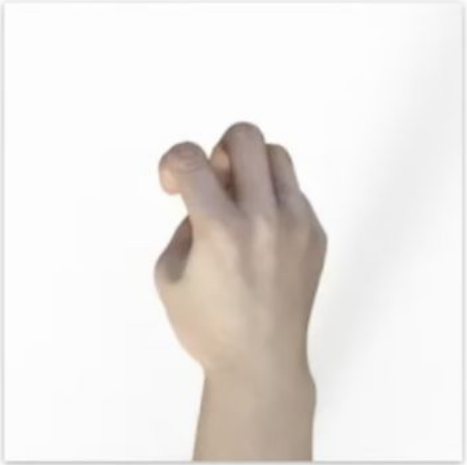
HOCHSCHULE DER MEDIEN

Deep Learning

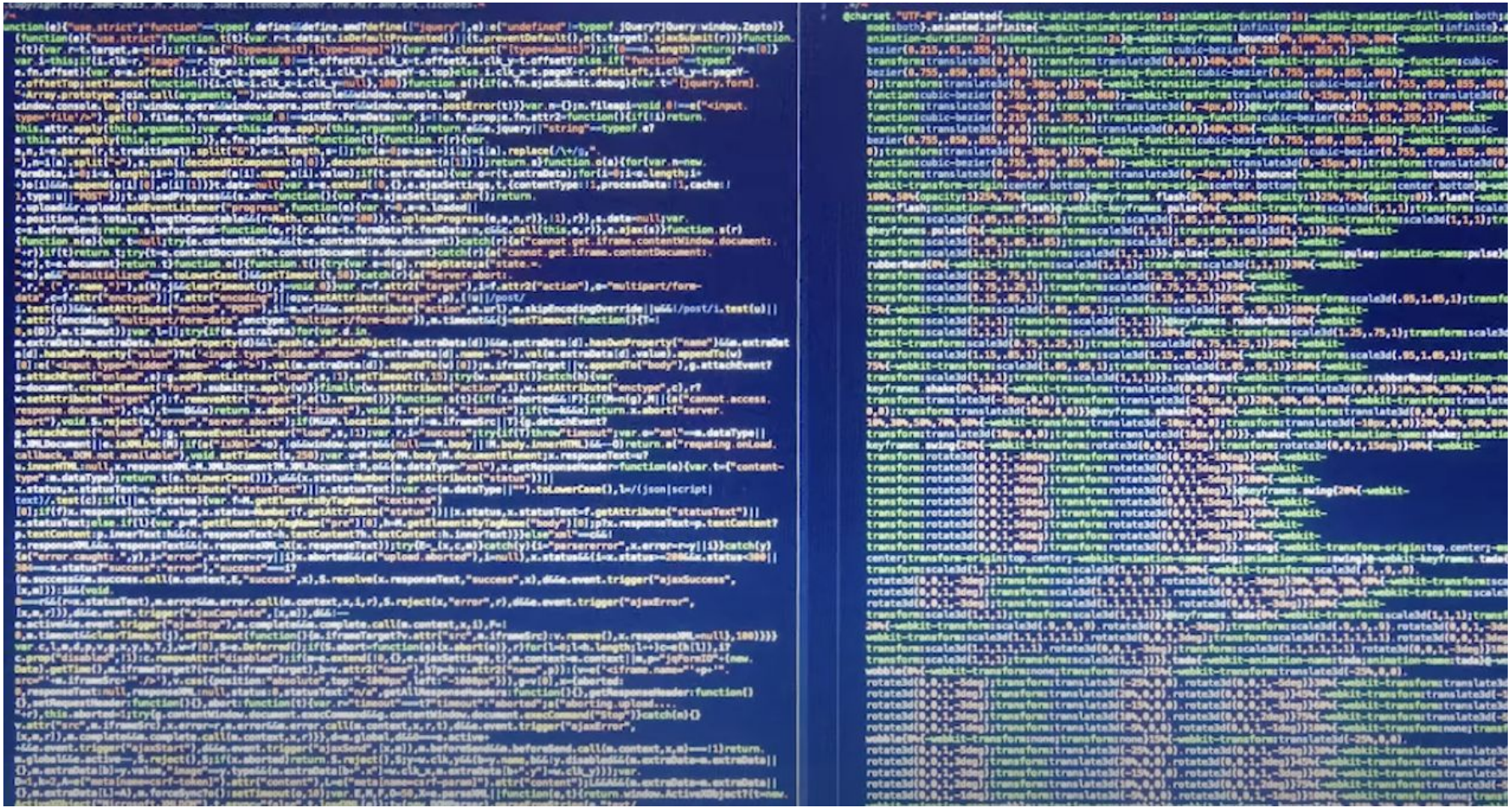
Computer vision

Prof. Dr. Jan Kirenz
HdM Stuttgart

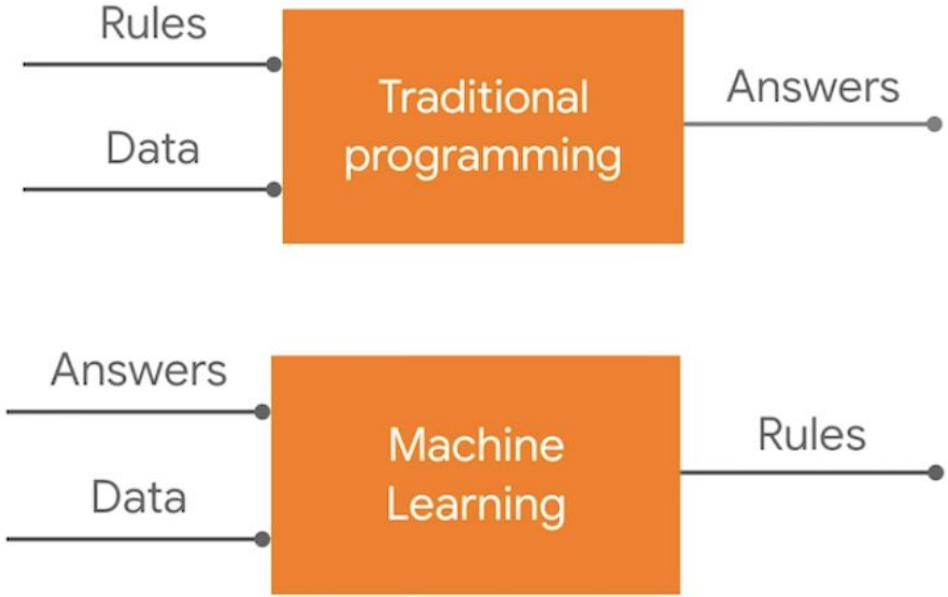


Source: Moroney (2019)

Source: Moroney (2019)



Source: Moroney (2019)



Source: Moroney (2019)

Data and Labels

```
01010010101001010101
00101010100101110101
00101010010101001010
1001010100101010
```

Label = Rock

```
10010100111110101011
10101011101010111010
10101111010101011111
1110001111010101
```

Label = Scissors

```
10101001010010101010
10101001001001000100
10011111010101111101
0100100111101011
```

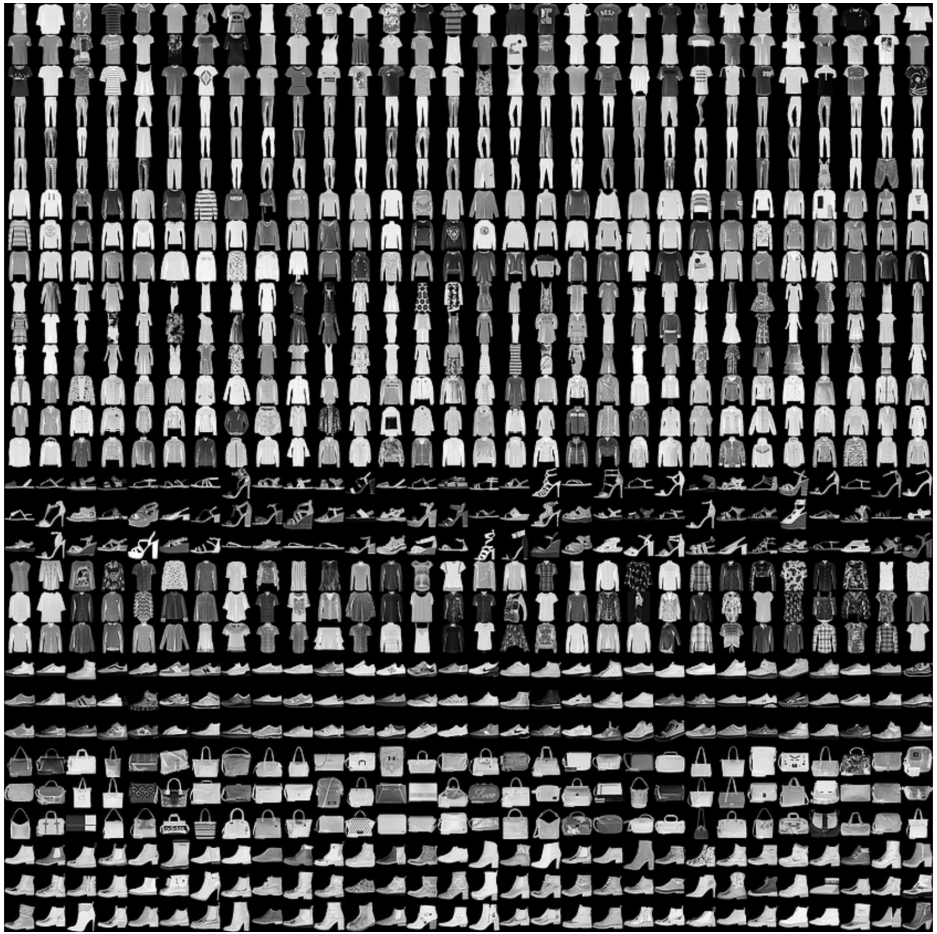
Label = Paper

Source: Moroney (2019)

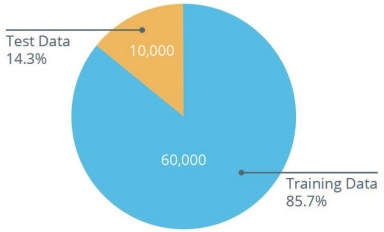
Example with fashion data

Source: Moroney (2019)

Source: Moroney (2019)

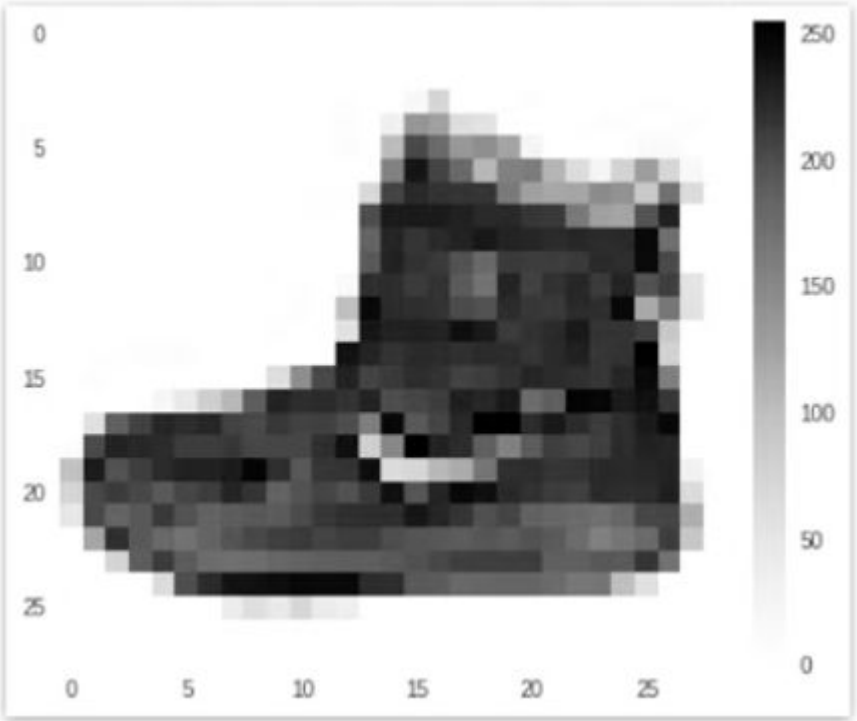


Label	Class
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot



Fashion MNIST

- 70k greyscale Images
- 10 Categories
- Images are 28x28
- Can train a neural net



Source: Moroney (2019)

Import libraries

```
import tensorflow as tf

from tensorflow import keras
```

```
fashion_mnist = keras.datasets.fashion_mnist

(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

Import data

```
import tensorflow as tf

from tensorflow import keras


fashion_mnist = keras.datasets.fashion_mnist

(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

Create training and test images and labels

```
import tensorflow as tf

from tensorflow import keras


fashion_mnist = keras.datasets.fashion_mnist

(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

9

Class of type of clothing (ankle boot)

Neural network design

```
model = keras.Sequential)[
        keras.layers.Flatten(input_shape=(28, 28)),
        keras.layers.Dense(128, activation=tf.nn.relu),
        keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

Input shape equals size of our images (x)

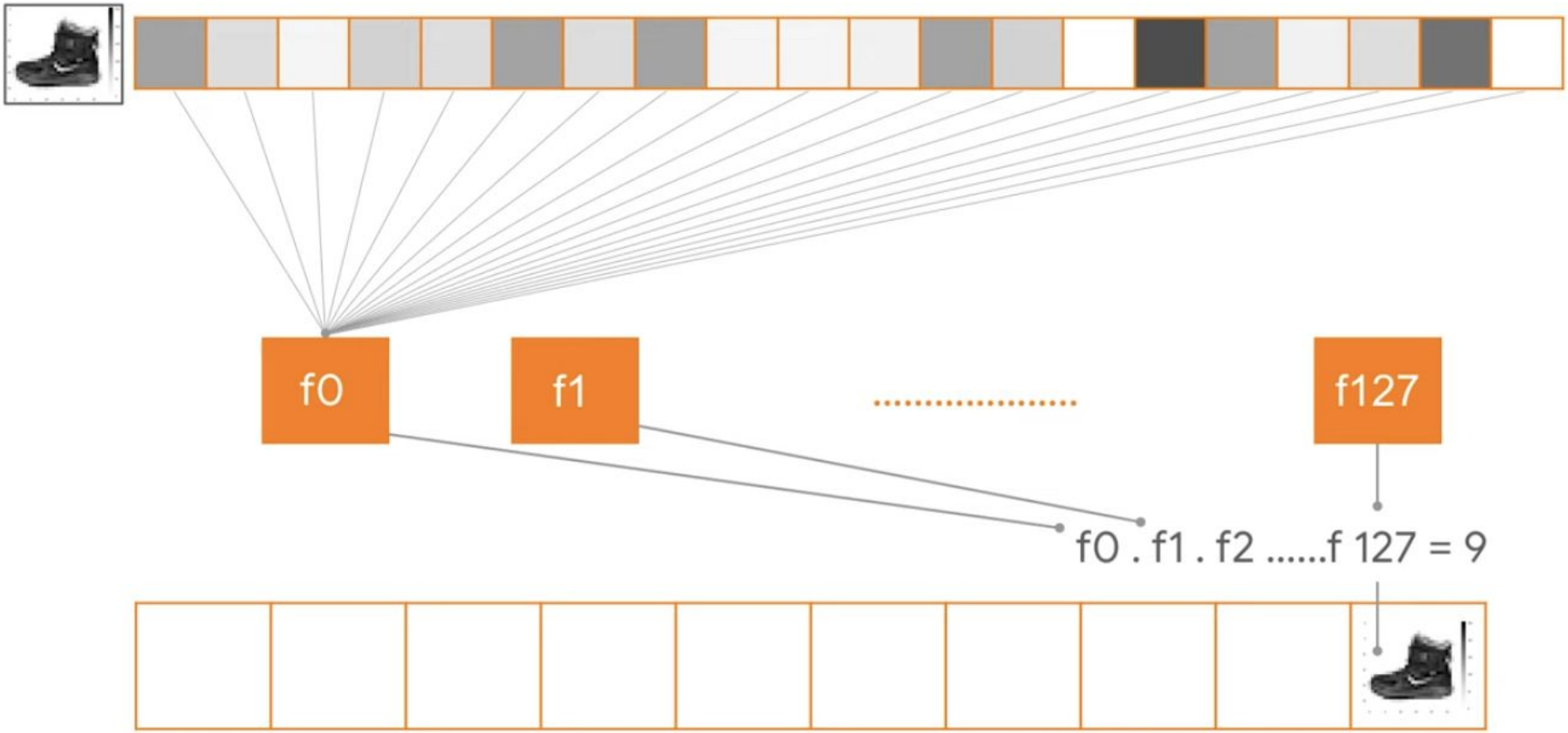
```
model = keras.Sequential)[
        keras.layers.Flatten(input_shape=(28, 28)),
        keras.layers.Dense(128, activation=tf.nn.relu),
        keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

Number of different items in our dataset (labels)

```
model = keras.Sequential)[
        keras.layers.Flatten(input_shape=(28, 28)),
        keras.layers.Dense(128, activation=tf.nn.relu),
        keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

Number of functions

```
model = keras.Sequential)[
        keras.layers.Flatten(input_shape=(28, 28)),
        keras.layers.Dense(128, activation=tf.nn.relu),
        keras.layers.Dense(10, activation=tf.nn.softmax)
])
```



Source: Moroney (2019)

Compile model with optimizer and loss

```
model = keras.Sequential)[
        keras.layers.Flatten(input_shape=(28, 28)),
        keras.layers.Dense(128, activation=tf.nn.rel),
        keras.layers.Dense(10, activation=tf.nn.softmax)
])
model.compile(optimizer=tf.train.AdamOptimizer(),
              loss='sparse_categorical_crossentropy')
```

Activation functions

```
model = keras.Sequential)[
        keras.layers.Flatten(input_shape=(28, 28)),
        keras.layers.Dense(128, activation=tf.nn.relu),
        keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

Rectified Linear Unit (ReLU) returns a value if it's greater than zero

```
model = keras.Sequential)[
        keras.layers.Flatten(input_shape=(28, 28)),
        keras.layers.Dense(128, activation=tf.nn.relu),
        keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

```
if (x>0) {
      return x;
}
else{
      return 0;
}
```

Softmax is picking the biggest number in a set and sets it to 1. All other values become 0.

```
model = keras.Sequential)[
        keras.layers.Flatten(input_shape=(28, 28)),
        keras.layers.Dense(128, activation=tf.nn.relu),
        keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

0	1	2	3	4	5	6	7	8	9
0.02	0.01	0.02	0.01	0.05	0.01	0.08	0.02	0.01	**9.78**
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**1.00**

Model training with 5 repetitions

```
model.fit(train_images, train_labels, epochs=5)
```

Test model performance with test data

```
test_loss, test_acc = model.evaluate(test_images, test_labels)
```

Model predictions with new images

```
predictions = model.predict(my_images)
```

Resources

The slides are based on the excellent video tutorial “Basic Computer Vision with ML (ML Zero to Hero - Part 2)” by Lawrence Moroney.