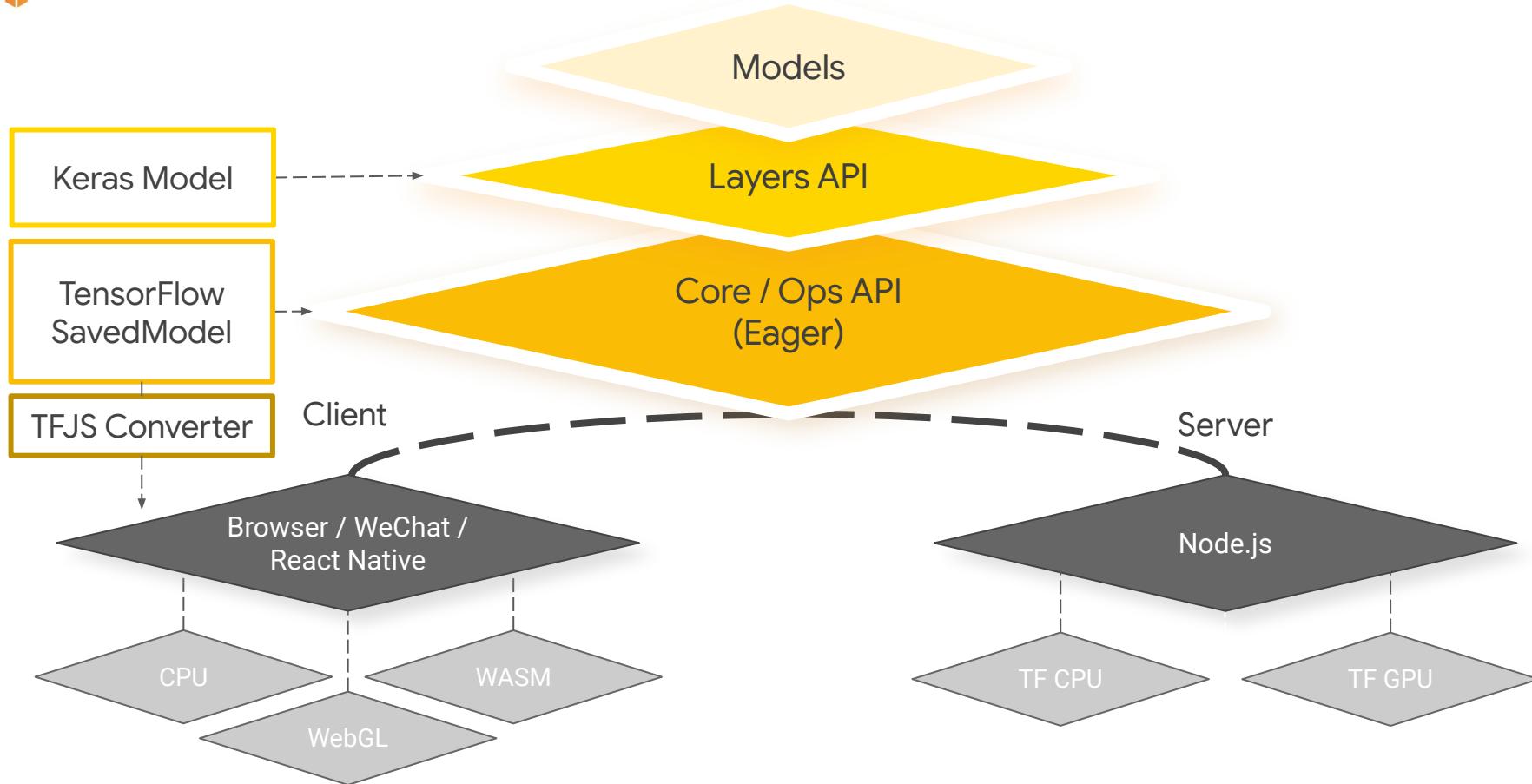
`value` (`number`|`boolean`|`string`|`Uint8Array`) The value of the scalar.

24



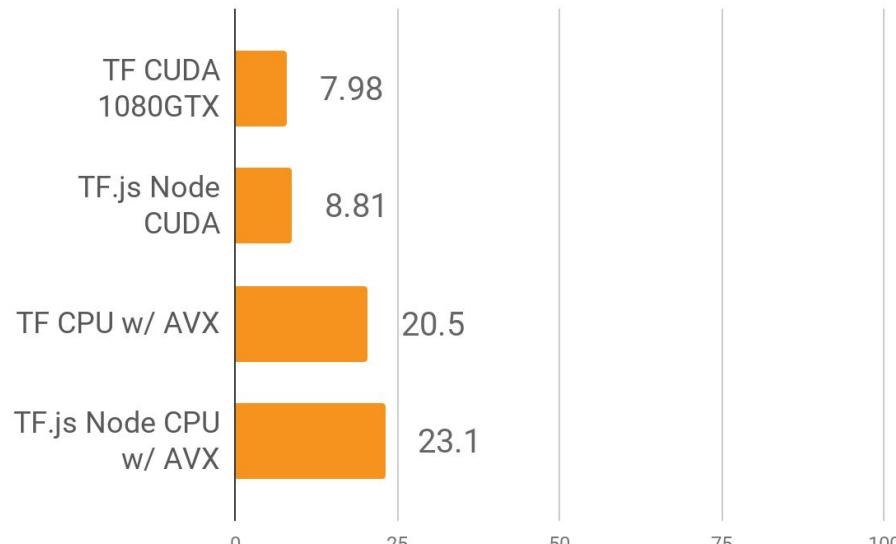






Model Inference Performance Only

Server

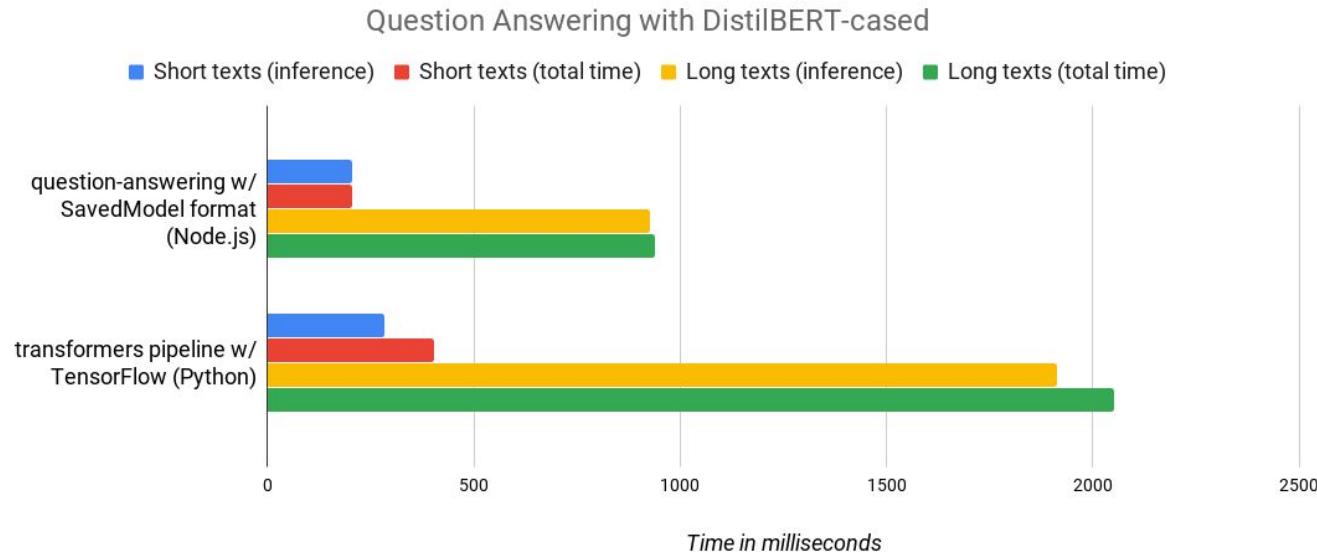


MobileNetV2 1.0_224 model inference time (ms)

Note: If you have a lot of pre / post processing written in Node.js, you will get the benefits of the JIT at runtime, which can be significant outside of inference alone.

Hugging Face DistilBERT

2x Perf boost using Node.js

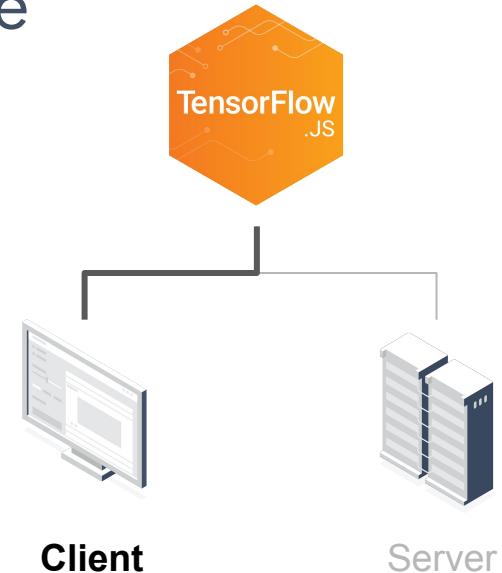


5 client side super powers

Harder / impossible to achieve server side

1. Privacy
2. Lower Latency
3. Lower Cost
4. Interactivity
5. Reach and Scale

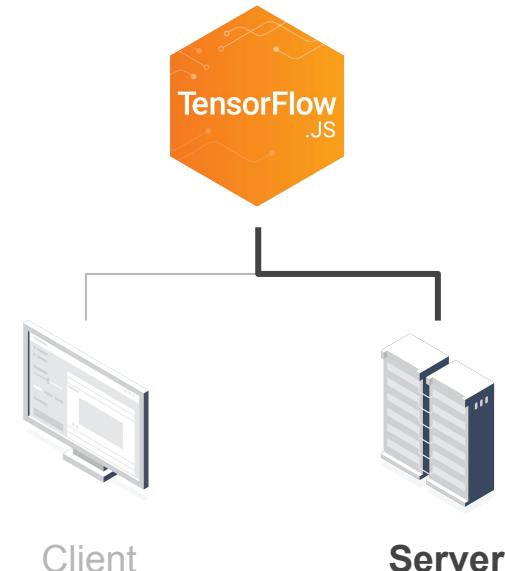
(GPU support on [84%](#) of devices via WebGL)



Server side / Node.js benefits

That make TensorFlow.js easy to use

1. Use TensorFlow SavedModel without conversion
2. Run larger models than client side (GPU memory limits)
3. Code in 1 language - if you already use JS
(67.8% of people use JS in development already)
4. Performance - C bindings / JIT boost for pre/post processing



Resources

Learn more and get inspired

Learn more

Get started fast!

Website / API: tensorflow.org/js

Models: tensorflow.org/js/models

Github Code: github.com/tensorflow/tfjs

Google Group: tfjs@tensorflow.org

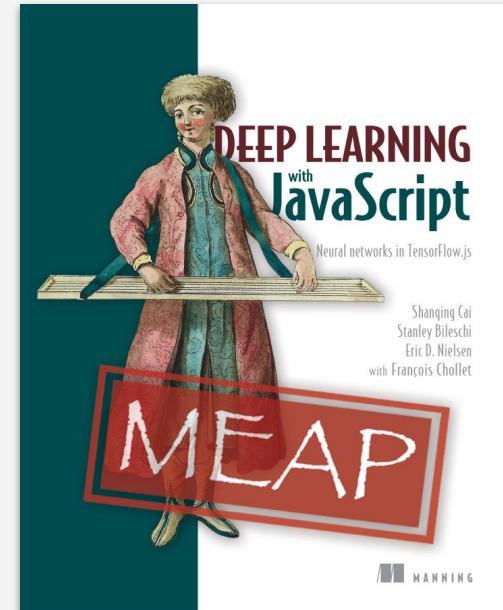
Codepen: codepen.io/topic/tensorflow

Glitch: glitch.com/@TensorFlowJS

Recommended Reading

Deep Learning with JavaScript

manning.com/books/deep-learning-with-javascript





#MadeWithTFJS

Thanks to the global community for use of videos shown on this slide: Jimmy (@MLBlock3), Manish Raj (@manishr41883690), Ben Farrell (@bfarrellforever), Wen (@yiwen_lin), Junya Ishihara (@jishiha), Rogerio Chaves (@_rchaves_), Amruta (@prenalys), FollowTheDarkside (@eatora22), and Alexandre Devaux (@AlexandreDevaux).

What will you make?



Machine Learning
for everyone.

#MadeWithTFJS

Video by [FollowTheDarkside](#), (@eatora22), Japan

Stay in touch



@howdevelop



linkedin.com/in/shivaylamba

#MadeWithTFJS



TensorFlow

Quiz Time :

1. What are the types of APIs provided by TFJS for writing your ownn code?



Quiz Time :

2. What are the current types of models provided by Teachable Machines?



TensorFlow

Quiz Time :

3. Name any two benefits of
running TFJS on Server Side



Quiz Time :

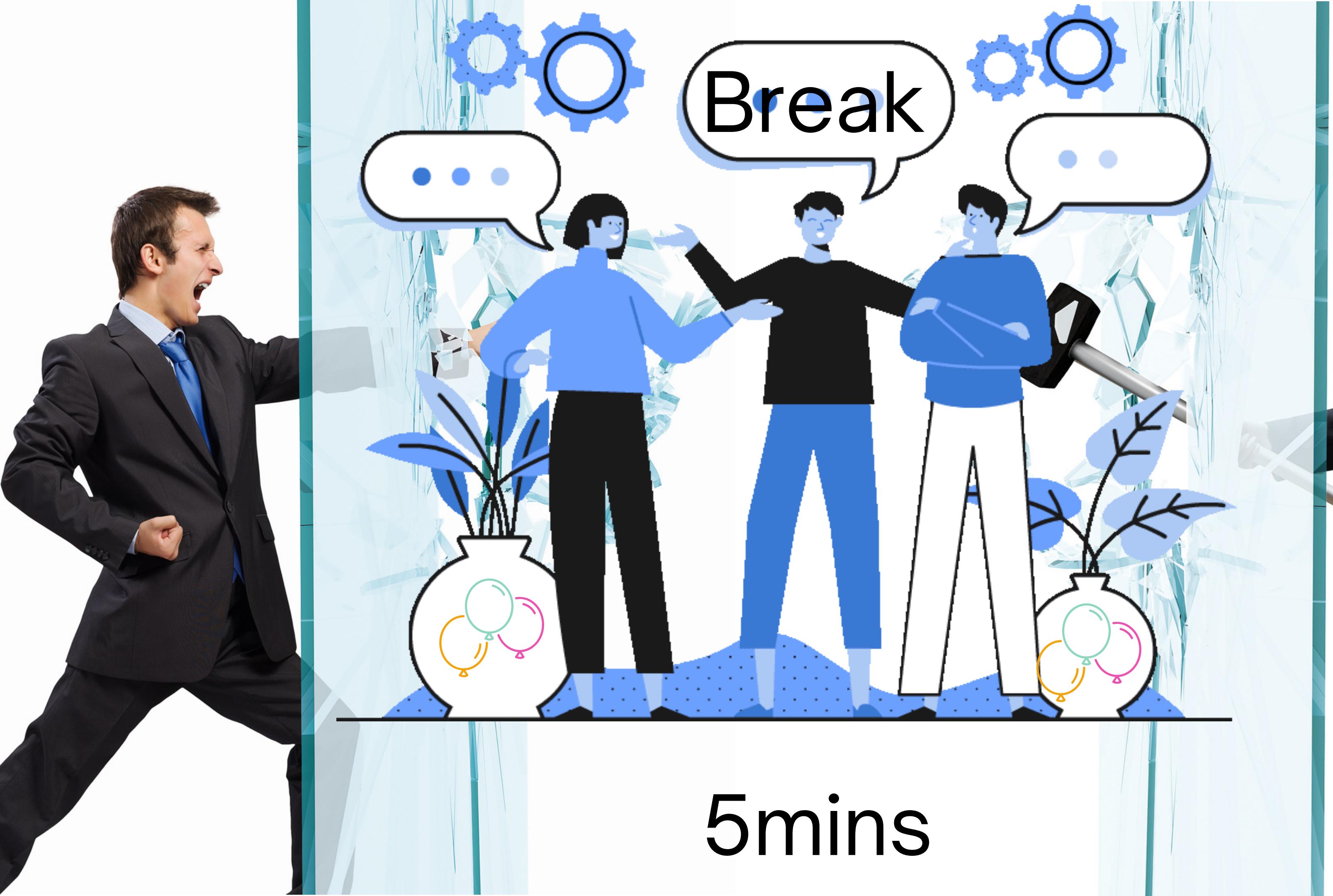
4. For how many classes is the
TFJS COCO-SSD Model for?



Quiz Time :

5. Name a library that has been built on top of Tensorflow.JS? (popular for creative professionals)

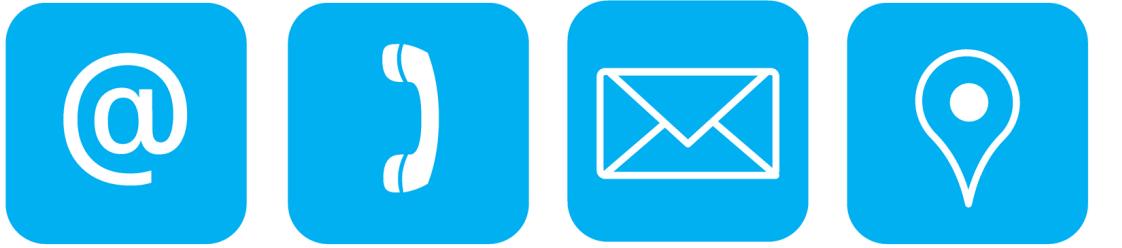




5mins

Wrap Up

HOWTO



Remember!

- Re-Search
- Be short and clear
- Re-mind
- Q&A over Slack

Linkedin: shivaylamba

Email: shivaylamba@gmail.com

GOUP Slack: @shivaylamba





Join Us!

HTTPS://GOUPAZ.COM

HTTPS://METABOB.COM

1 Community Managers

2 Tech Writers

3 In/Out Ambassadors

4 Marketing Creators & Editors

5 Course Creators

6 Project Creators

Thank You

Culture

#egoless #collaborative #competent #decentralized #scalable #fun

Open source

#creator #contributor

Diversity

#age #gender #location #economics #religion #politicalview

How can we do
better?