Introducing TensorFlow:

A framework for machine learning

Department of Computer Science, City University of Hong Kong

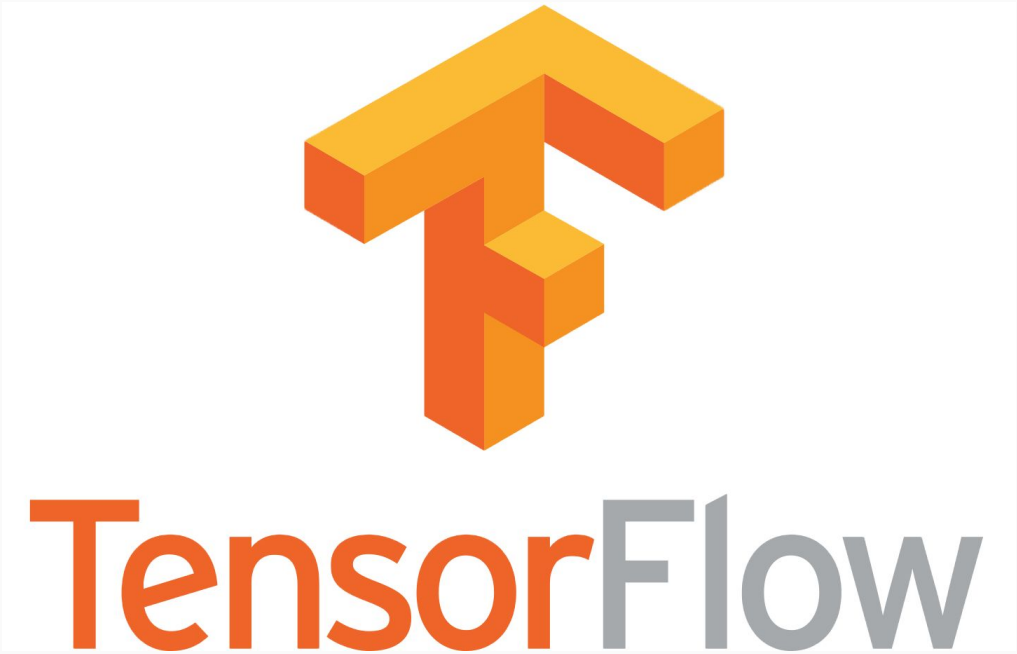
Table of contents

Introduction

Introduction

What is TensorFlow?

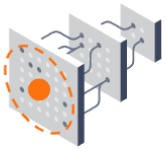
TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.



Introduction

Why Tensorflow?

- Easy to use



Easy model building

Build and train ML models easily using intuitive high-level APIs like Keras with eager execution, which makes for immediate model iteration and easy debugging.

Robust ML production anywhere

Easily train and deploy models in the cloud, on-prem, in the browser, or on-device no matter what language you use.

Powerful experimentation for research

A simple and flexible architecture to take new ideas from concept to code, to state-of-the-art models, and to publication faster.

Introduction

Why Tensorflow?

- Easy to use
- Widely used

Figure 1: Companies using TensorFlow

Introduction

Why Tensorflow?

- Easy to use
- Widely used

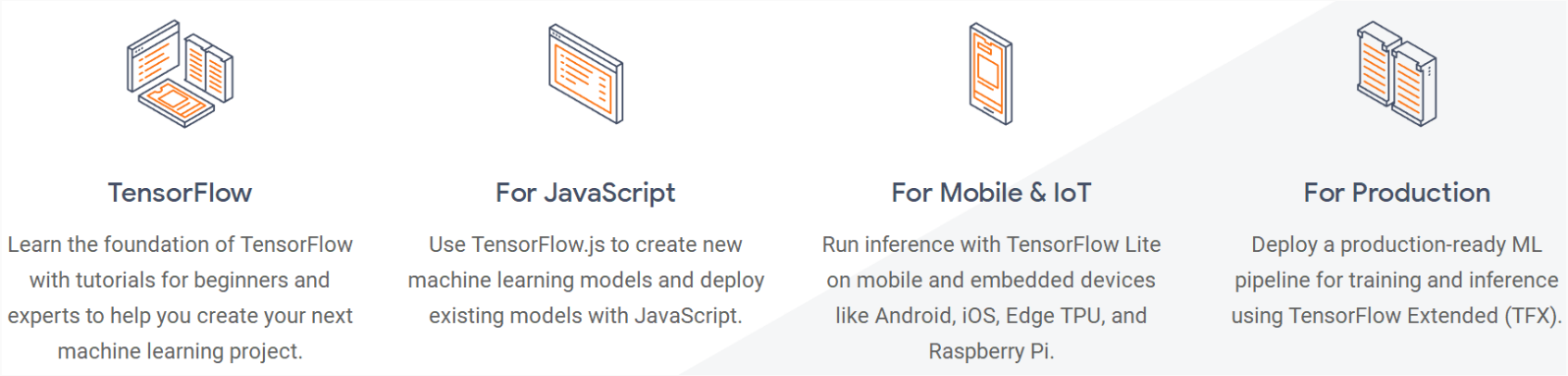
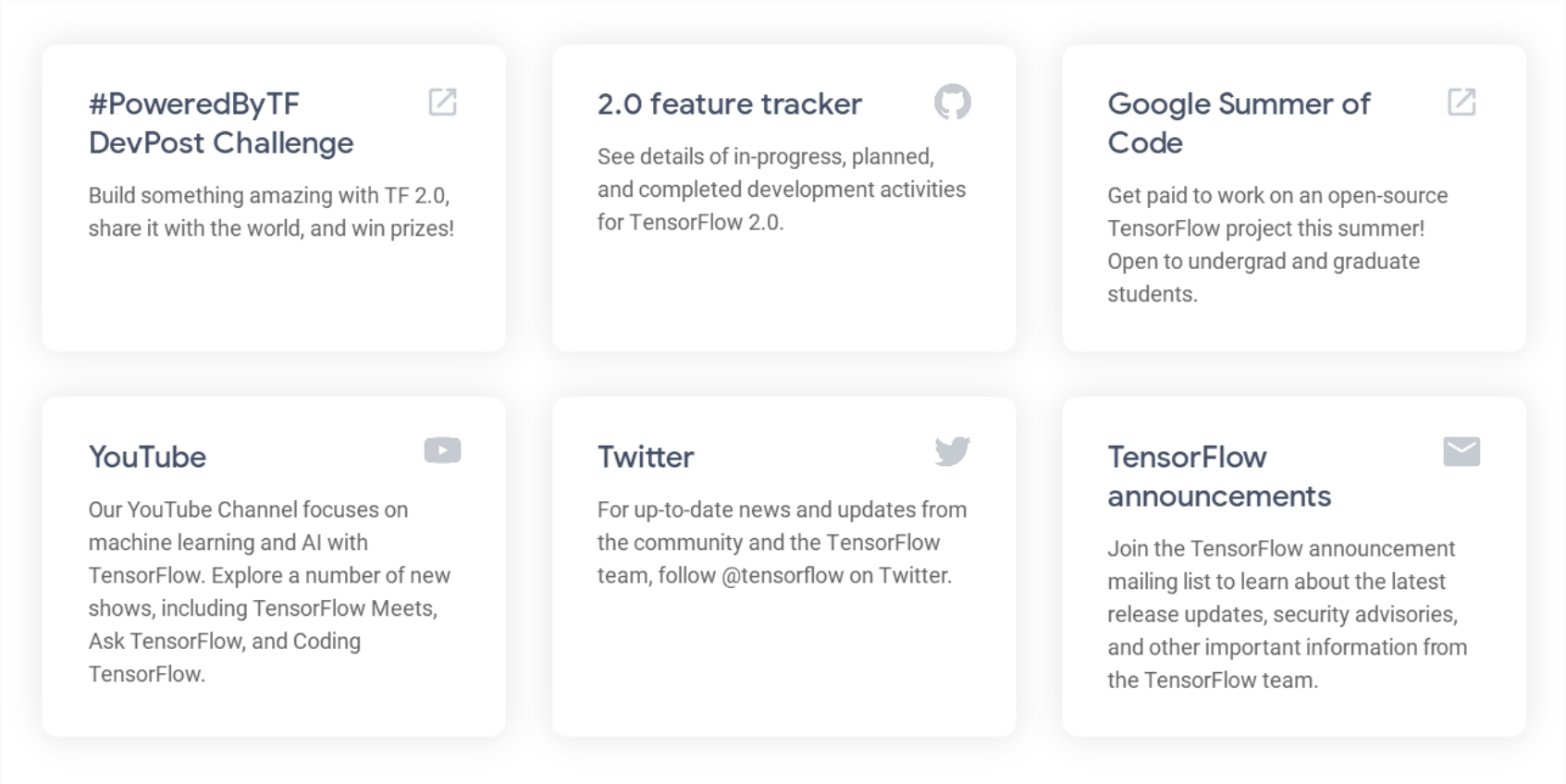


Figure 2: Different platforms of TensorFlow

Introduction

Why Tensorflow?

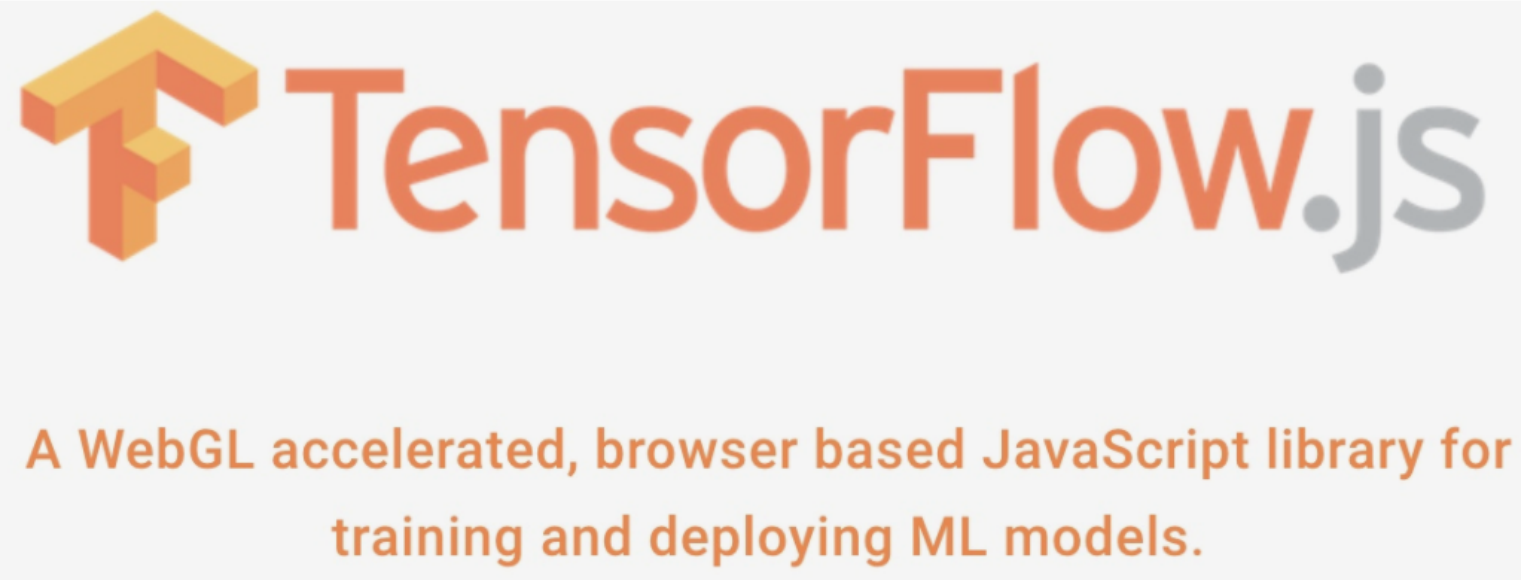
- Easy to use
- Widely used
- Resources and community support



Introduction

What is TensorFlow.js?

TensorFlow.js is a library for developing and training ML models in JavaScript, and deploying in browser or on Node.js



Setup

Setup

There are two main ways to get TensorFlow.js up running in your browser:

- Using script tags (Add this code in your XX.html file)

```
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@1.0.0"></script>
```

- Installation from NPM and using a build tool like Parcel, WebPack, or Rollup.

```
yarn add @tensorflow/tfjs
npm install @tensorflow/tfjs
```

Demo

Demo

Linear regression:

- Objective function:

$$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

  where θ_0 and θ_1 are the parameters that we want to get.

- Loss function:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2,$$

  The loss function is **Mean squared error** (MSE), and we want to find a θ that minimize this loss function.

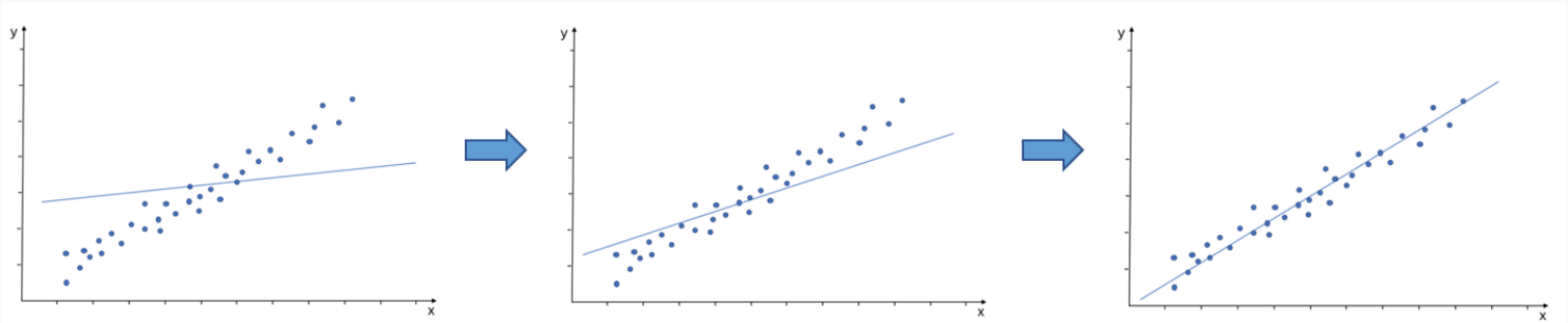
Demo

Linear regression:

- Gradient Descent:

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

  where α is the learning rate (e.g. 0.001). We repeat this process until θ convergence.
- Learning process:
  We randomly initialize a θ and update it using gradient descent method.



Demo

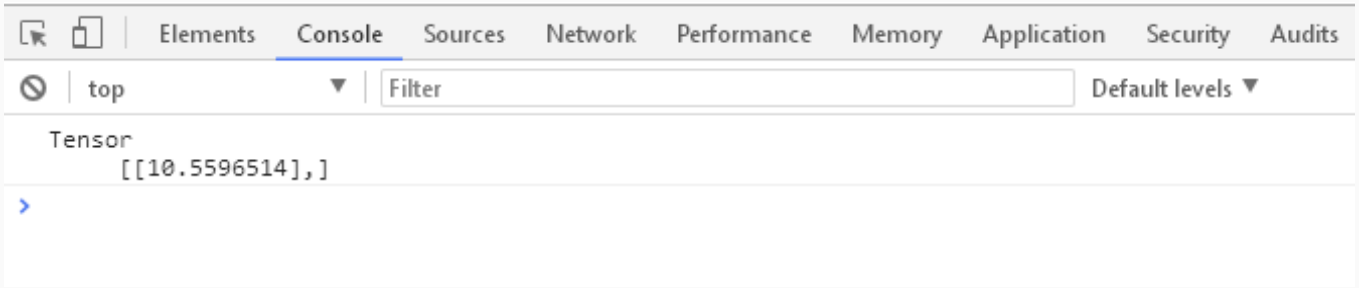
Linear regression:

```html
<html>
  <head>
    <!-- Load TensorFlow.js -->
    <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@1.0.0"> </script>

    <!-- Place your code in the script tag below. You can also use an external .js file -->
    <script>
      // Notice there is no 'import' statement. 'tf' is available on the index-page
      // because of the script tag above.

      // Define a model for linear regression.
      const model = tf.sequential();
      model.add(tf.layers.dense({units: 1, inputShape: [1]}));

      // Prepare the model for training: Specify the loss and the optimizer.
      model.compile({loss: 'meanSquaredError', optimizer: 'sgd'});

      // Generate some synthetic data for training.
      const xs = tf.tensor2d([1, 2, 3, 4, 5], [5, 1]);
      const ys = tf.tensor2d([1, 3, 5, 7, 9], [5, 1]);

      // Train the model using the data.
      model.fit(xs, ys, {epochs: 100}).then(() => {
        // Use the model to do inference on a data point the model hasn't seen before:
        // Open the browser devtools to see the output
        model.predict(tf.tensor2d([6], [1, 1])).print();
      });
    </script>
  </head>

  <body>
  </body>
</html>
```

Demo

Linear regression:

- Save the code above in a .html file (e.g. LRdemo.html), open it with browser, then open devtools (F12-Console) to see the output. (Code can be found in `https://ymq115599.github.io/Demo_LR.html`)



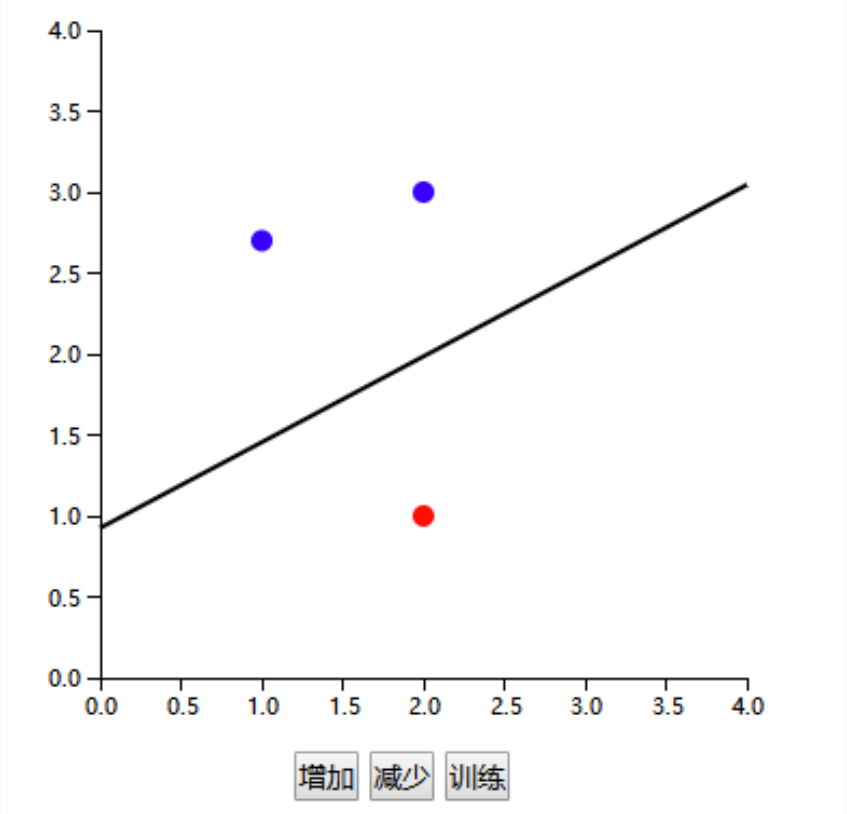
- Change the dataset (xs and ys) and see the difference.

```
18      // Generate some synthetic data for training.
19      const xs = tf.tensor2d([1, 2, 3, 4, 5], [5, 1]);
20      const ys = tf.tensor2d([1, 3, 5, 7, 9], [5, 1]);
```

Demo

Linear regression:

- *Can you visualize the result in webpage? (add codes in $< body >$ part, a demo visualizing Support Vector Machine can be found in `https://ymq115599.github.io/SVM_Demo.html`)



Reference

References

https://tensorflow.google.cn/

https://www.tensorflow.org/js/tutorials/setup

https://js.tensorflow.org/api/latest/

https://www.tensorflow.org/js/guide

Q&A