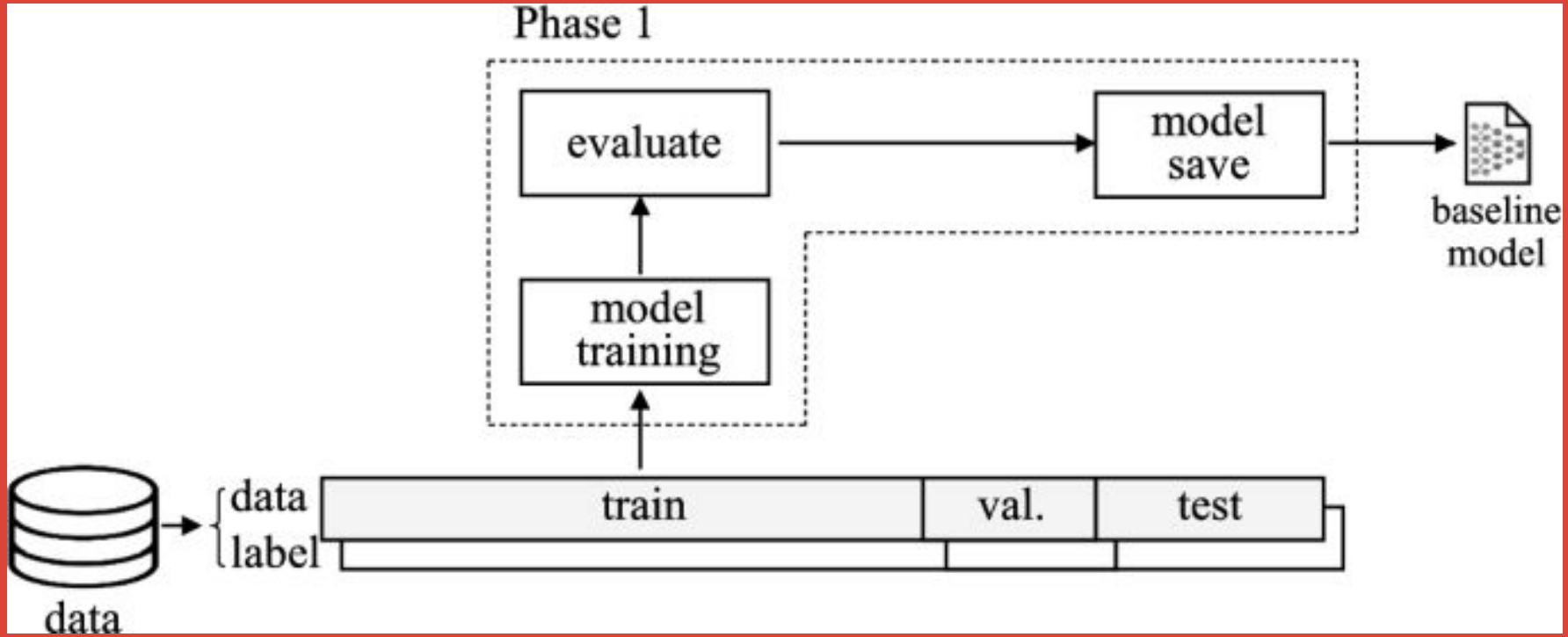
TensorFlow and Code Review

Deep Learning in Remote Sensing

Episode-2

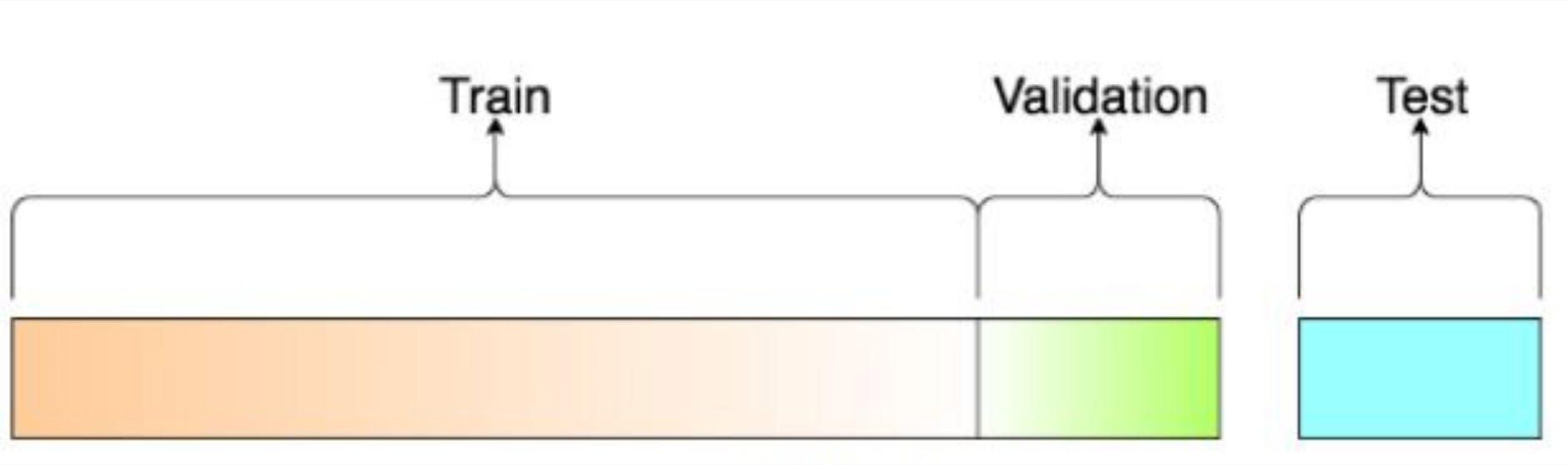
İrem KÖMÜRCÜ
iremkomurcu.com
iremkomurcubm@gmail.com

Create General Baseline Model



Source: Article, Mooseop Kim, Chi Yoon Jeong,
Label-preserving data augmentation for mobile sensor data

Data Visualization of the Splits

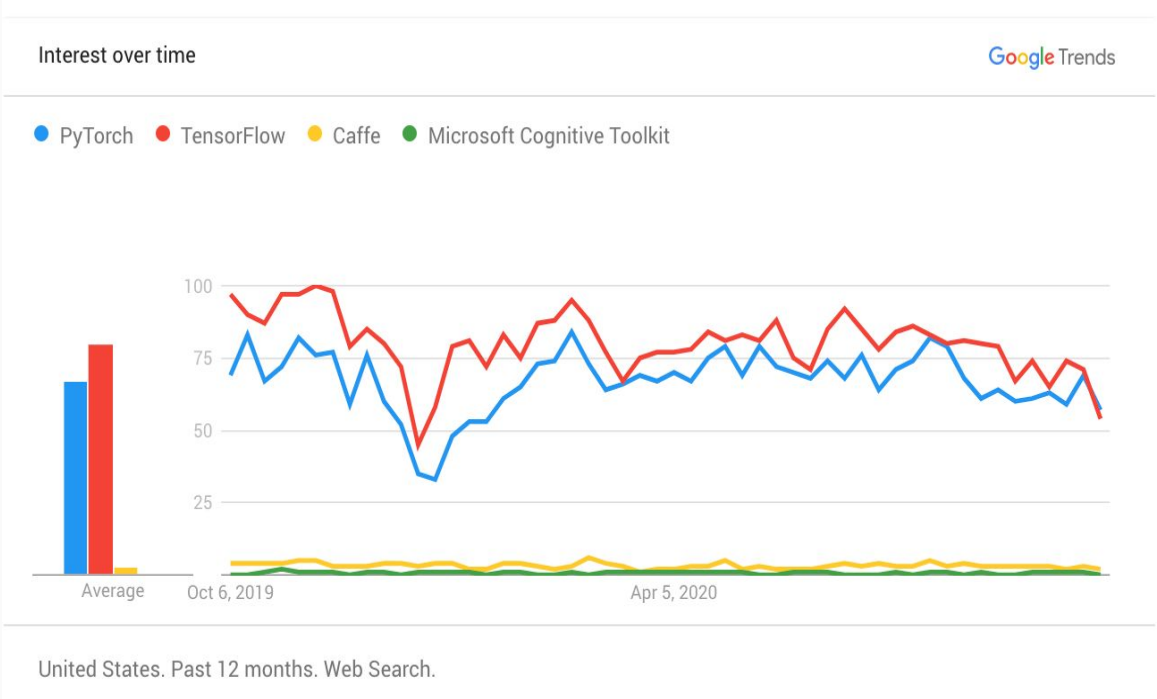


Source: Medium, towardsdatascience, Tarang Shah

- Create, Google Brain Team
- Open Source
- Comprehensive, flexible ecosystem of tools, libraries
- Community Resources
- Easy model building
- Supporting multi language
- Google Cloud Platform, Google Cloud Speech etc

Some **DL Frameworks**

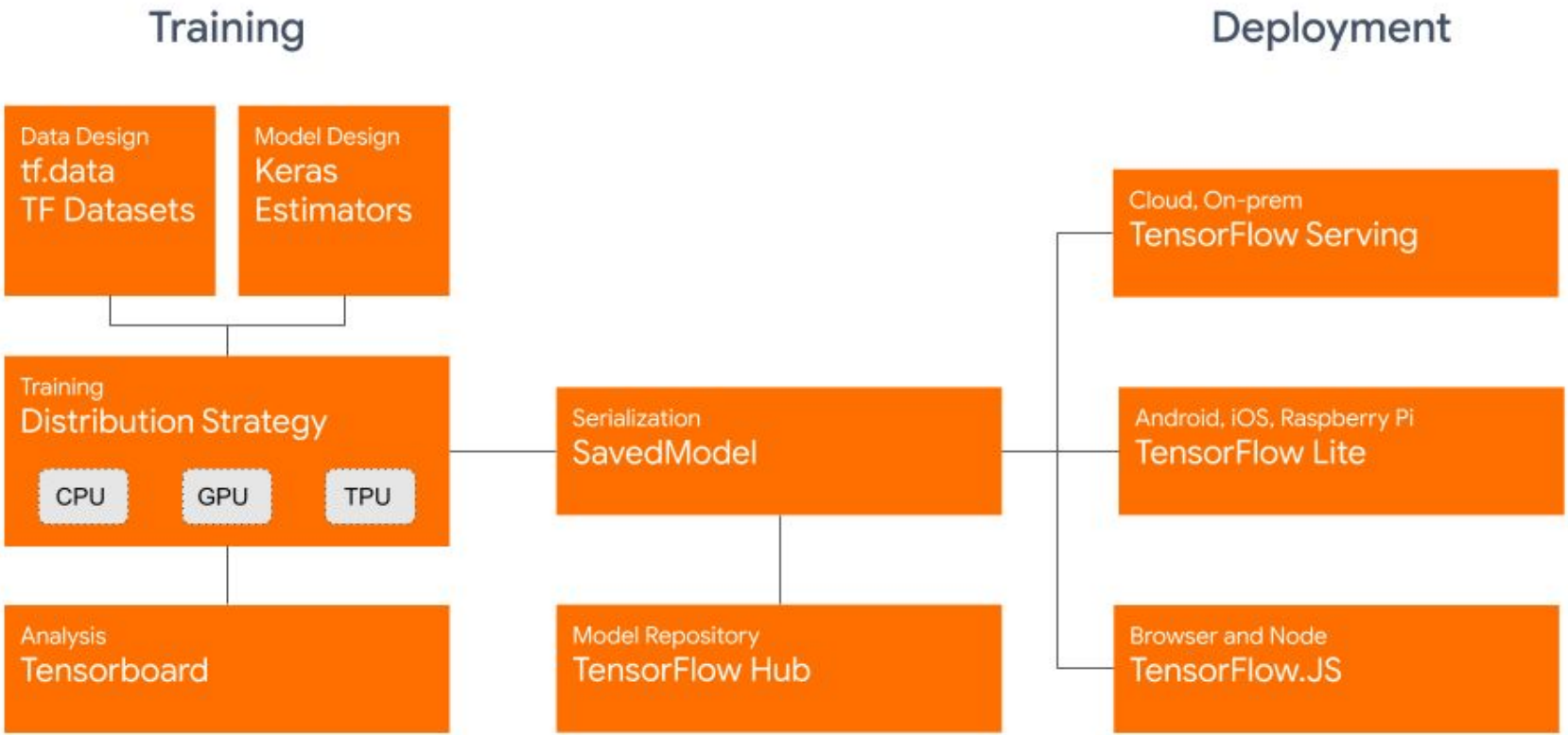
- Tensorflow
- Keras
- Pytorch
- MXNet
- Theano
- Caffe



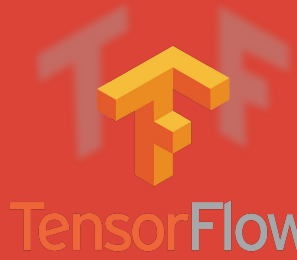
Source: tooploox.com

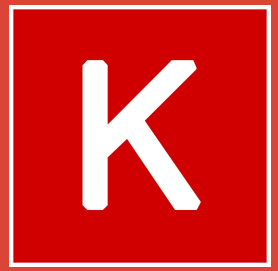
Tensorflow 1.x

Tensorflow 2.x



Source: blog.tensorflow.org

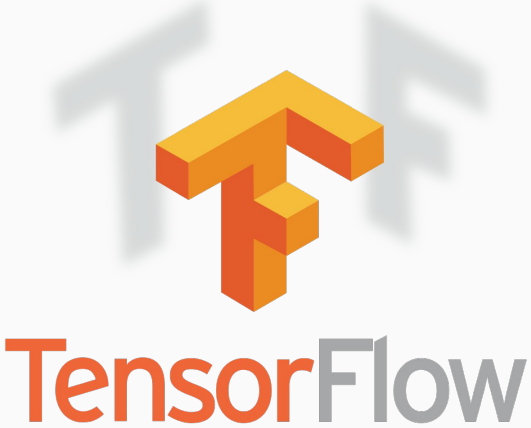


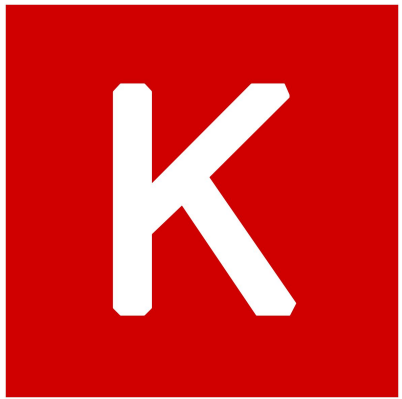



```
import tensorflow as tf

from tensorflow.keras.layers import Dense, Flatten, Conv2D
from tensorflow.keras import Model
```

Tensorflow & Keras

- Data preprocessing
- Model
- Applications
- Layer
- Optimizer
- Activation
- Loss Function
- Callbacks
- Metrics





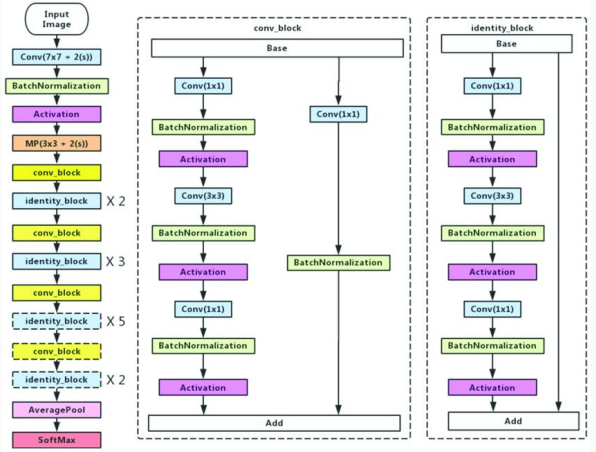
Data Preprocessing

```
tf.keras.preprocessing.image_dataset_from_directory(
    directory,
    labels="inferred",
    label_mode="int",
    class_names=None,
    color_mode="rgb",
    batch_size=32,
    image_size=(256, 256),
    shuffle=True,
    seed=None,
    validation_split=None,
    subset=None,
    interpolation="bilinear",
    follow_links=False,
)
```

```
tf.keras.preprocessing.text.Tokenizer(
    num_words=None,
    filters='!"#$%&()*+,-./:;<=>?@[\\]^_`{|}~\t\n',
    lower=True,
    split=" ",
    char_level=False,
    oov_token=None,
    document_count=0,
    **kwargs
)
```

Tensorflow & Keras **Applications**

- Resnet50
- VGG16
- Xception
- EfficientNet
- Inception_v3
- Densenet
- Mobilenet

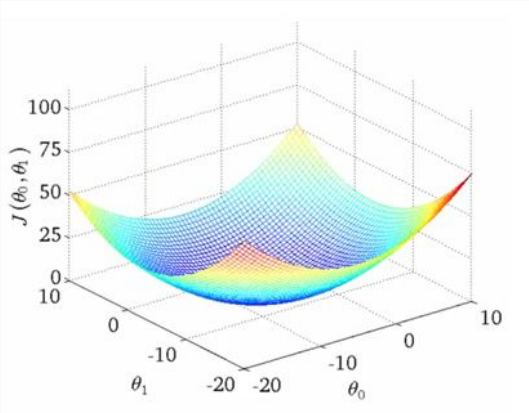


Optimizer

Adam Optimizer Formula

```
for t in range(num_iterations):
    g = compute_gradient(x, y)
    m = beta_1 * m + (1 - beta_1) * g
    v = beta_2 * v + (1 - beta_2) * np.power(g, 2)
    m_hat = m / (1 - np.power(beta_1, t))
    v_hat = v / (1 - np.power(beta_2, t))
    w = w - step_size * m_hat / (np.sqrt(v_hat) + epsilon)
```

Adam.py hosted with ❤ by GitHub view raw




```
tf.keras.optimizers.Adam(
    learning_rate=0.001,
    beta_1=0.9,
    beta_2=0.999,
    epsilon=1e-07,
    amsgrad=False,
    name="Adam",
    **kwargs
)
```

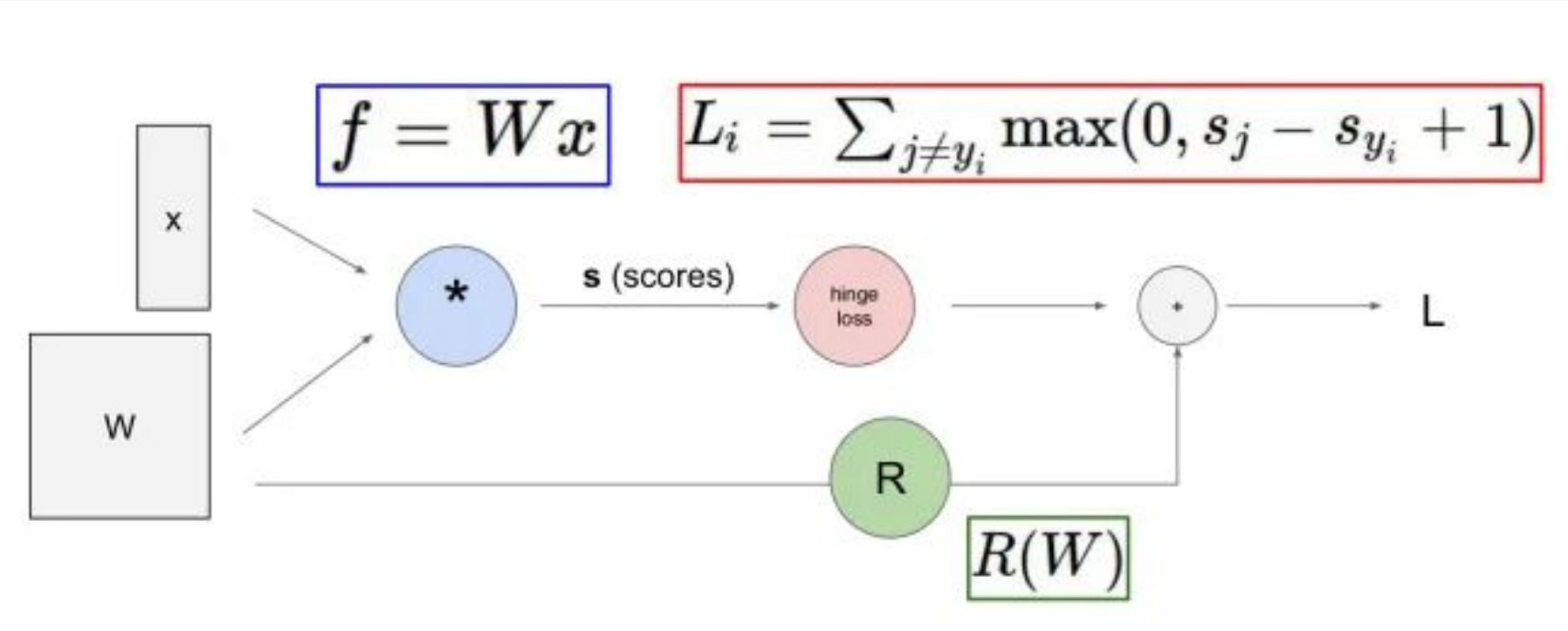
Optimizer

- Adam
- SGD
- RMSprop
- Adadelta
- Adagrad
- Adamax
- ...

```
from tensorflow import keras
from tensorflow.keras import layers

model = keras.Sequential()
model.add(layers.Dense(64, kernel_initializer='uniform', input_shape=(10,)))
model.add(layers.Activation('softmax'))

opt = keras.optimizers.Adam(learning_rate=0.01)
model.compile(loss='categorical_crossentropy', optimizer=opt)
```

```
# pass optimizer by name: default parameters will be used
model.compile(loss='categorical_crossentropy', optimizer='adam')
```

Loss Function



Loss Function

```
from tensorflow import keras
from tensorflow.keras import layers

model = keras.Sequential()
model.add(layers.Dense(64, kernel_initializer='uniform', input_shape=(10,)))
model.add(layers.Activation('softmax'))

opt = keras.optimizers.Adam(learning_rate=0.01)
model.compile(loss='categorical_crossentropy', optimizer=opt)
```

```
# pass optimizer by name: default parameters will be used
model.compile(loss='categorical_crossentropy', optimizer='adam')
```

Callbacks

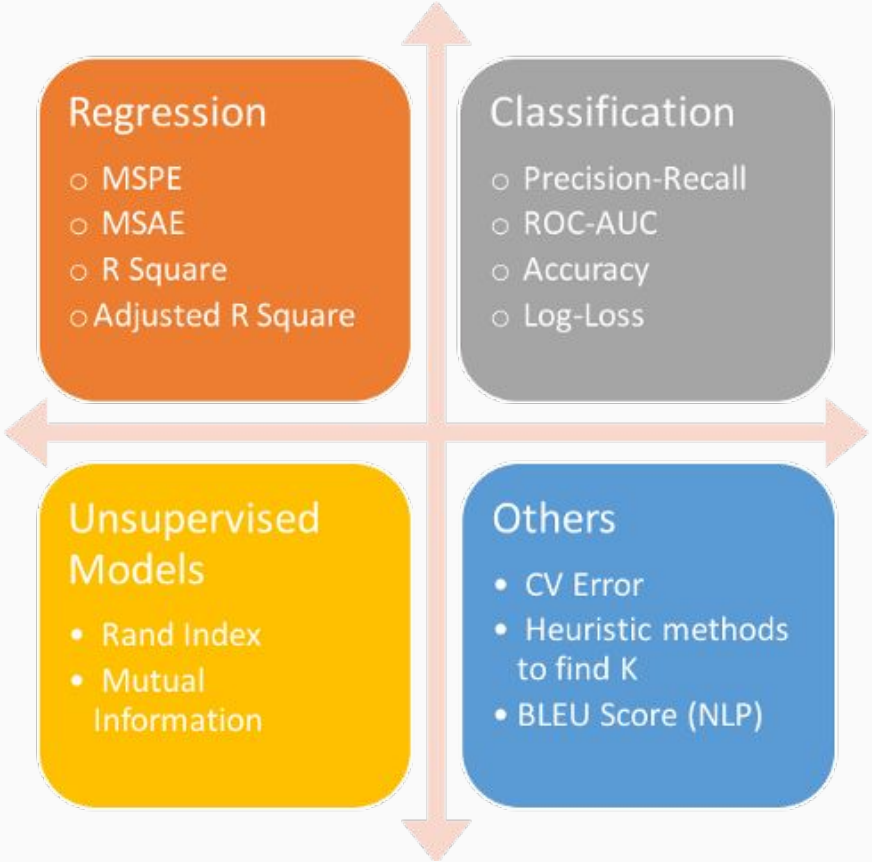
```
my_callbacks = [
    tf.keras.callbacks.EarlyStopping(patience=2),
    tf.keras.callbacks.ModelCheckpoint(filepath='model.{epoch:02d}-{val_loss:.2f}.h5'),
    tf.keras.callbacks.TensorBoard(log_dir='./logs'),
]
model.fit(dataset, epochs=10, callbacks=my_callbacks)
```

Training

```
print("Fit model on training data")
history = model.fit(
    x_train,
    y_train,
    batch_size=64,
    epochs=2,
    # We pass some validation for
    # monitoring validation loss and metrics
    # at the end of each epoch
    validation_data=(x_val, y_val),
)
```

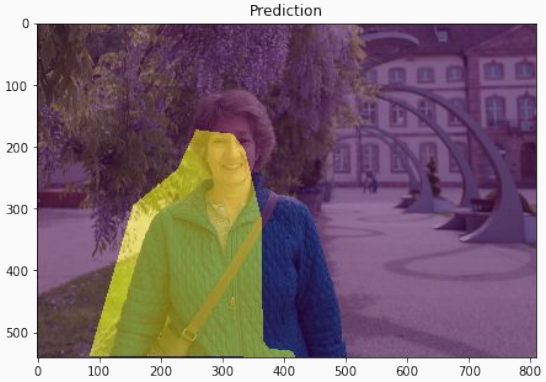
```
Fit model on training data
Epoch 1/2
782/782 [==============================] - 1s 1ms/step - loss: 0.5879 - sparse_categorical_acc
Epoch 2/2
782/782 [==============================] - 1s 824us/step - loss: 0.1732 - sparse_categorical_a
```

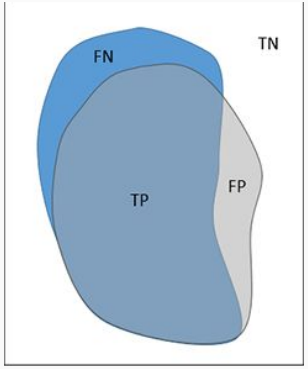
Metrics



Source: kdnuggets.com

Metrics

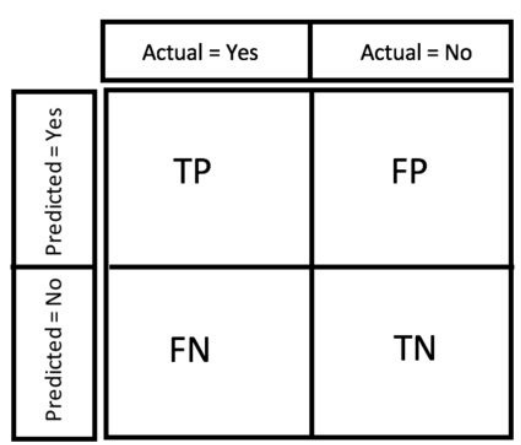




	Actual = Yes	Actual = No
Predicted = Yes	TP	FP
Predicted = No	FN	TN

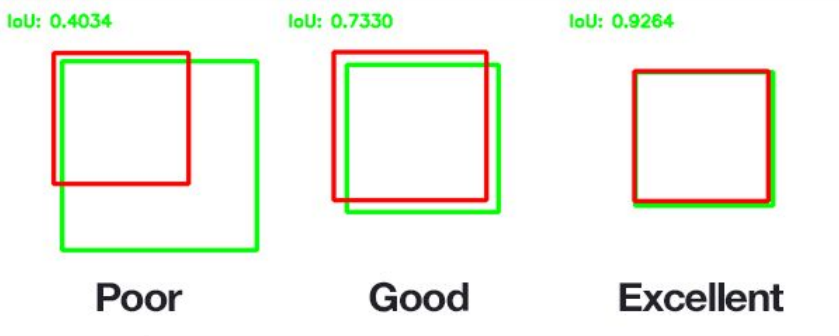
Metrics

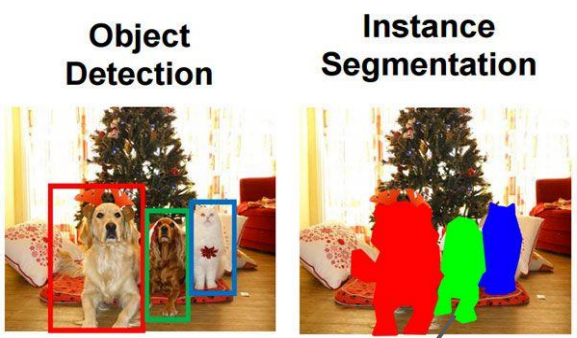
Metric	Formula
True positive rate, recall	$\frac{TP}{TP+FN}$
False positive rate	$\frac{FP}{FP+TN}$
Precision	$\frac{TP}{TP+FP}$
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
F-measure	$\frac{2 \cdot precision \cdot recall}{precision + recall}$



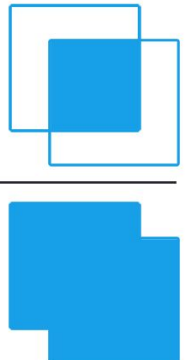
Source 1: deepai.org
Source 2: kdnuggets.com

Metrics





$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

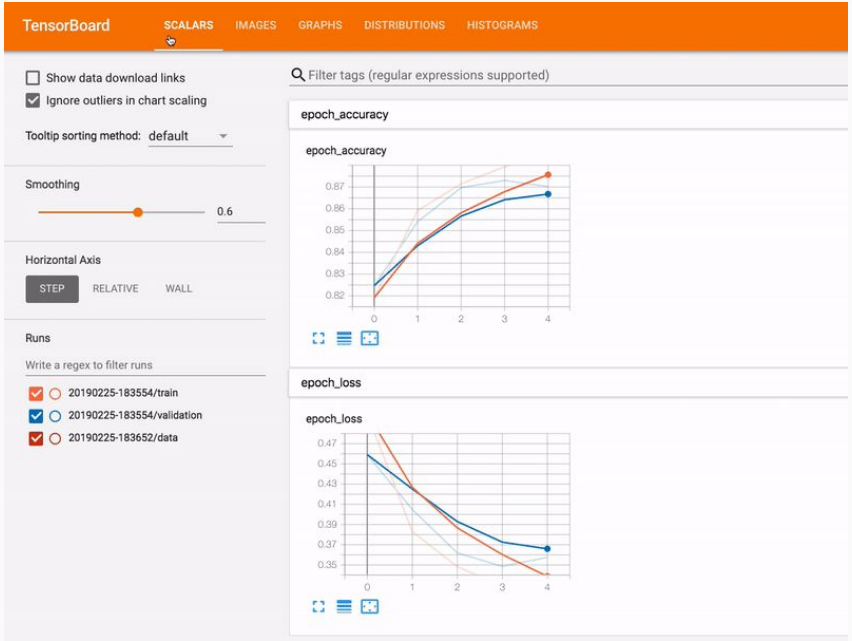


TensorBoard

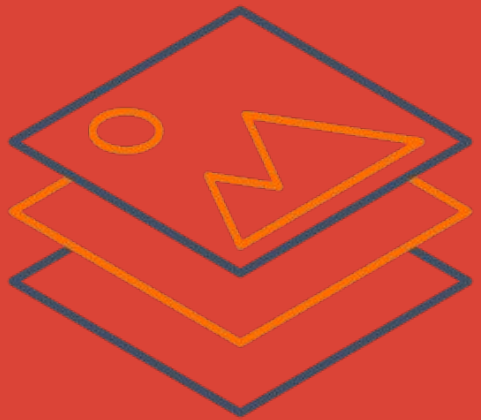
for visual debugging

```
model = create_model()
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

log_dir = "logs/fit/" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir, histogram_freq=1)

model.fit(x=x_train,
          y=y_train,
          epochs=5,
          validation_data=(x_test, y_test),
          callbacks=[tensorboard_callback])
```

```
%tensorboard --logdir logs/fit
```



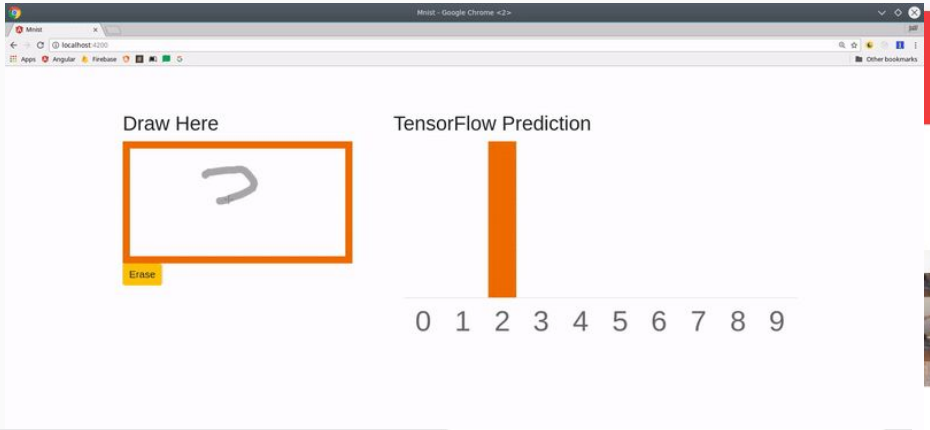
TensorFlow.js for machine learning on the web



TensorFlow.js

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

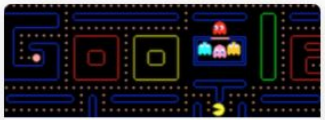
iremkomurcu.com



Emoji Scavenger Hunt

Use your phone's camera to identify emojis in the real world. Can you find all the emojis before time expires?

Explore demo ↗ View code



Webcam Controller

Play Pac-Man using images trained in your browser.

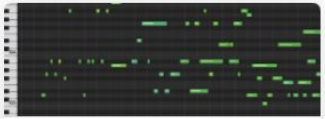
Explore demo ↗ View code

Teachable Machine

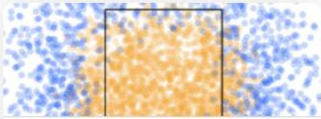
No coding required! Teach a machine to recognize images and play sounds.

Explore demo ↗ View code

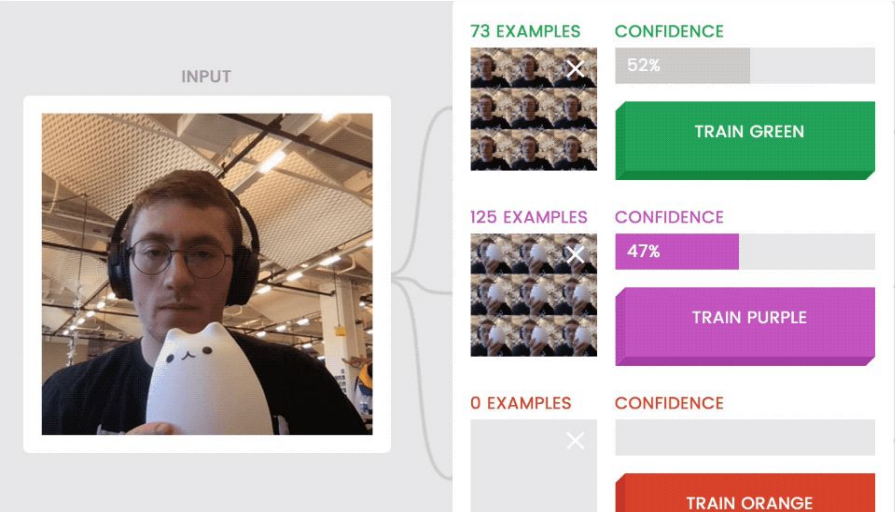
Move Mirror



Performance RNN



Node.js Pitch Prediction



TensorFlow.js

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

- Using Script Tags
- Installation from NPM and using a build tool like Parcel, WebPack or Rollup

TensorFlow Lite
for mobile and embedded ML

TensorFlow Lite

https://www.tensorflow.org/lite/examples/

TensorFlow Lite example apps

A collection of TensorFlow Lite apps.

Image classification

Test an image classification solution with a pre-trained model that can recognize 1000 different types of items from input frames on a mobile camera.

Try it on Android
Try it on iOS
Try it on Raspberry Pi

Object detection

Explore an app using a pre-trained model that draws and labels bounding boxes around 1000 different recognizable objects from input frames on a mobile camera.

Try it on Android
Try it on iOS
Try it on Raspberry Pi

Pose estimation

Explore an app that estimates poses of people in an image.

Try it on Android
Try it on iOS

Speech recognition

Explore an app that uses a microphone to spot keywords

Gesture recognition

Train a neural network to recognize gestures caught on

Smart reply

Generate reply suggestions to input conversational chat

Practice Session

Use of GitHub
Github Code Review
Segmentation Example

Please visit on YouTube video to talk about this presentation and practice session.
You can find the video link in the my GitHub repo.

THANKS

Does anyone have any questions?

iremkomurcubm@gmail.com
iremkomurcu.com