USC University of Southern California

ECE 105: Introduction to Electrical Engineering

Lecture 17

Intro to hardware ML

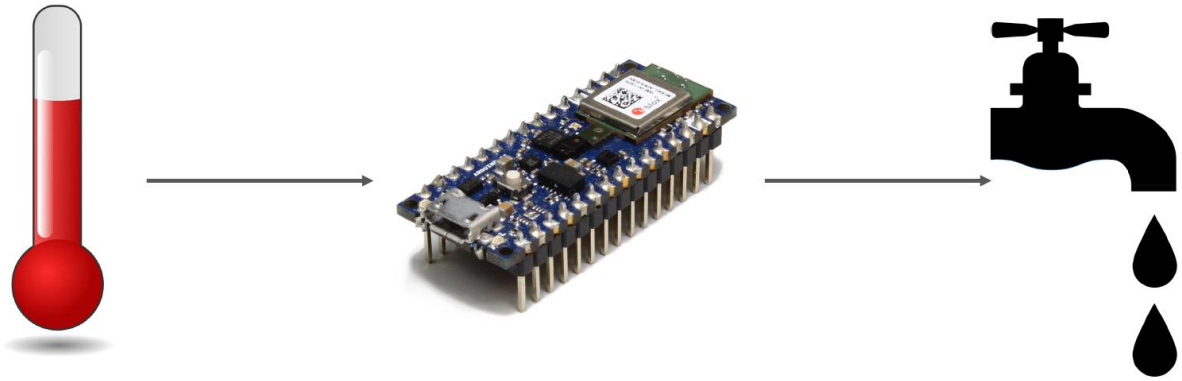
Yasser Khan

Rehan Kapadia

Why use NN when your can hardcode?

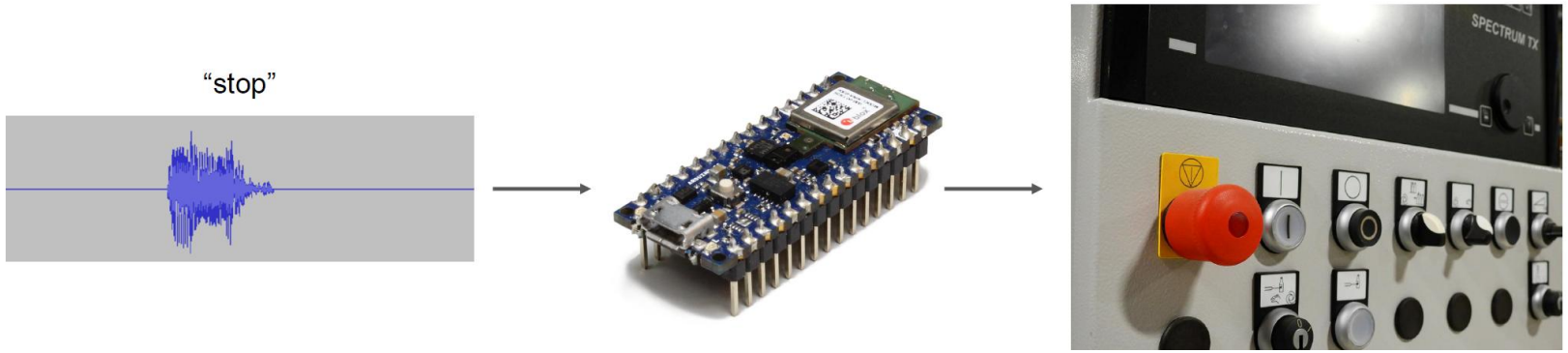
USC University of Southern California

Deterministic



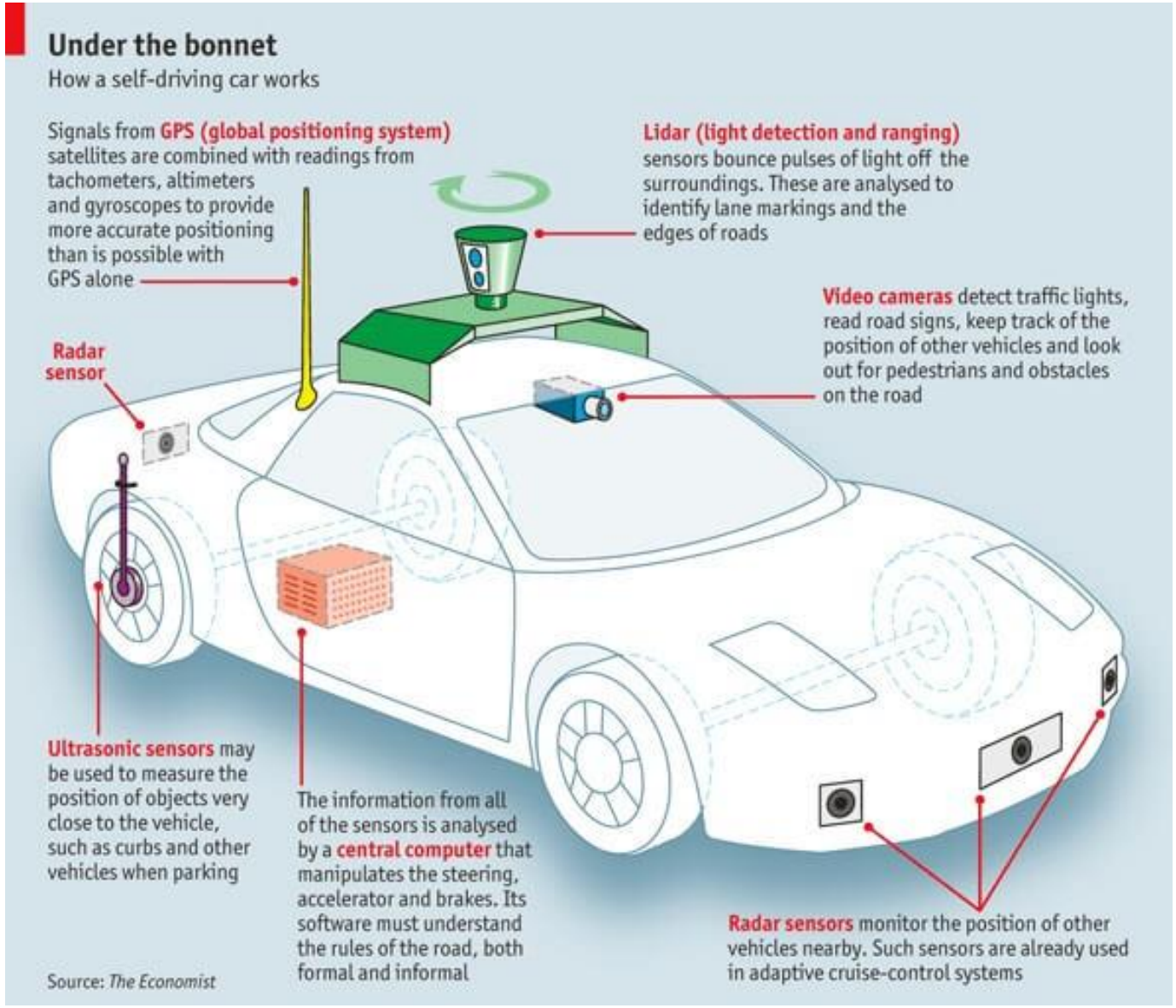
Temperature measurement is absolute

Probabilistic



Voice can vary

Self-driving car



On-device machine learning applications in the single mW and below

USC University of Southern California

Vibration and motion

Any 'signal'

Predictive maintenance, sensor fusion, accelerometer, pressure, lidar/radar, speed, shock, vibration, pollution, density, viscosity, etc.

Voice and sound

Recognition and creation

Keyword spotting, speech recognition, natural language processing, speech synthesis, sound recognition, etc.

Vision

Images and video

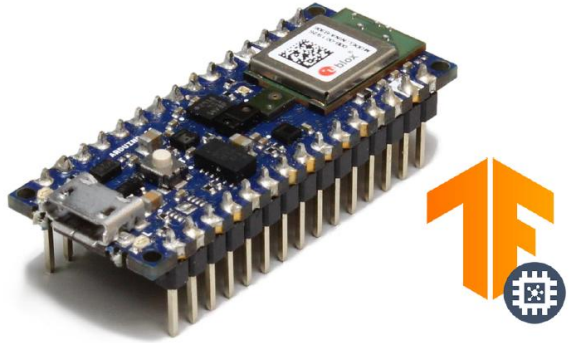
Object detection, face unlock, object classification etc.

Sub-mW computing vs 100s of mW computing

USC
University of
Southern California

Single Board Computer

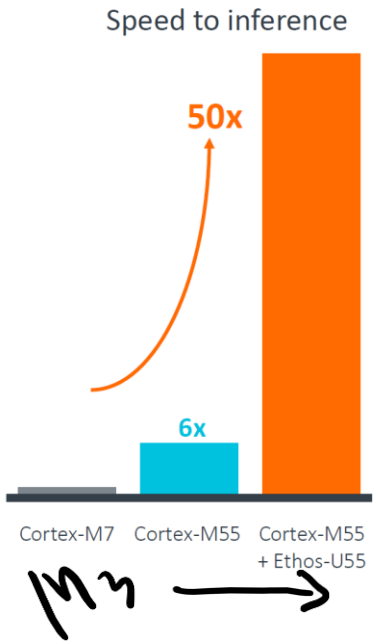
- More powerful (faster processor, more memory)
- Runs full, general purpose operating system (OS)
- Can provide full command line or graphical user interface
- Requires more power

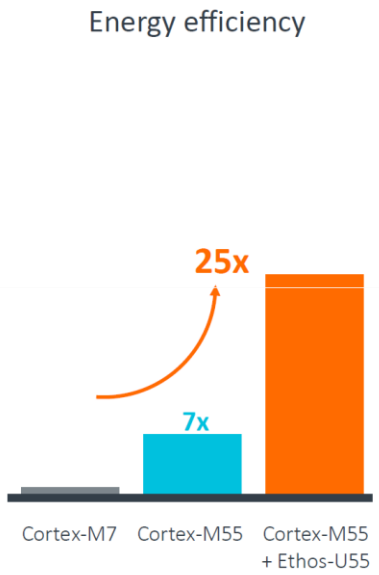


Microcontroller

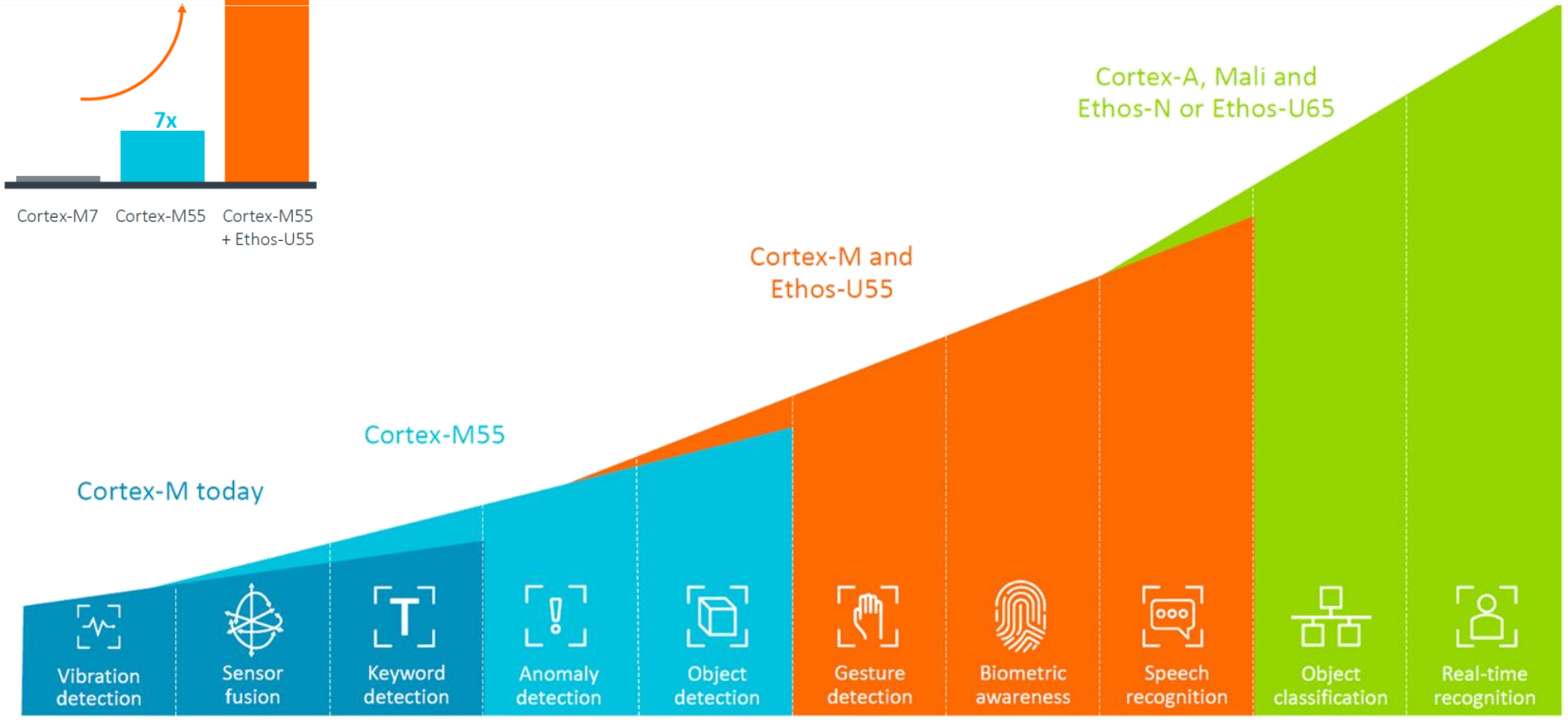
- Less powerful
- Bare-metal (superloop) or real-time operating system (RTOS)
- Limited or no user interface
- Requires less power

Cortex-M series microcontrollers are becoming powerful

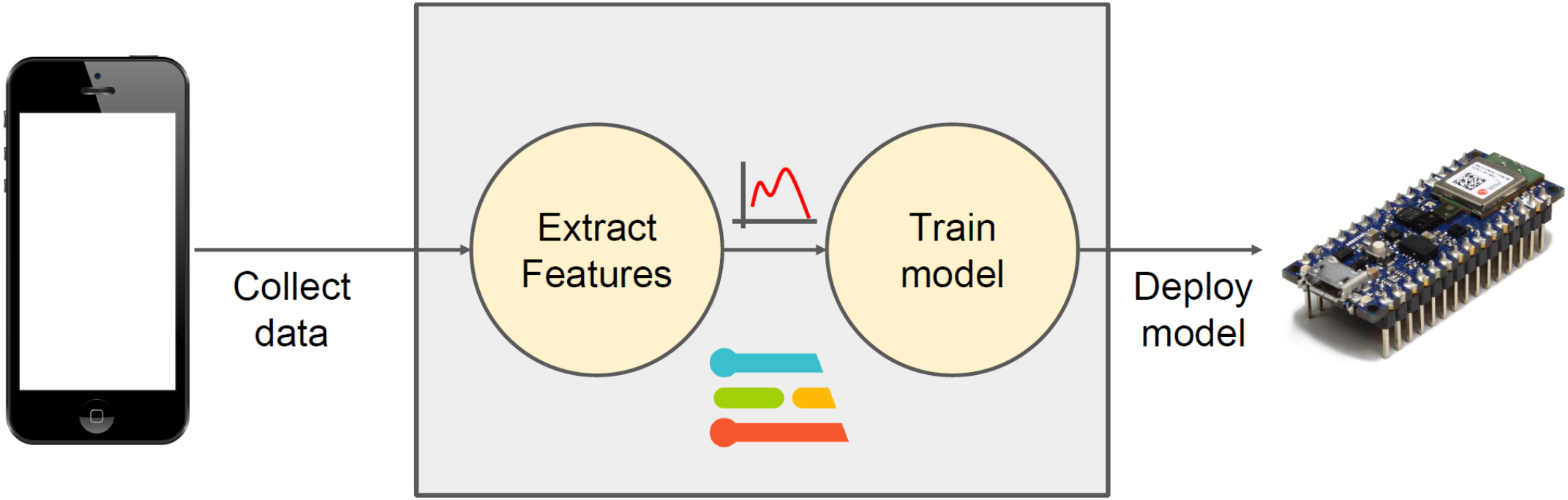




nRF52840 is built around the 32-bit ARM® Cortex™-M4 CPU with floating point unit running at 64 MHz.



ML in microcontrollers TinyML



Motion classification from accelerometer data

Acceleration (m/s^2)

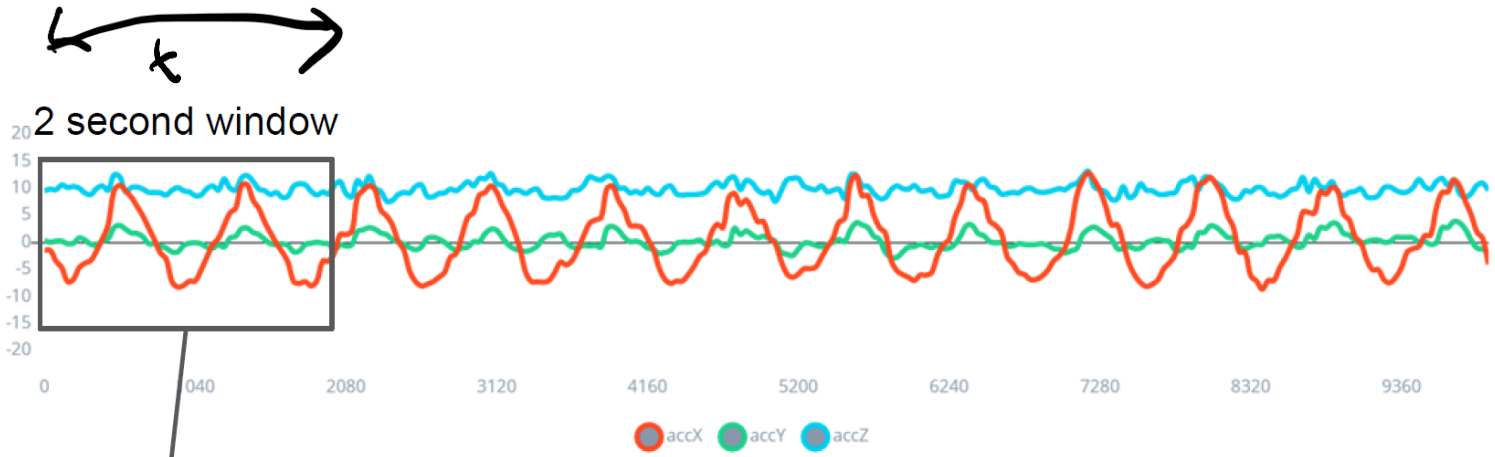
x	-1.4
y	0.4
z	9.6

Model

- Left-right
- Up-down
- Circle
- Idle

class1 → class4

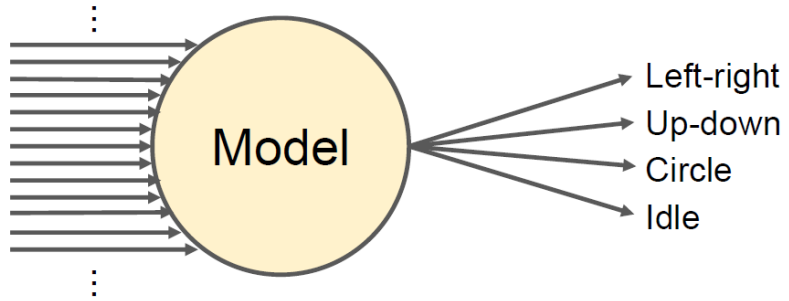
Motion classification from accelerometer data



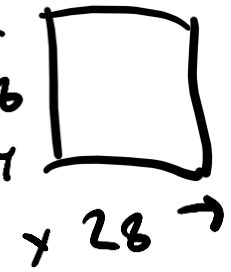
Acceleration (m/s²)

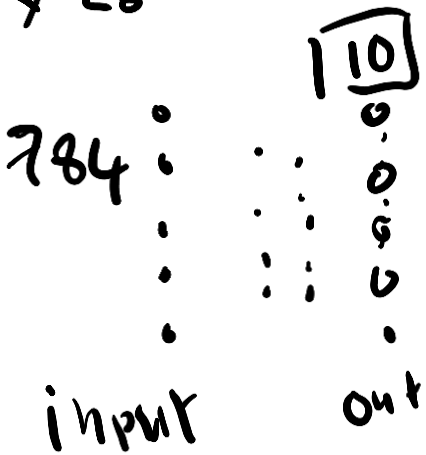
x	-1.4	-1.4	-2.8	-3.4	-4.0	⋯
y	0.4	0.4	0.1	0.2	0.3	⋯
z	9.6	9.6	9.9	9.7	9.8	⋯

125 samples for each axis



375 total inputs to the model!



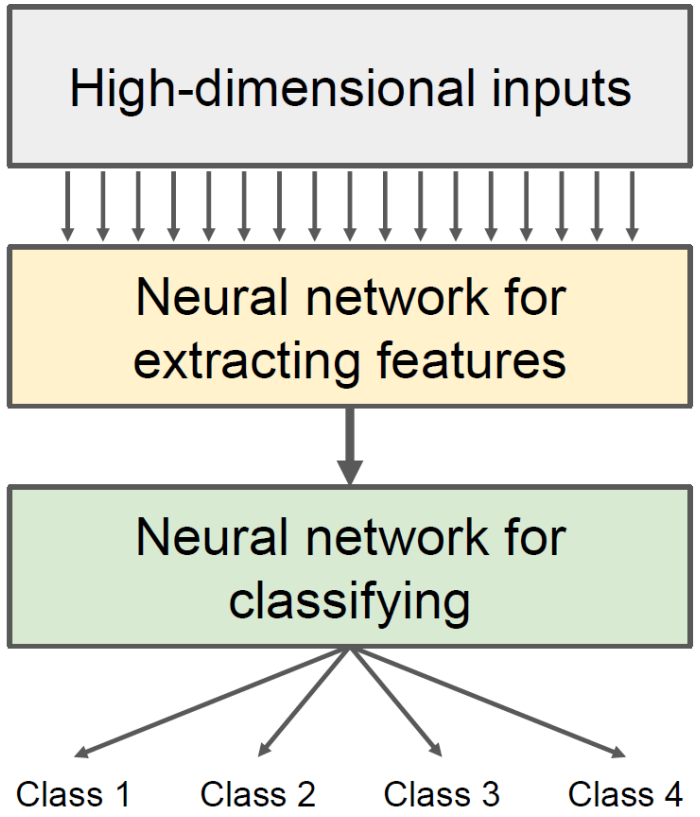


Way too many inputs to the model

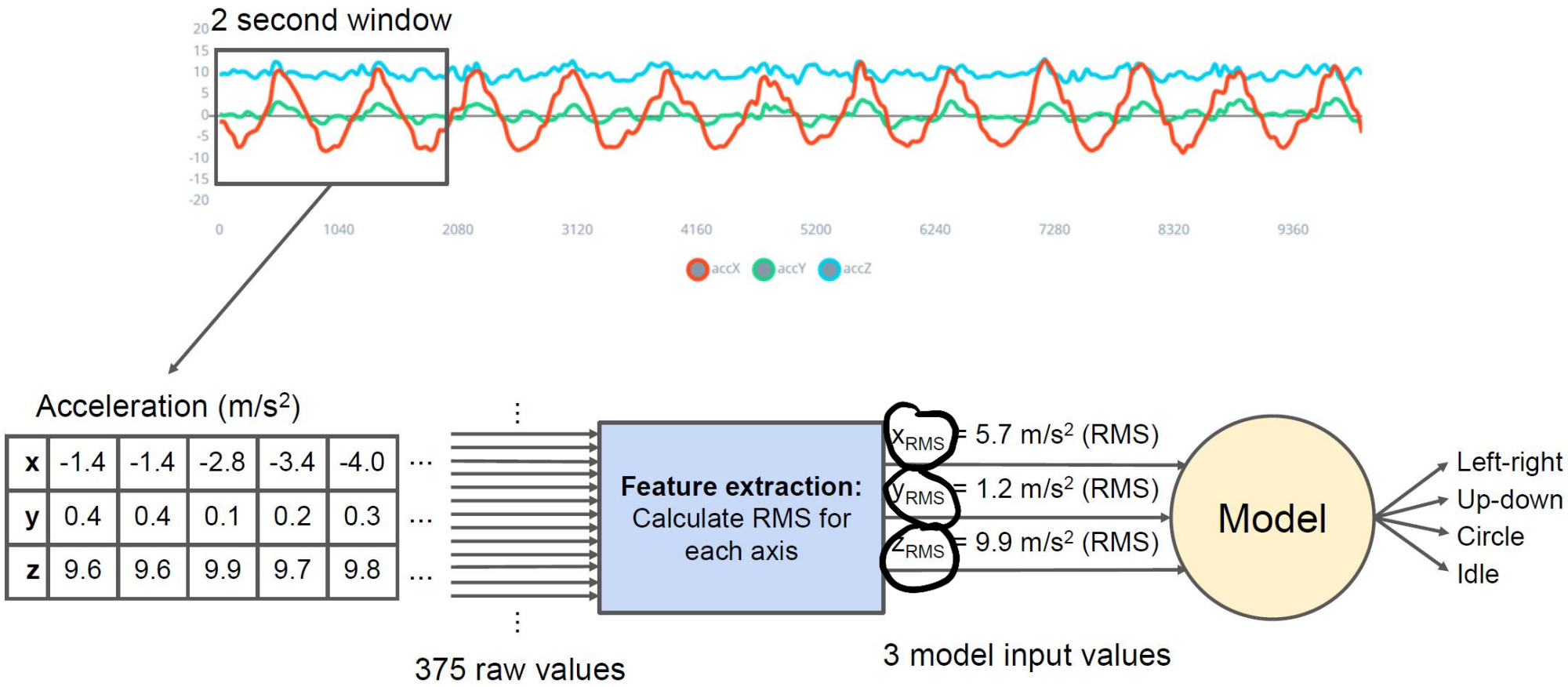
USC University of Southern California

Problems with deep learning

1. Computational complexity
2. Requires lots of training data



Feature extraction

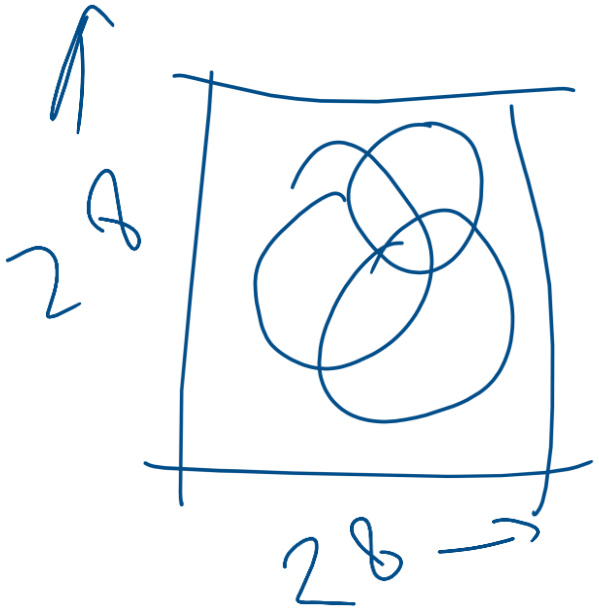


125 x 3

375 → 3 inputs

What is a feature

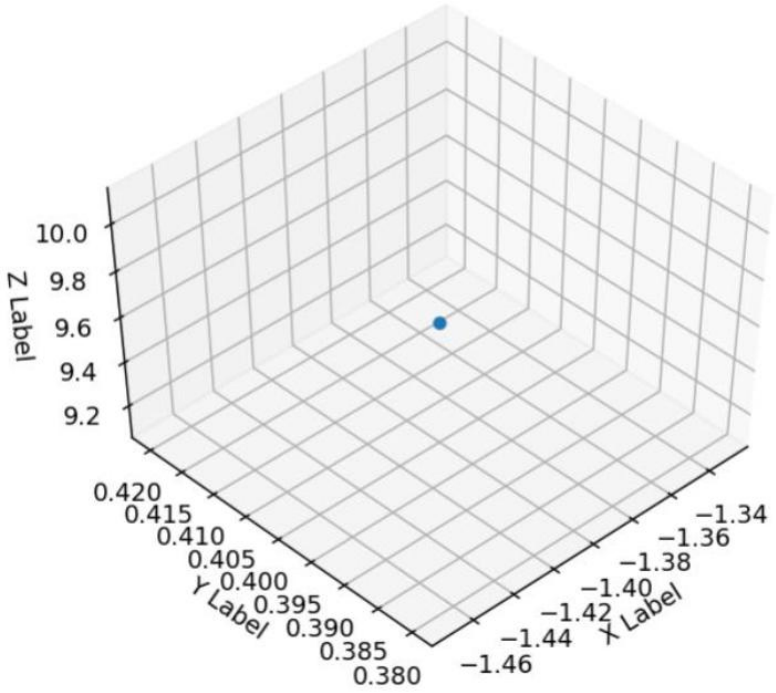
- Individual measurable property or characteristic of a phenomenon being observed



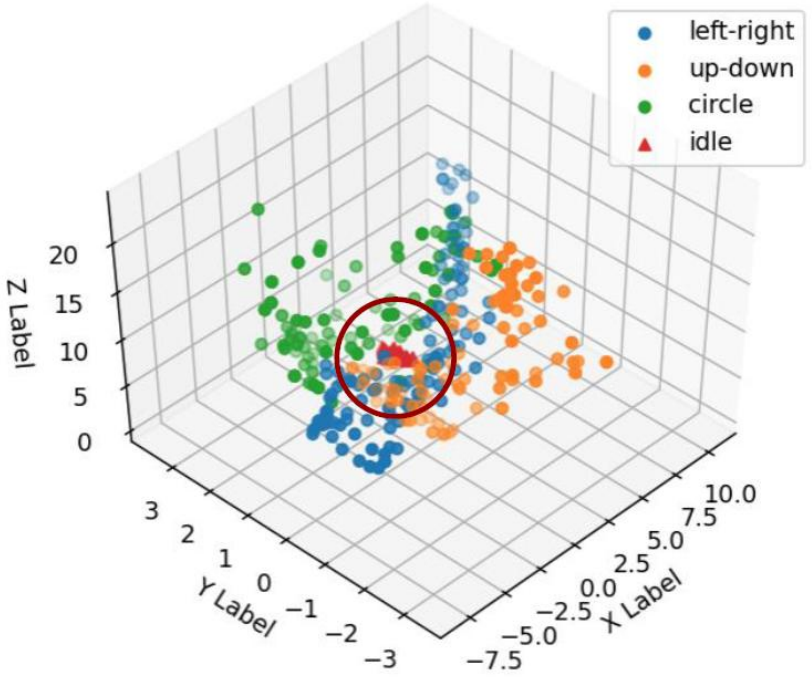
Raw data

USC University of Southern California

1 (x, y, z) accelerometer point from “left-right” sample

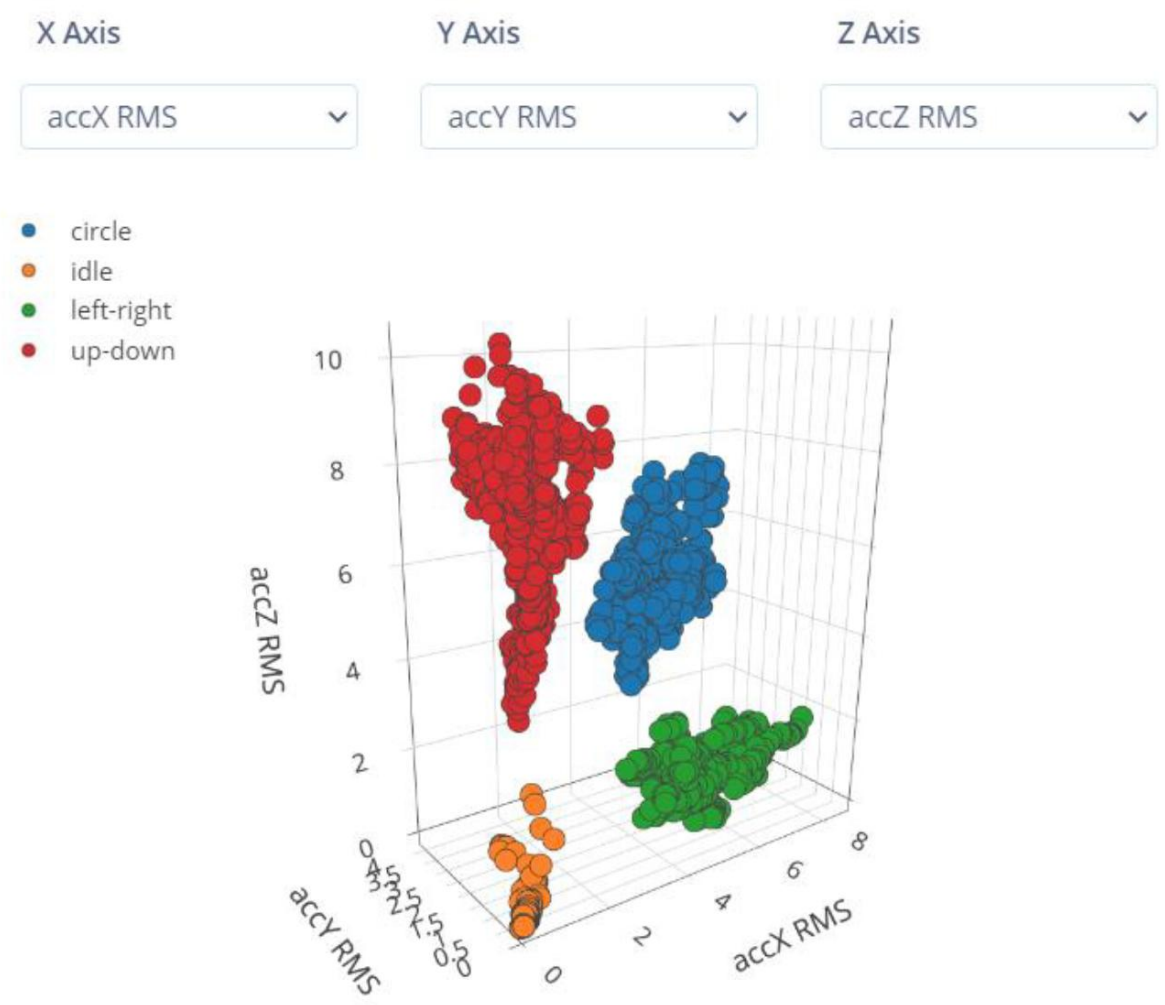


Many (x, y, z) accelerometer points from all classes

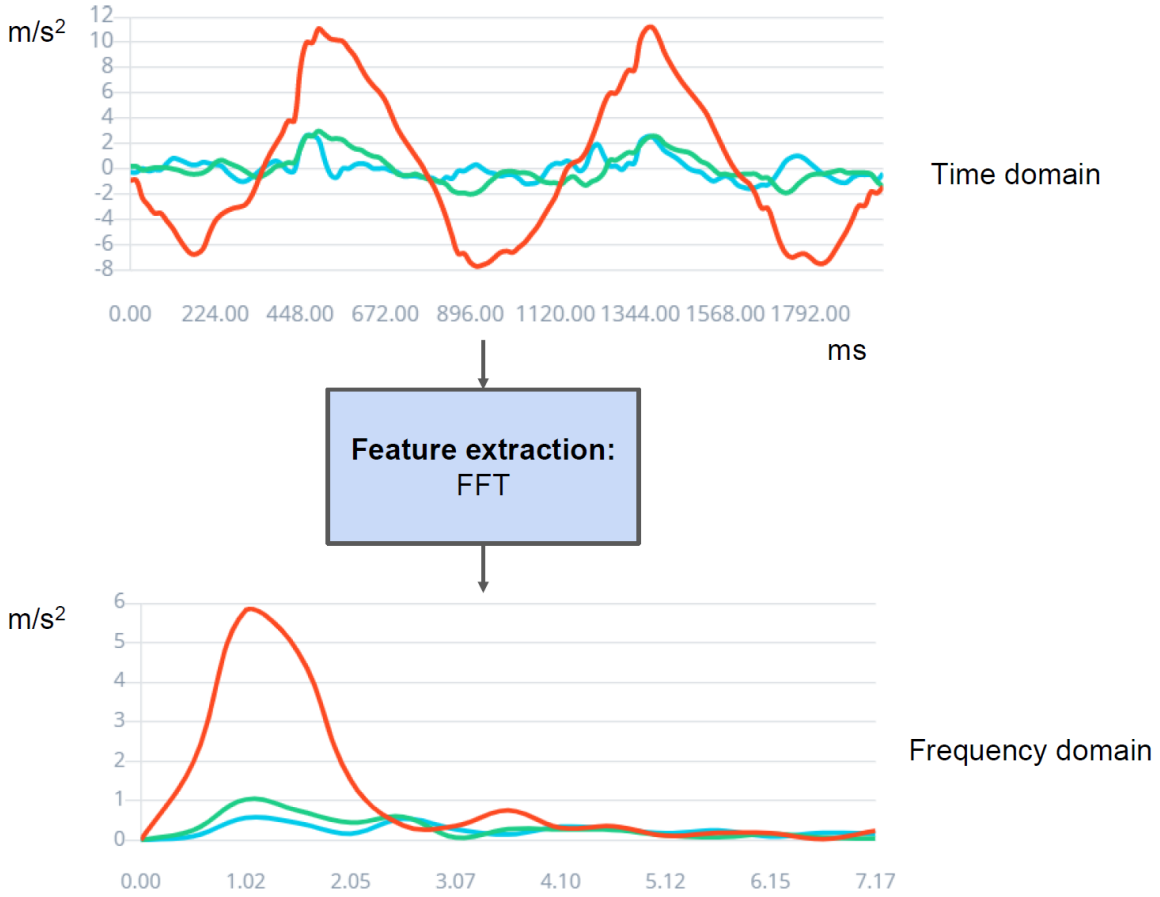


RMS of acceleration in each axis as the feature

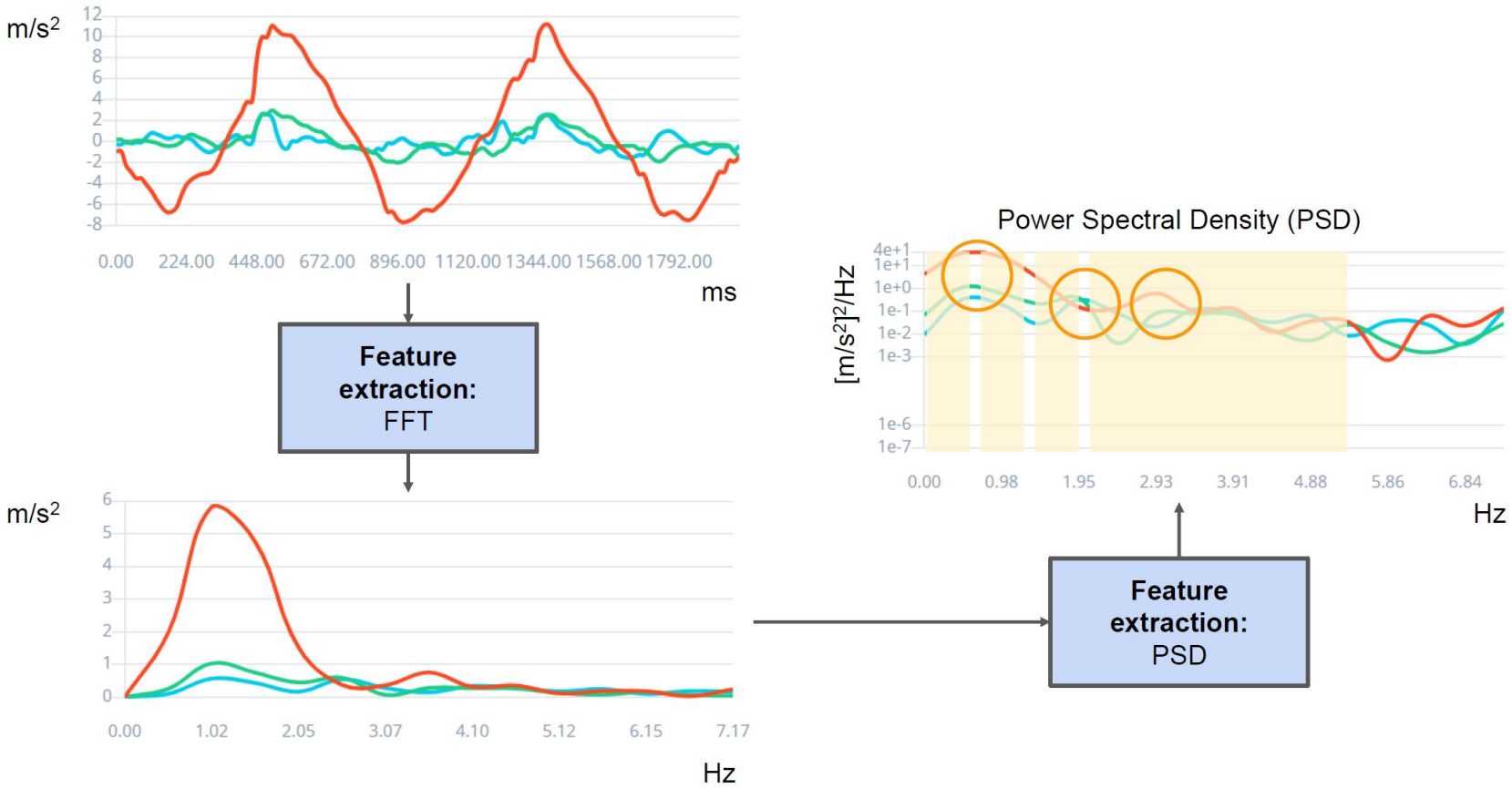
USC University of Southern California



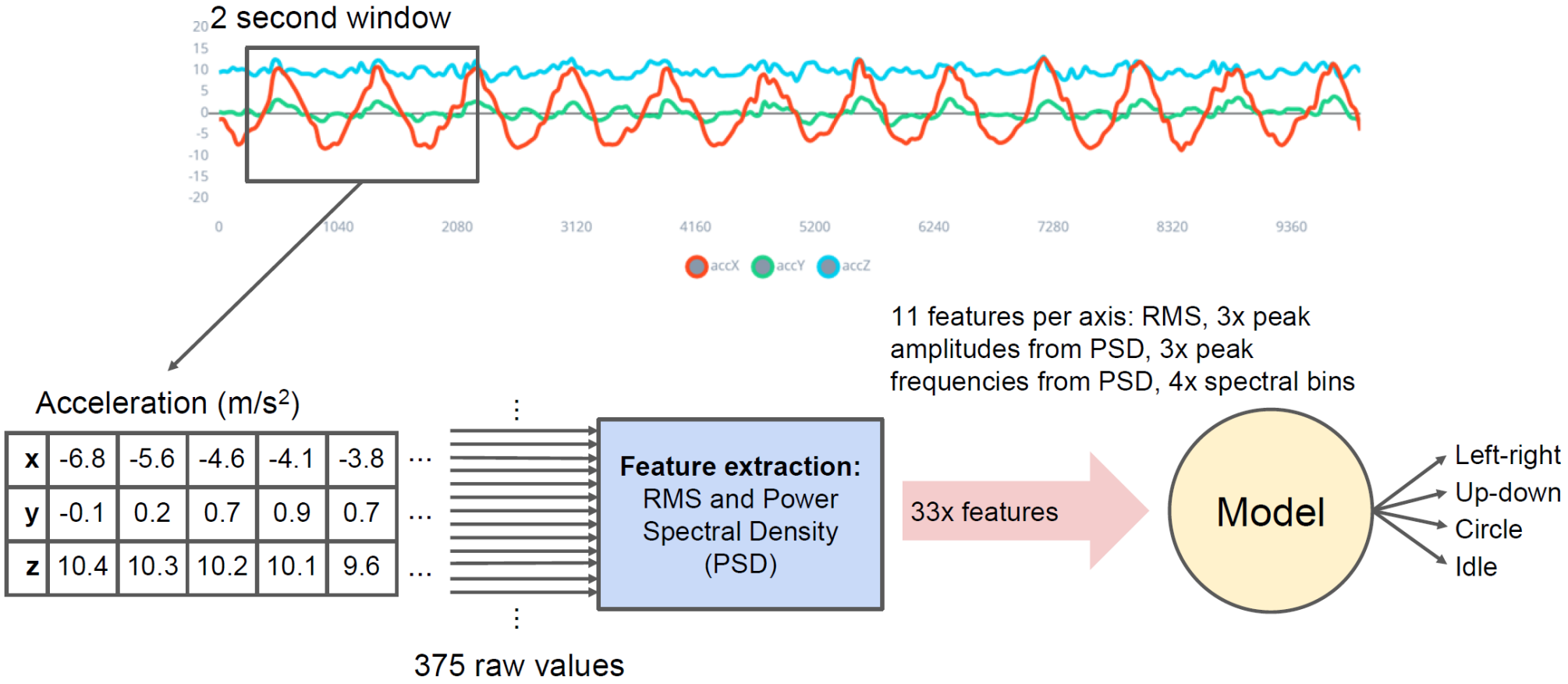
Features don’t need to be only in the time domain



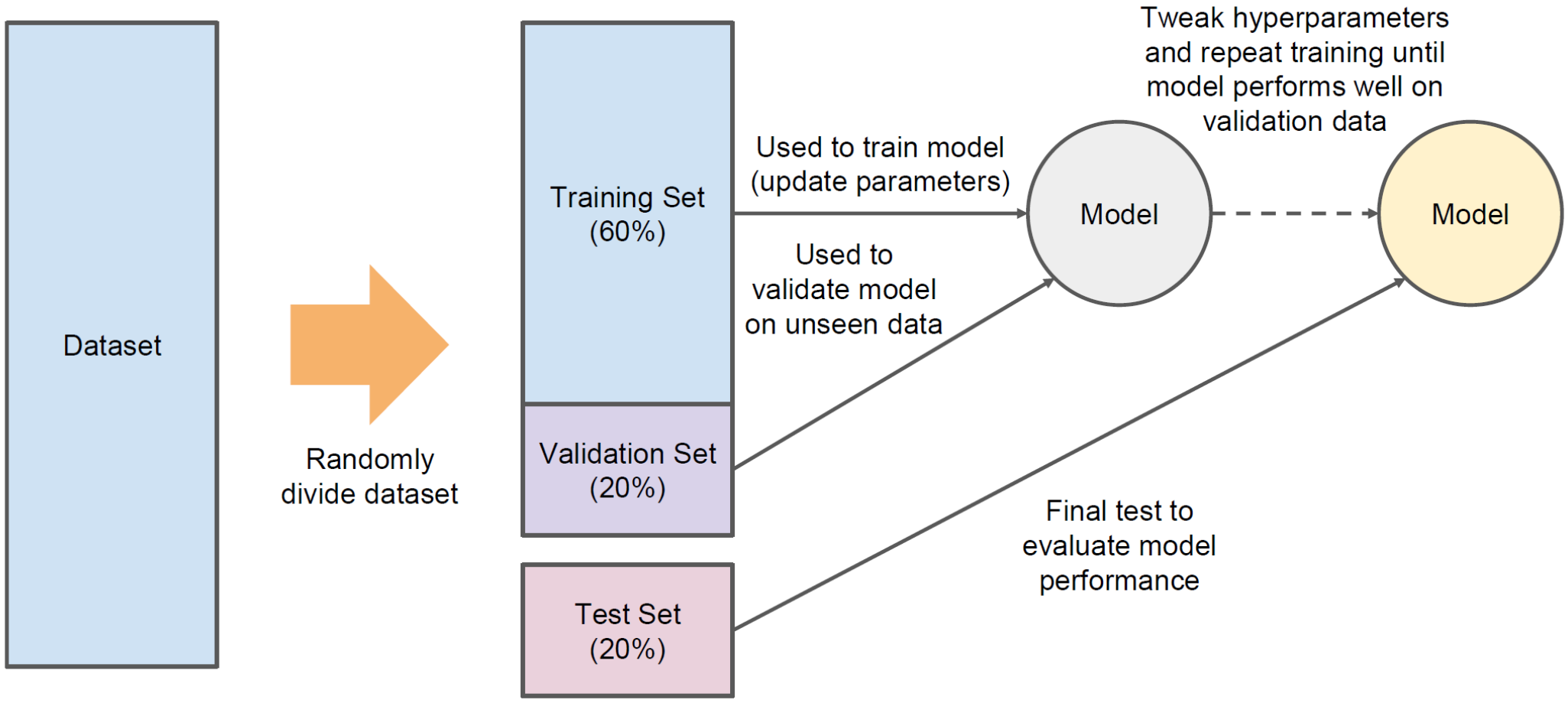
Features in the frequency domain



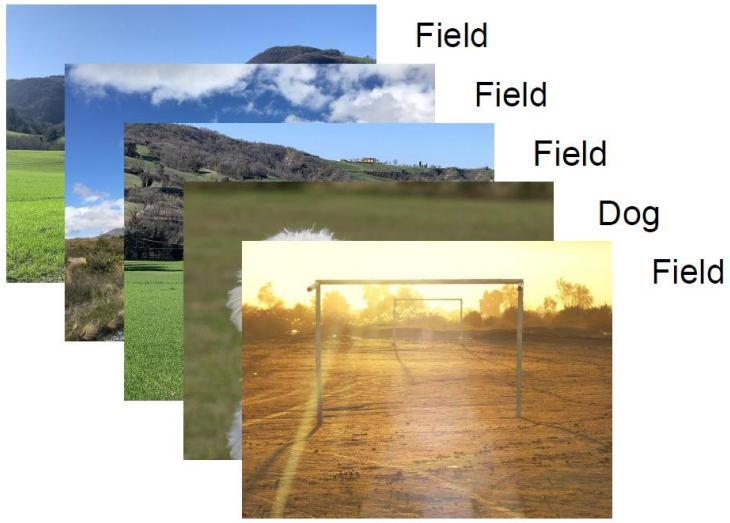
Using both time and frequency domain features

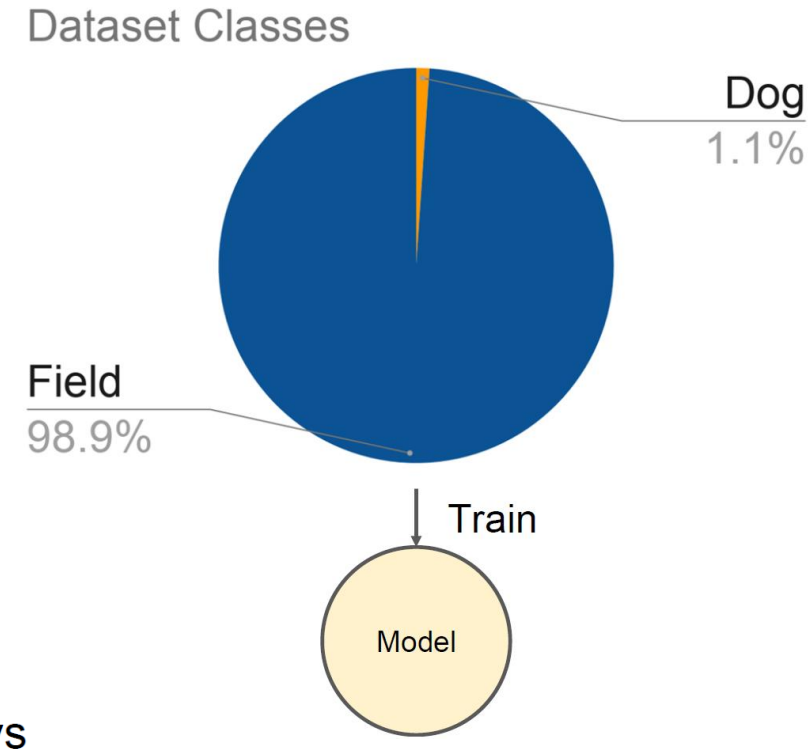


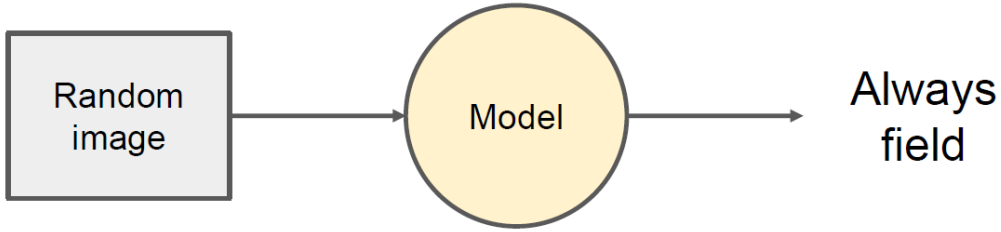
Holdout Method for training, validation, and testing



Dataset balance



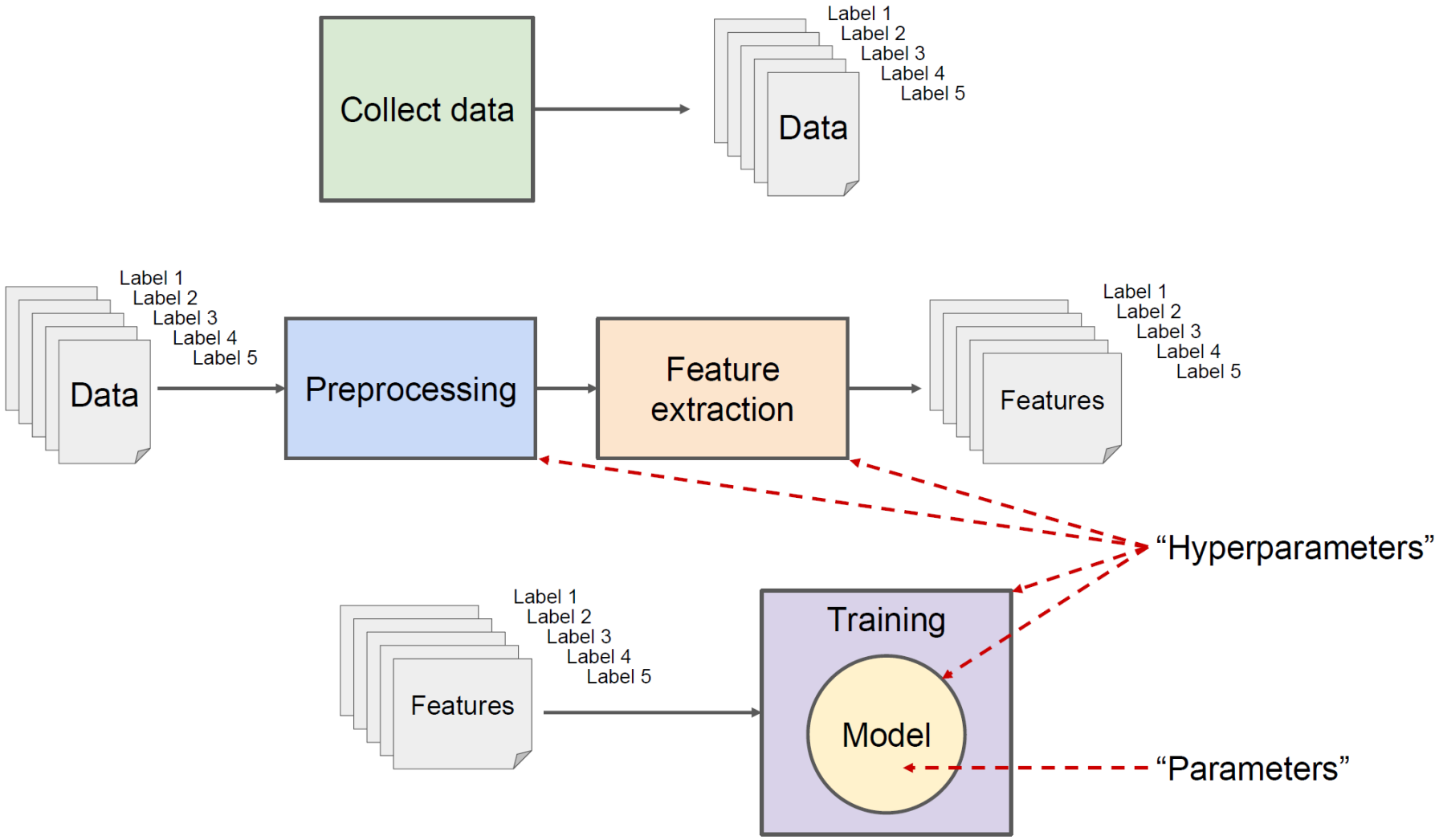




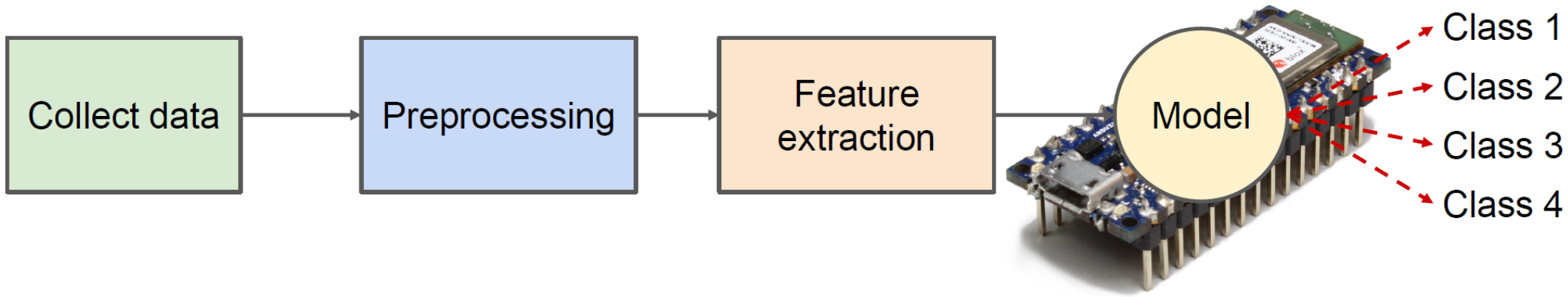
99% accuracy!

"Naive classifier"

Data flow diagram

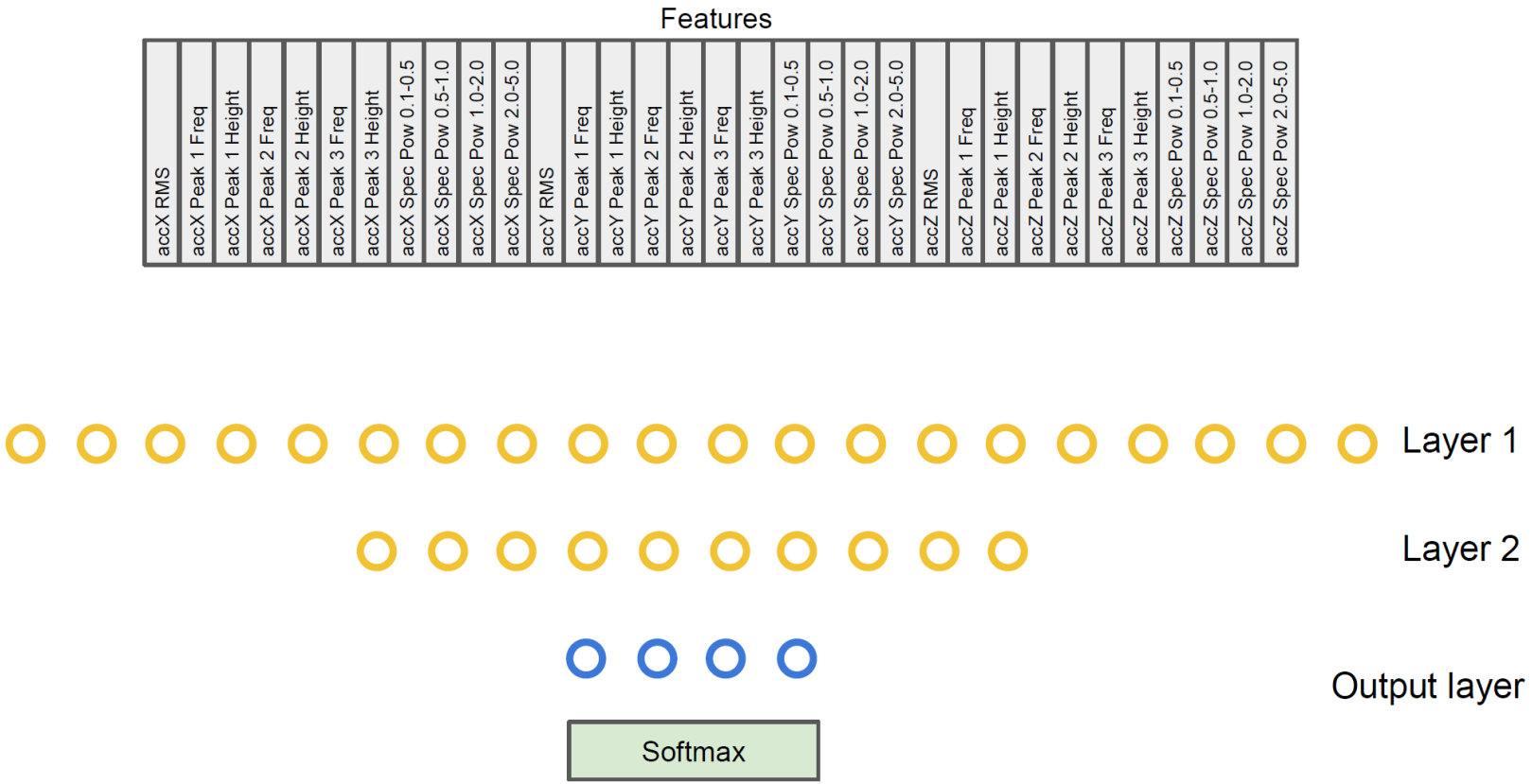


Final implementation

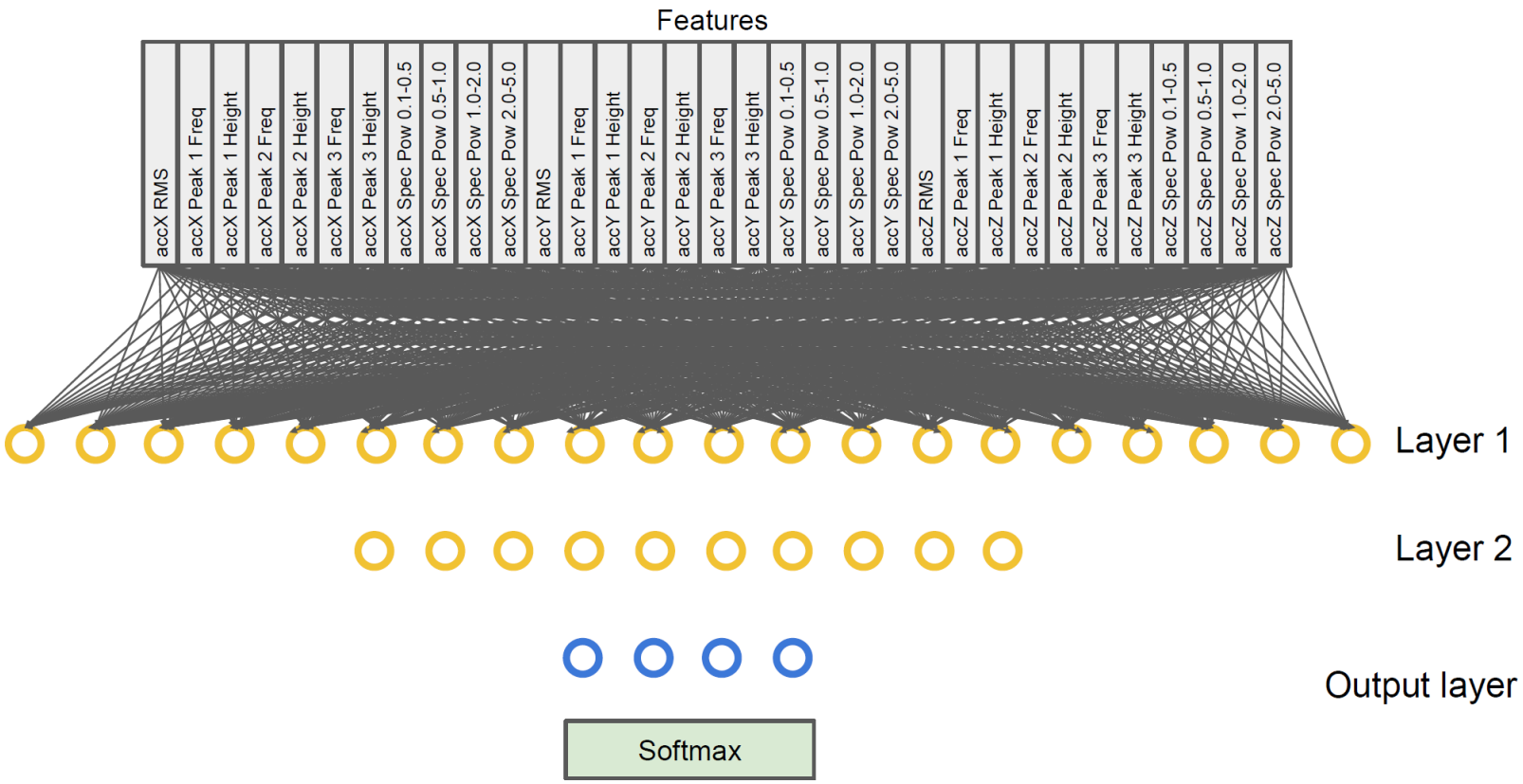


Inference: using the machine learning model to make predictions on unseen data in the wild

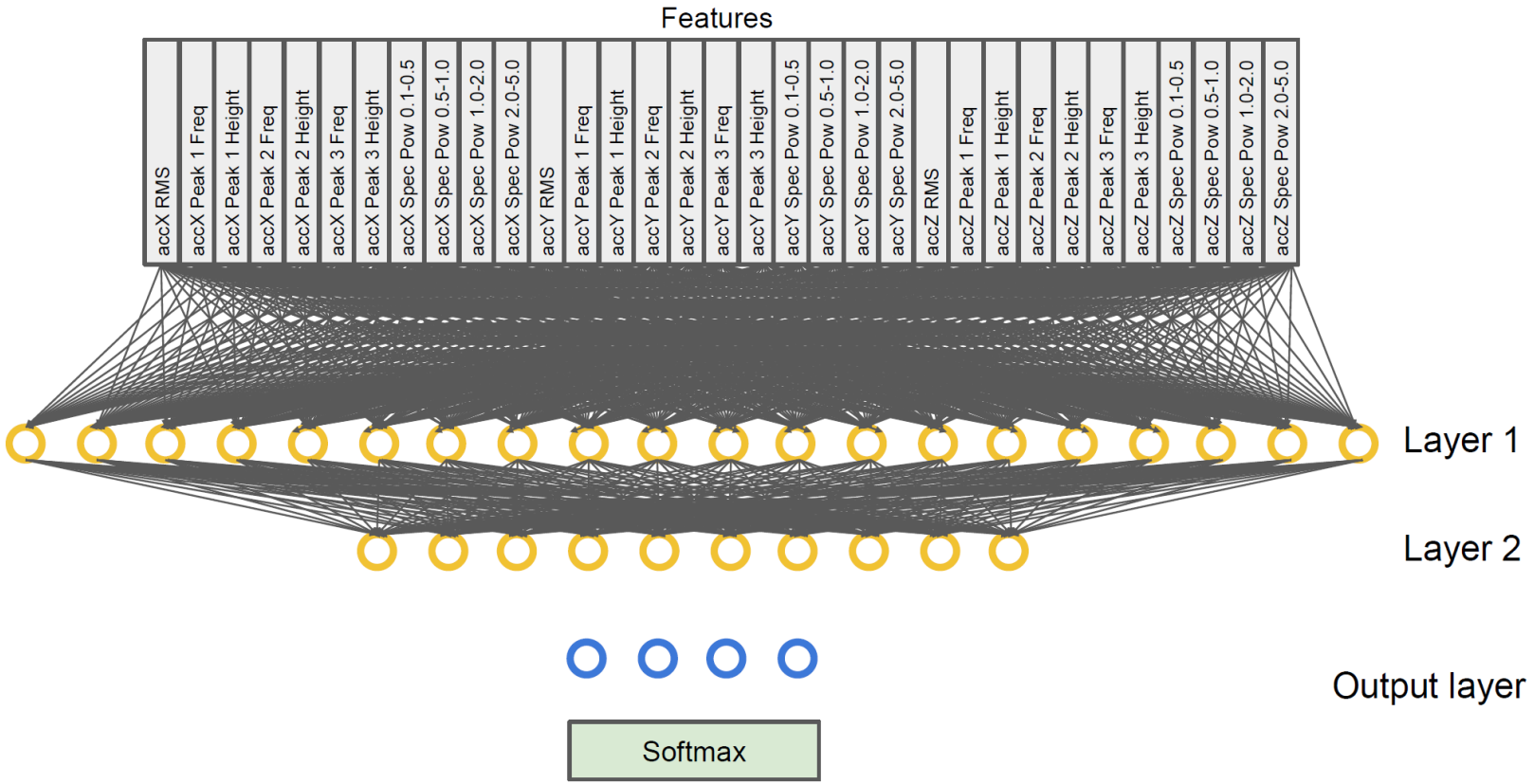
Final implementation



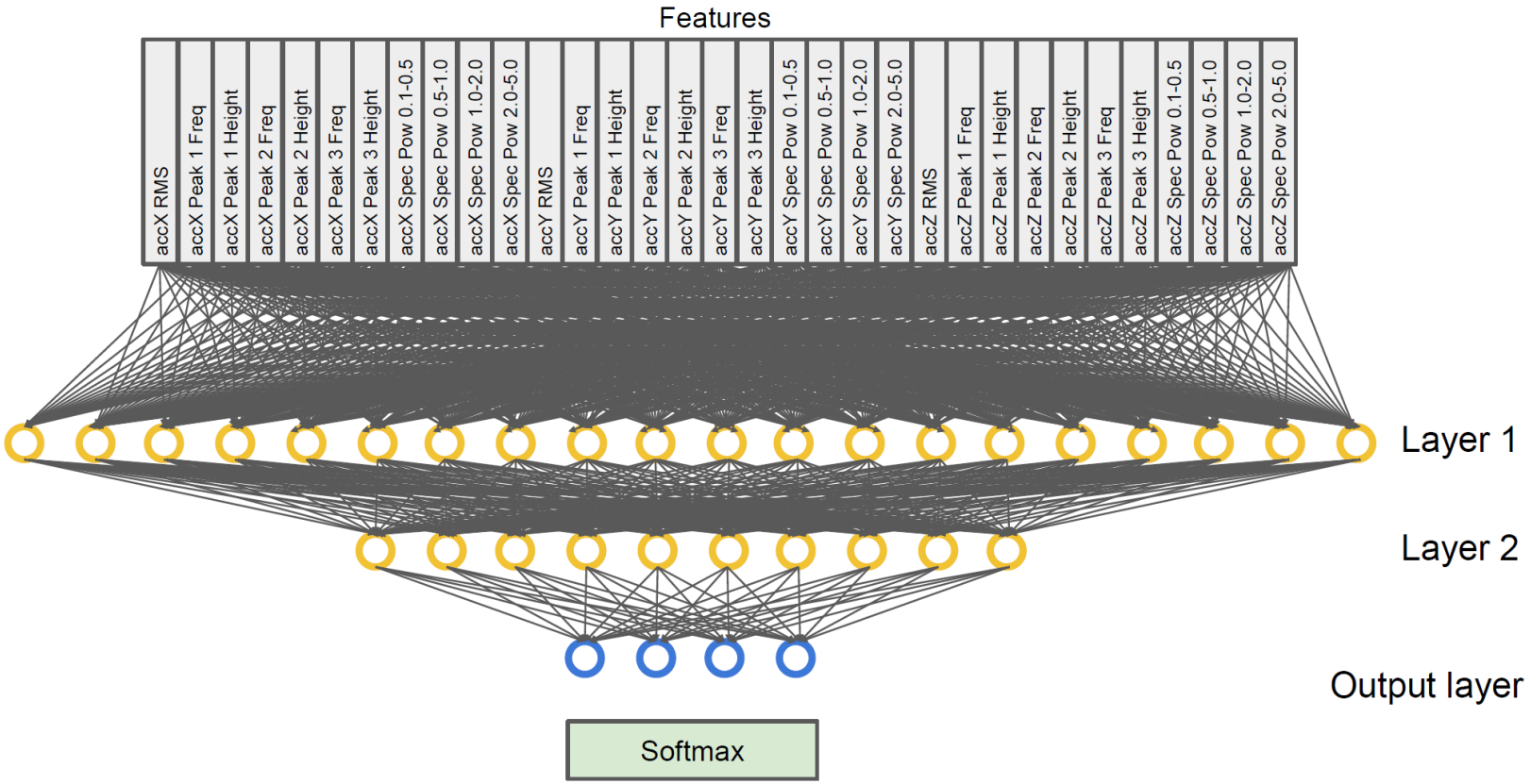
Final implementation



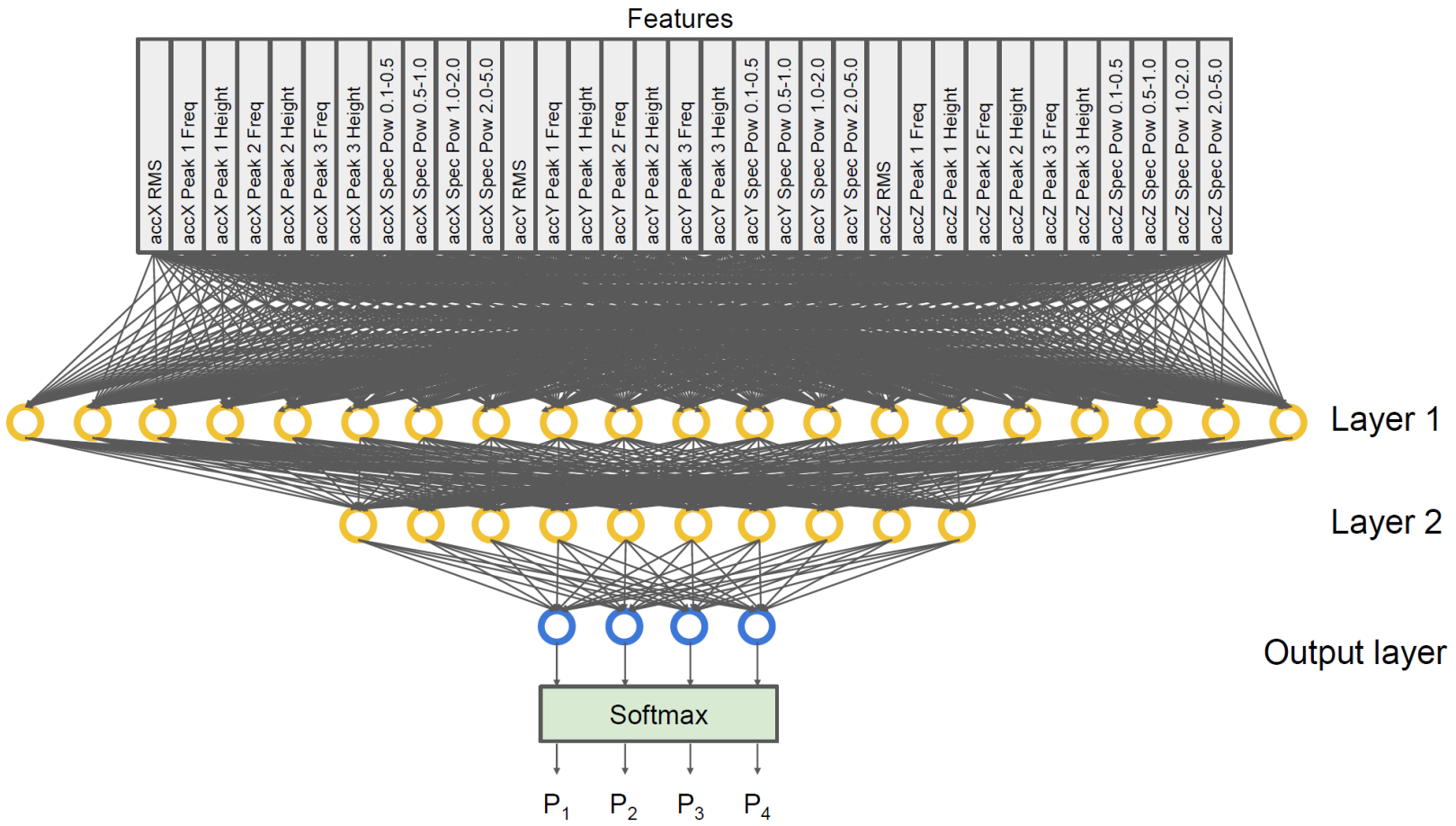
Final implementation



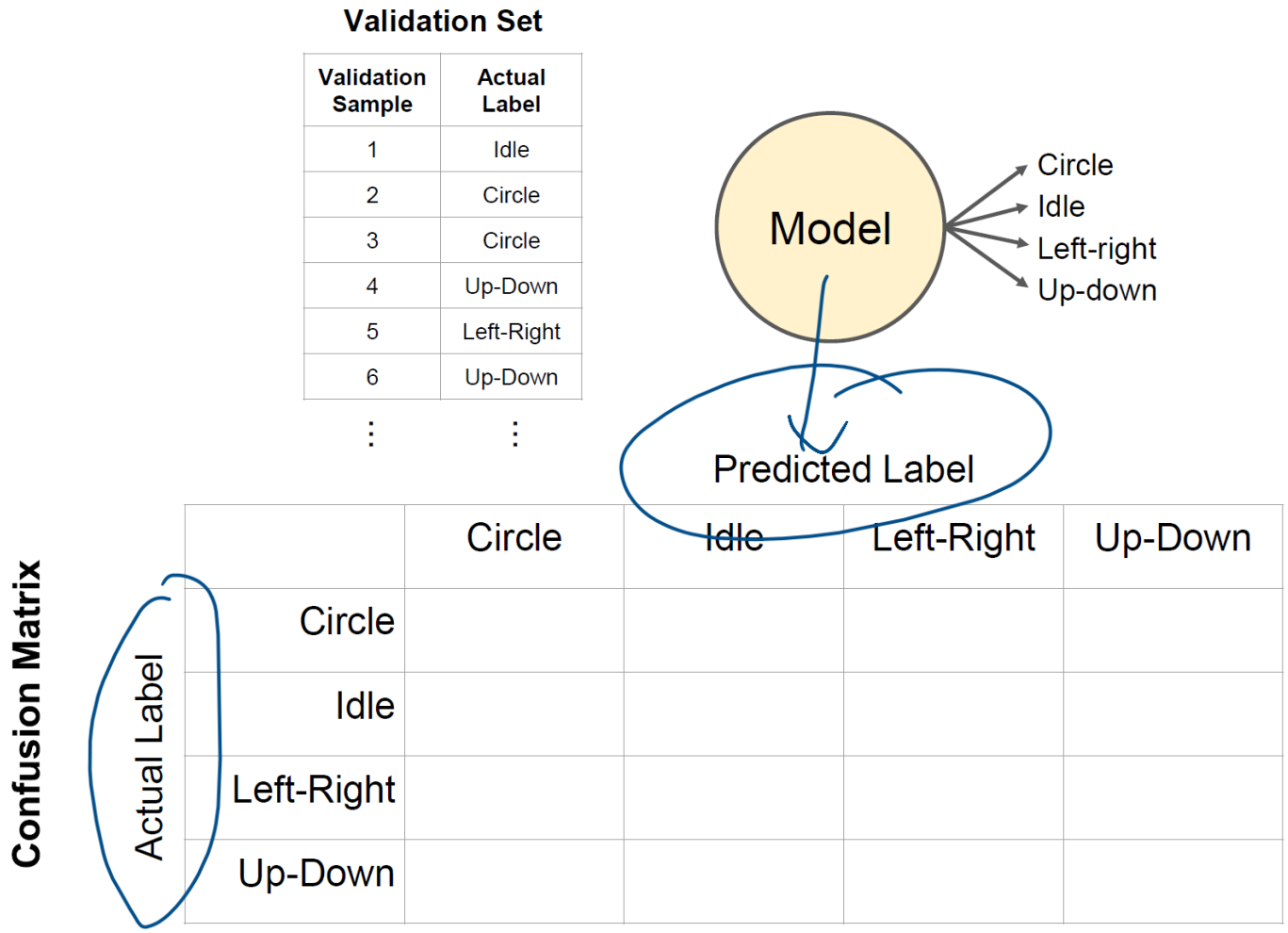
Final implementation



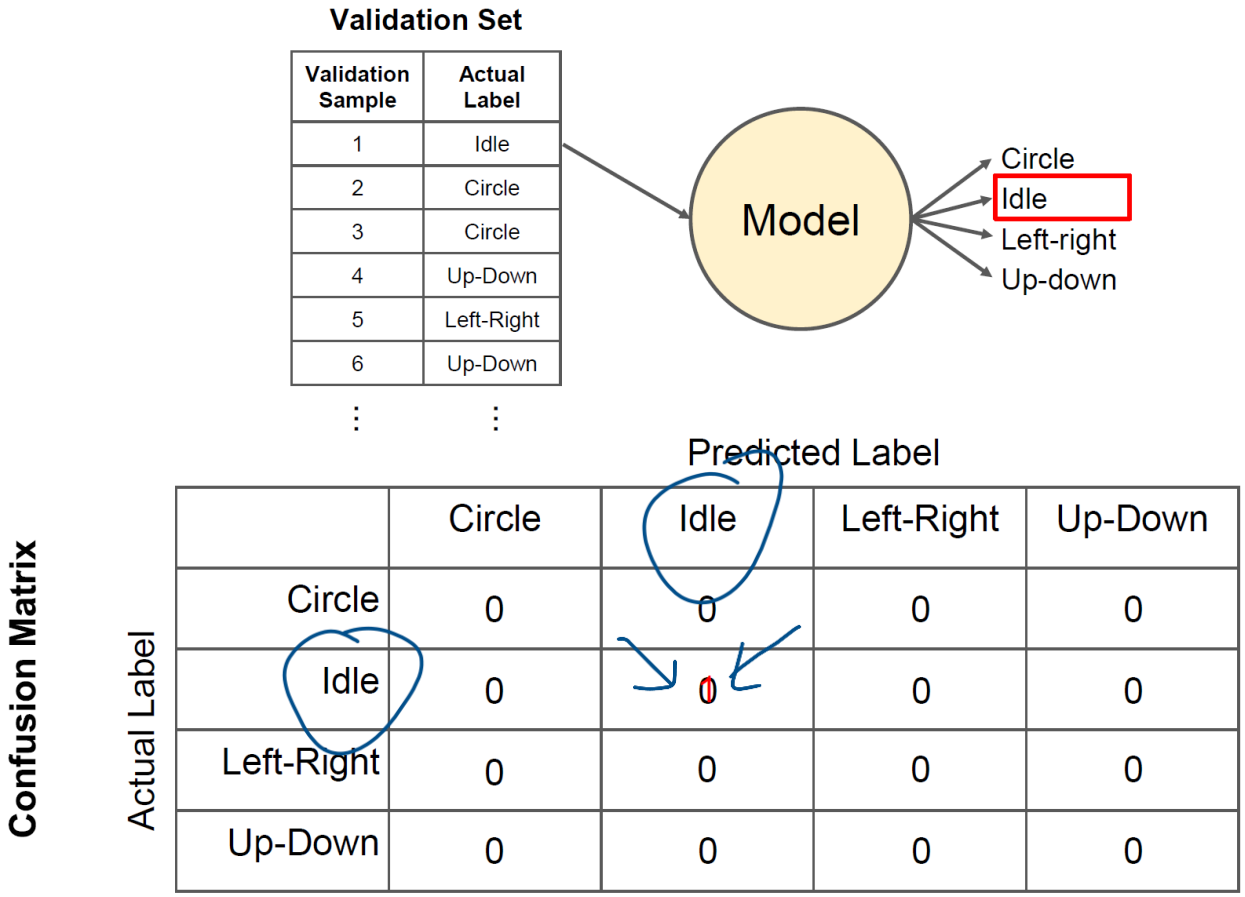
Final implementation



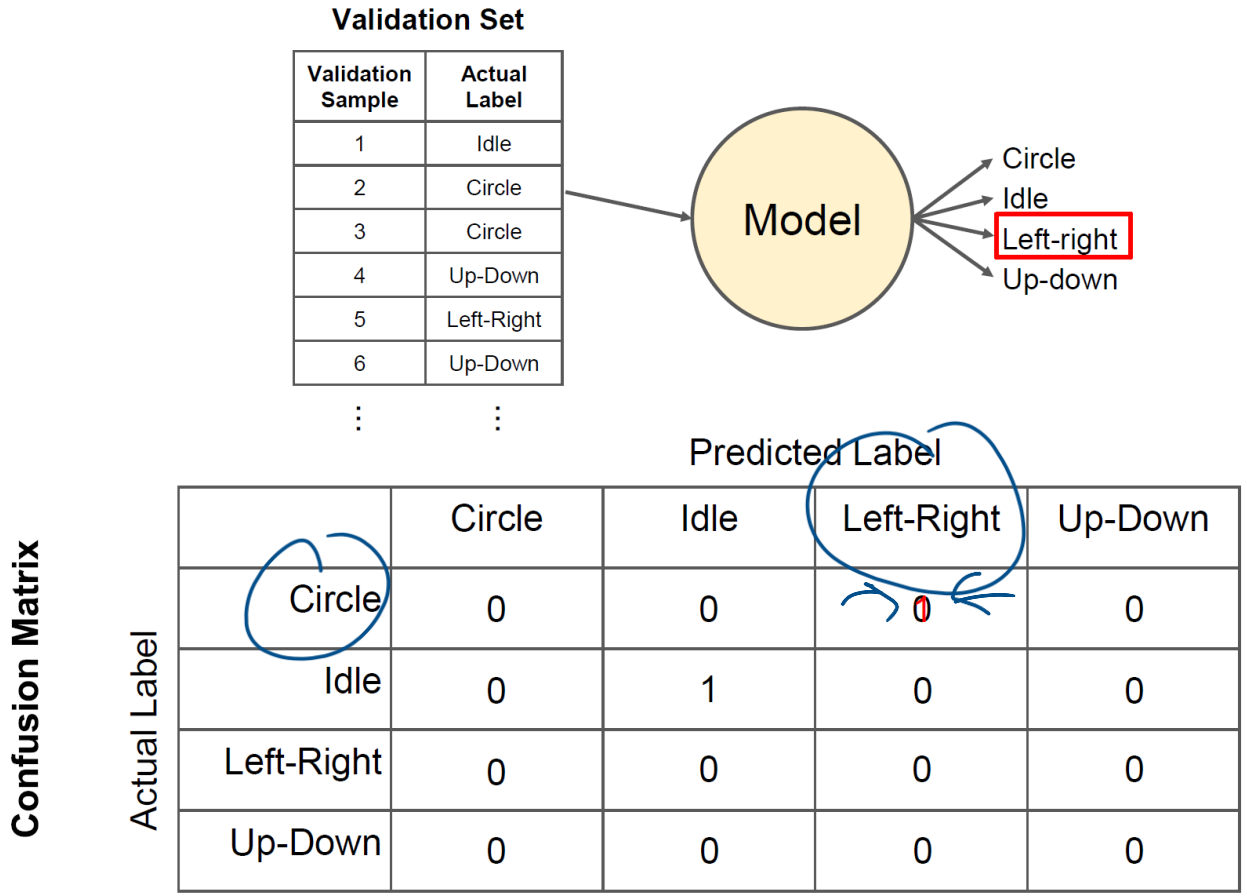
Confusion matrix in classification



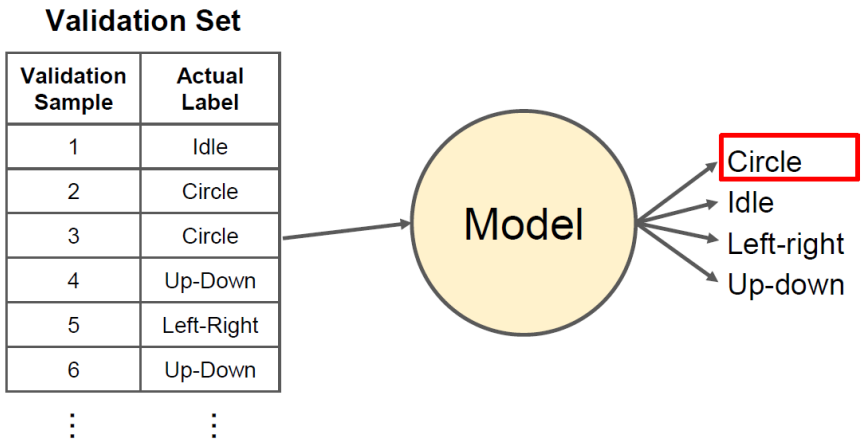
Confusion matrix in classification



Confusion matrix in classification



Confusion matrix in classification



Confusion Matrix

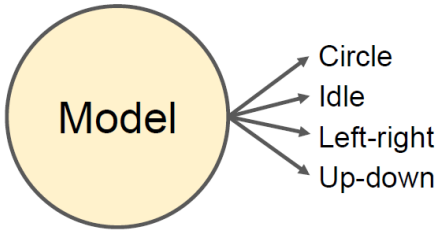
Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	0	0	1	0
Idle	0	1	0	0
Left-Right	0	0	0	0
Up-Down	0	0	0	0

Confusion matrix in classification

USC University of Southern California

Validation Set

Validation Sample	Actual Label
1	Idle
2	Circle
3	Circle
4	Up-Down
5	Left-Right
6	Up-Down
⋮	⋮



Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

Confusion matrix accuracy

Total accuracy: $$\frac{\sum correct}{\sum all} = \frac{813}{1000} = 0.813$$

Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

Is this a good dataset?

Total accuracy: $\frac{\sum correct}{\sum all} = \frac{973}{1000} = 0.973$

Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	0	6	0	0
Idle	0	973	0	0
Left-Right	0	11	0	0
Up-Down	0	10	0	0

Positives and negatives

Confusion Matrix

Actual Label \ Predicted Label	Positive	Negative
Positive	38	17
Negative	3	42

- True Positive (TP): Predicted positive matches actual positive
- True Negative (TN): Predicted negative matches actual negative
- False Positive (FP) ("Type I Error"): Predicted positive does not match actual negative
- False Negative (FN) ("Type II Error"): Predicted negative does not match actual positive

False positives, false negatives

Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

True Positive (TP): Predicted positive matches actual positive

True Negative (TN): Predicted negative matches actual negative

False Positive (FP) ("Type I Error"): Predicted positive does not match actual negative

False Negative (FN) ("Type II Error"): Predicted negative does not match actual positive

Accuracy for a single class

USC University of Southern California

Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

Accuracy

$$ACC = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{\square + \square}{\text{Total}}$$

$$= \frac{199 + 728}{1000} = 0.927$$

True Positive Rate, Sensitivity

Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

True Positive Rate (TPR), Sensitivity, Recall, Hit Rate

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} = \frac{\square}{\square + \square}$$

$$= \frac{199}{199 + 38} = 0.840$$

True Negative Rate, Selectivity

Confusion Matrix

Predicted Label

Actual Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

True Negative Rate (TNR), Specificity, Selectivity

$$TNR = \frac{TN}{N} = \frac{TN}{TN + FP} = \frac{\square}{\square + \square}$$

$$= \frac{728}{728 + 35} = 0.954$$

Precision (Positive Predictive Value)

Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

Positive Predictive Value (PPV), Precision

$$PPV = \frac{TP}{TP + FP} = \frac{\square}{\square + \square}$$

$$= \frac{199}{199 + 35} = 0.850$$

F1 score

Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

F1 Score

$$F_1 = 2 \cdot \frac{PPV \cdot TPR}{PPV + TPR} = \frac{2TP}{2TP + FP + FN}$$

$$= 2 \cdot \frac{0.850 \cdot 0.840}{0.850 + 0.840} = 0.845$$

Classifier metrics

Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	205	10	1	46
Idle	6	199	0	32
Left-Right	9	17	223	34
Up-Down	21	8	3	186

Per-class accuracy	0.907	0.927	0.936	0.856
F1 scores	0.815	0.845	0.875	0.721

Total accuracy: 0.813

F1 average: 0.818

F1 score is indicative of dataset balance

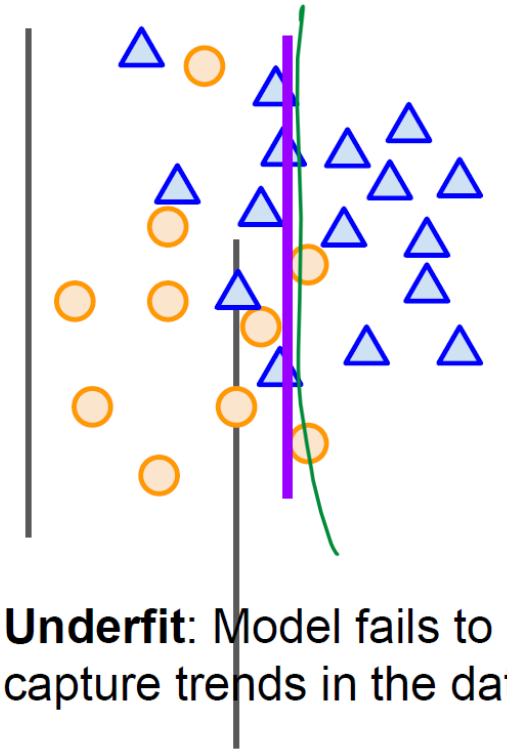
Confusion Matrix

Actual Label \ Predicted Label	Circle	Idle	Left-Right	Up-Down
Circle	0	6	0	0
Idle	0	973	0	0
Left-Right	0	11	0	0
Up-Down	0	10	0	0
Per-class accuracy	0.994	0.973	0.989	0.99
F1 scores	0.0	0.986	0.0	0.0

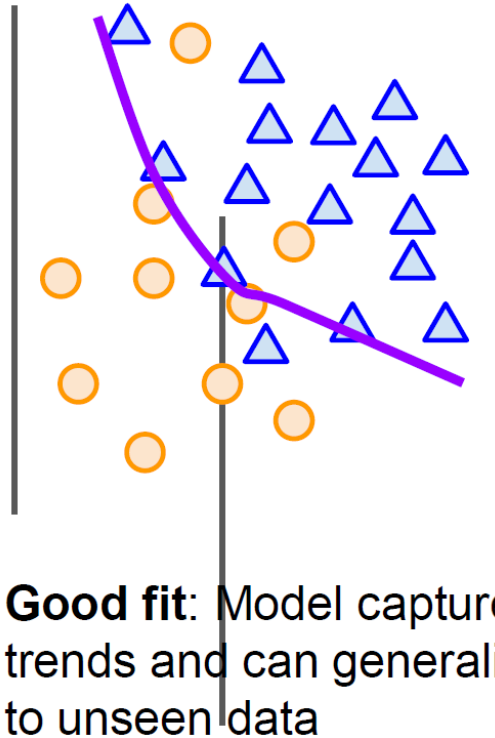
Total accuracy: 0.973

F1 average: 0.247

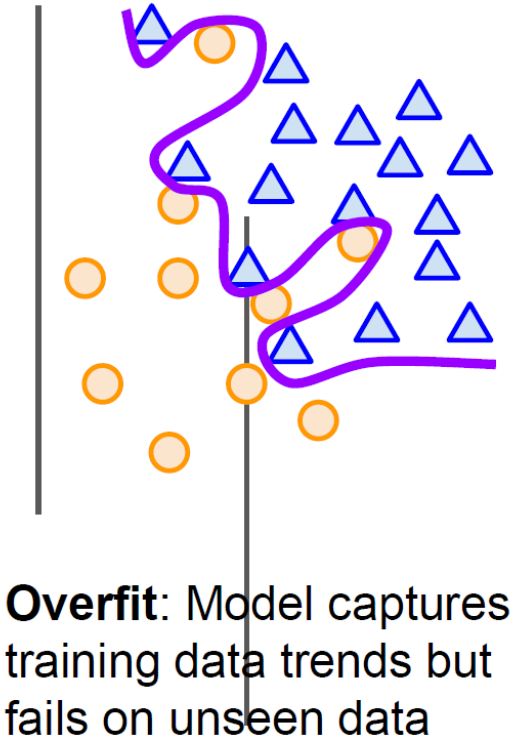
Model fit



Underfit: Model fails to capture trends in the data

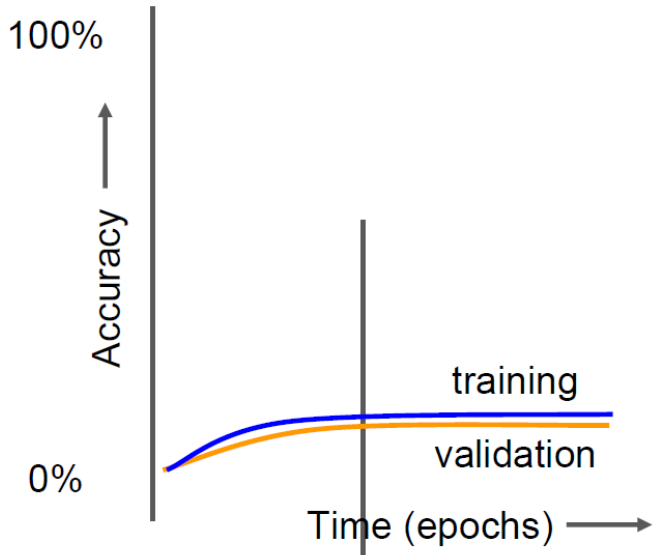


Good fit: Model captures trends and can generalize to unseen data

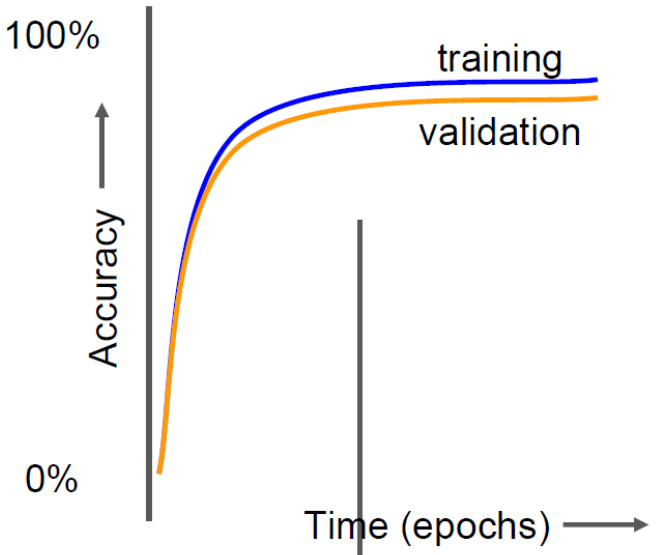


Overfit: Model captures training data trends but fails on unseen data

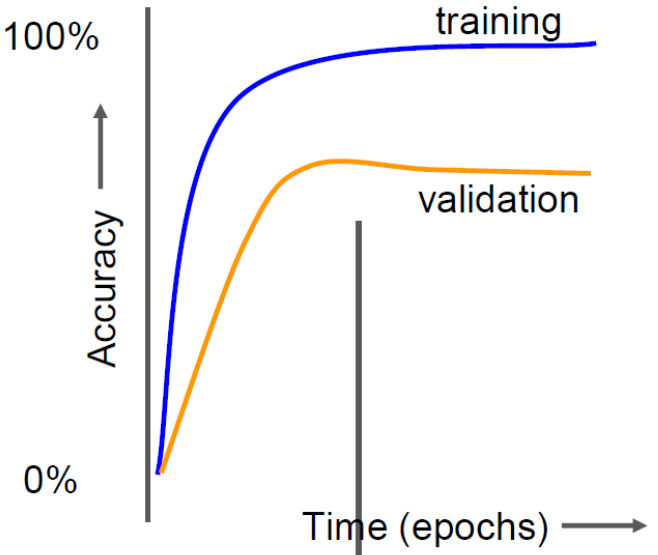
Accuracy vs epoch to understand dataset



Underfit: Model performs poorly on training and validation data

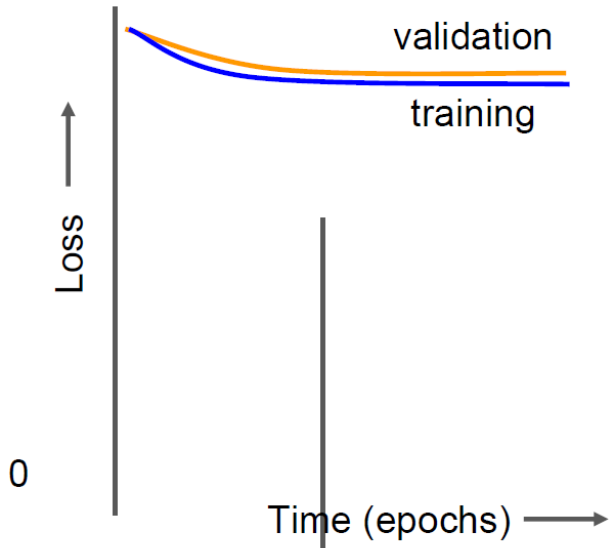


Good fit: Model generalizes well from training to validation data

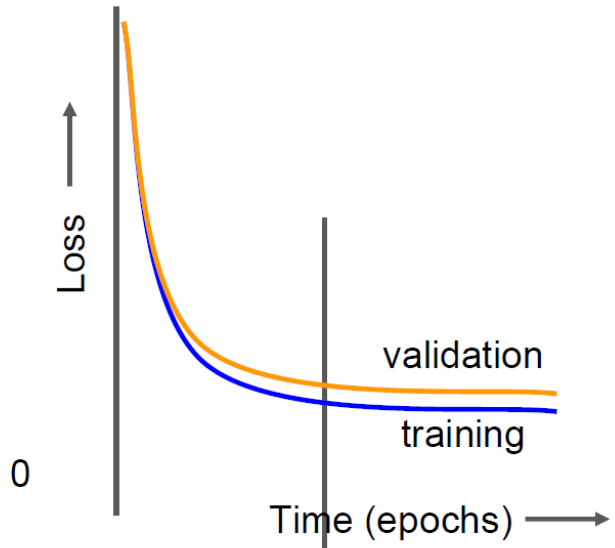


Overfit: Model predicts training data well but fails to generalize to validation data

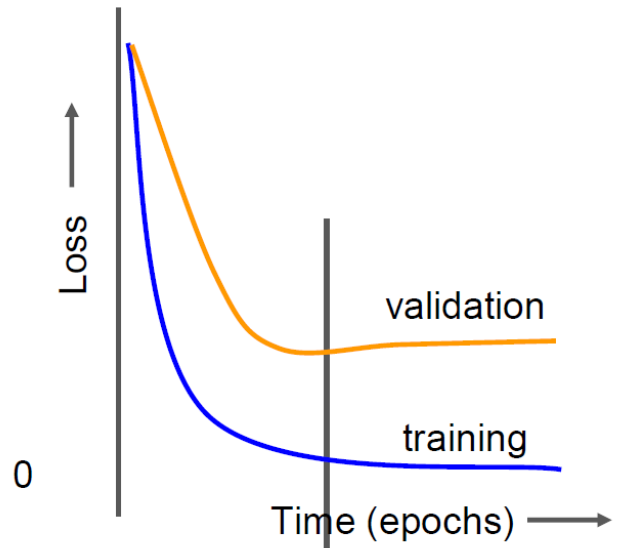
Loss vs epoch to understand dataset



Underfit: Model performs poorly on training and validation data

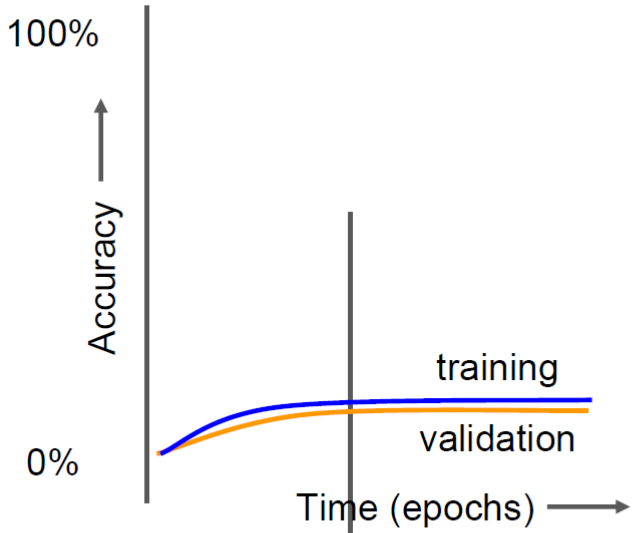


Good fit: Model generalizes well from training to validation data



Overfit: Model predicts training data well but fails to generalize to validation data

Fixing underfit

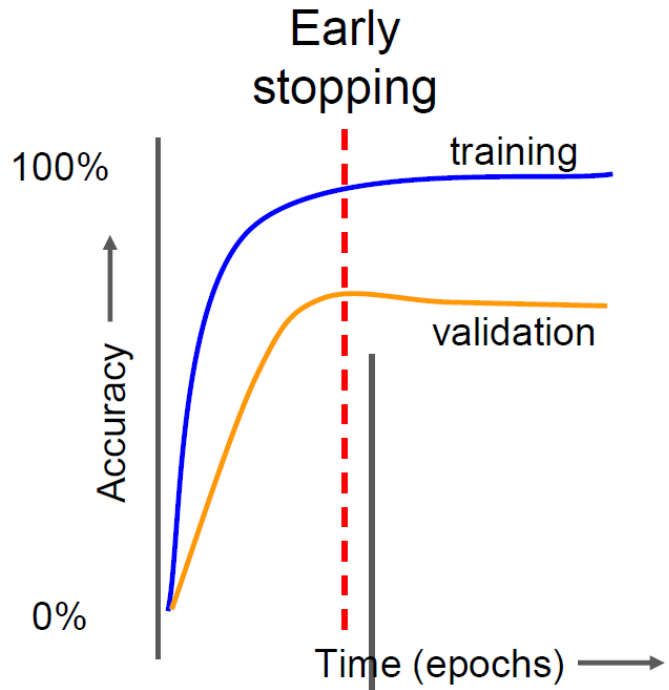


Underfit: Model performs poorly on training and validation data

- Get more data
- Try different features or more features
- Train for longer
- Try a more complex model (more layers, more nodes, etc.)

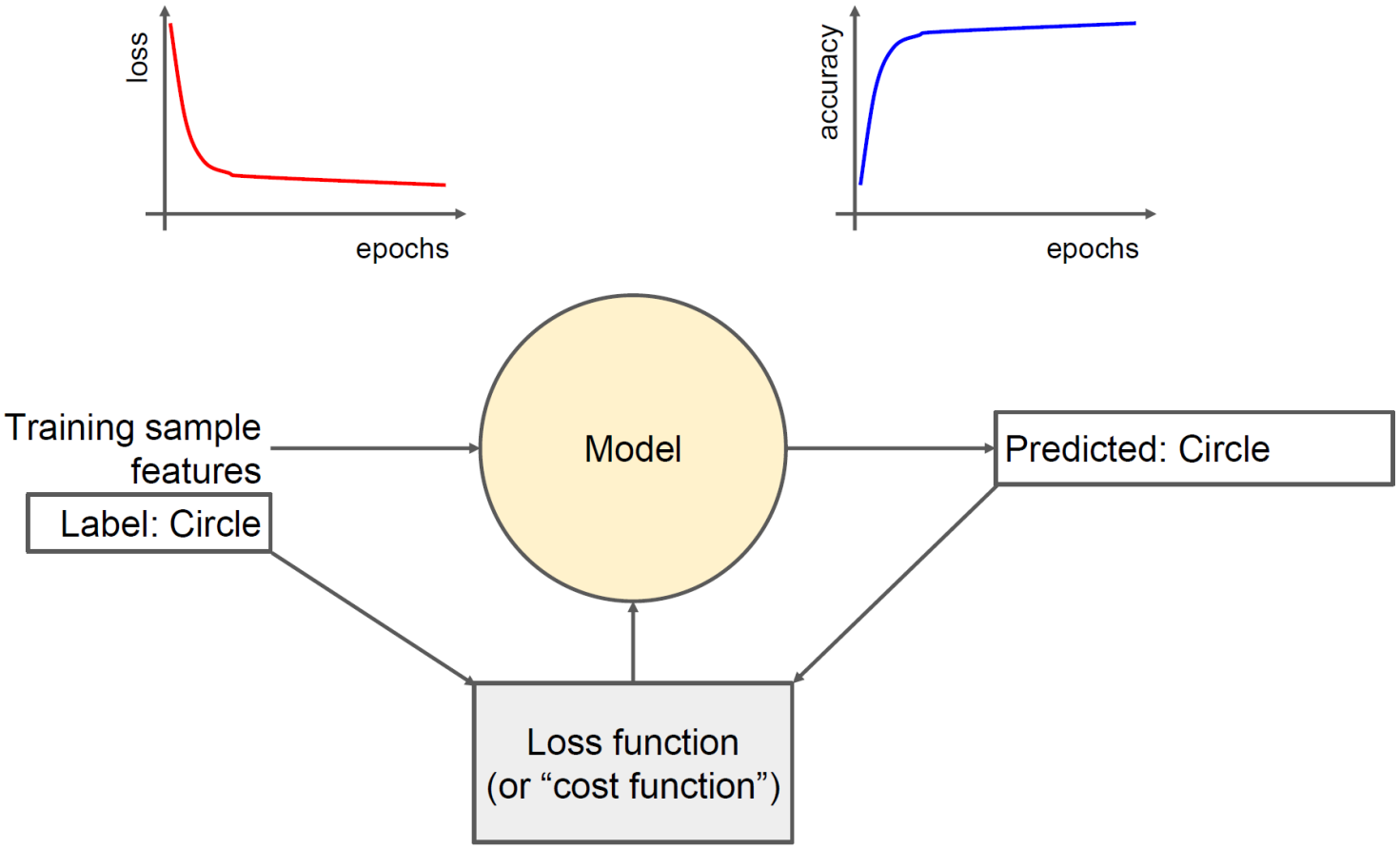
Fixing overfit

- Get more data
- Early stopping
- Reduce model complexity
- Add regularization terms
- Add dropout layers (for neural networks)



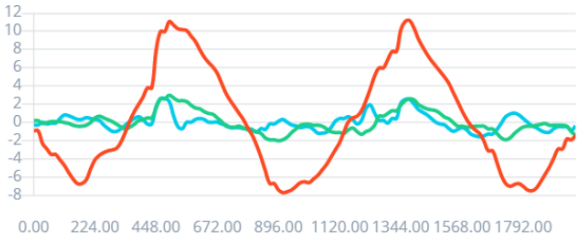
Overfit: Model predicts training data well but fails to generalize to validation data

Model characterization in realtime



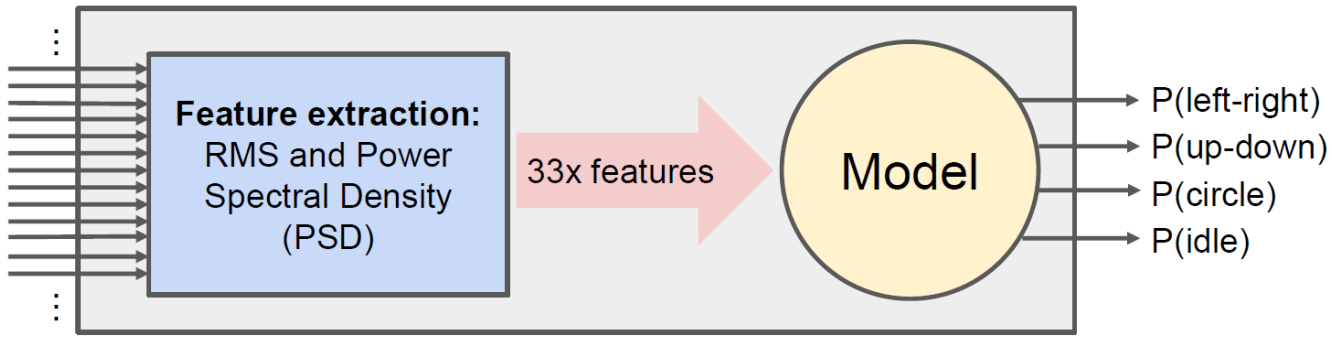
Output of the model

Sample accelerometer for 2 seconds



62.5 Hz sampling for 2 seconds with 3 axes = 375 values

Edge Impulse library



375 raw values

P(left-right)	= 0.9143
P(up-down)	= 0.0032
P(circle)	= 0.0581
P(idle)	= 0.0244

Action after classification

USC University of Southern California

```
if (p_left_right > 0.5) {
    // Do stuff
}

if (p_up_down > 0.5) {
    // Do some things
}

if (p_circle > 0.8) {
    // And now for something completely different
}
```