

Location, Gesture and Activity Sensing

Part IV: Inferring Activities from IMU

Department of Computer Science
and Engineering



INDIAN INSTITUTE OF TECHNOLOGY
KHARAGPUR

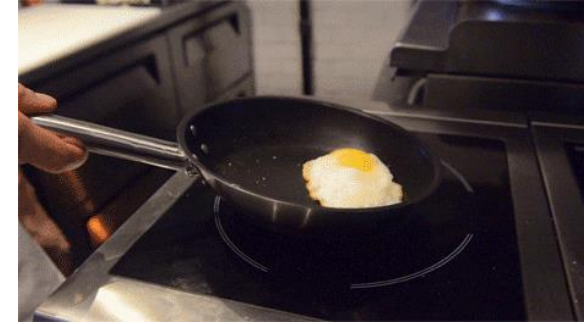
Sandip Chakraborty
sandipc@cse.iitkgp.ac.in

What We Have Learnt So Far?

- IMU sensors are noisy
 - Accelerometer measures true acceleration – needs to separate out the gravity component; however, gravity value gets polluted when the object is in motion
 - Integrations make the errors cumulative: Orientation and location estimations become noisy
 - Magnetic north pole can be used as an anchor, but it is also not free from noise
- Combine measurement logic with the application-enforced physics for gesture estimation
 - Statistical estimators/filters can help
 - Tradeoff between accuracy and real-time performance

The Next Steps – Gesture to Activities

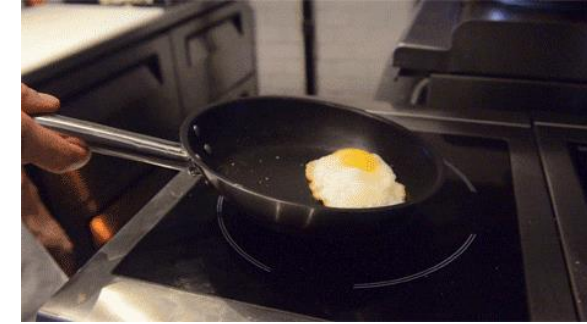
- Combine the gestures to infer the activities



Cooking

The Next Steps – Gesture to Activities

- Combine the gestures to infer the activities



However, the activity label can be more precise depending on the downstream application

**Preparing
poached egg**

Human Activity Recognition (HAR)

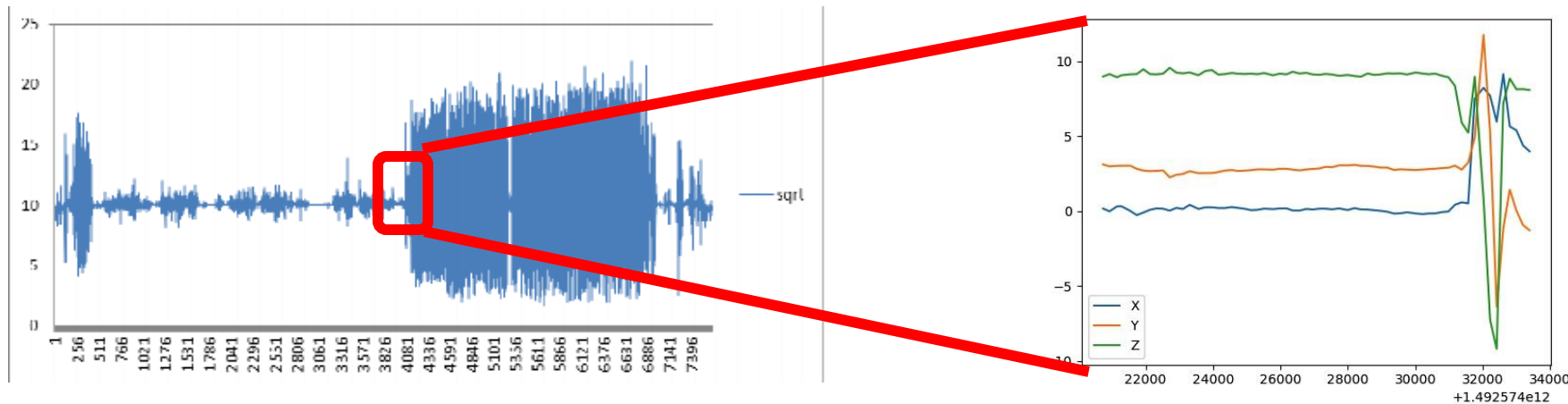
- Can use handcrafted statistical features from the gesture signatures
 - Mean, mode, median, kurtosis, etc.
- However, these features need to be extracted and engineered properly
 - Note that there is a fundamental difference between image/video processing and physical sensor data processing
 - IMU data is not visually recognizable like image or video data – so data annotation and model development is tricky
 - How do you understand which statistical features are more prominent for a specific type of activity patterns?
 - Also, activity signatures vary a lot across demography, age, gender, etc.

Human Activity Recognition (HAR)

- Can we use vision models for HAR?
 - The image representation of the accelerometer data and pass it to the vision-based model for activity classification?

Human Activity Recognition (HAR)

- Can we use vision models for HAR?
 - Take a matrix representation of the input: The image representation of the accelerometer data and pass it to the vision-based model for activity classification?
 - May lose a lot of contextual information depending on the resolution and granularity of the data – what would be the optimal resolution for the input?
 - Also, data from multiple modalities (accelerometer, gyroscope, compass) need to be combined to define the motion model properly

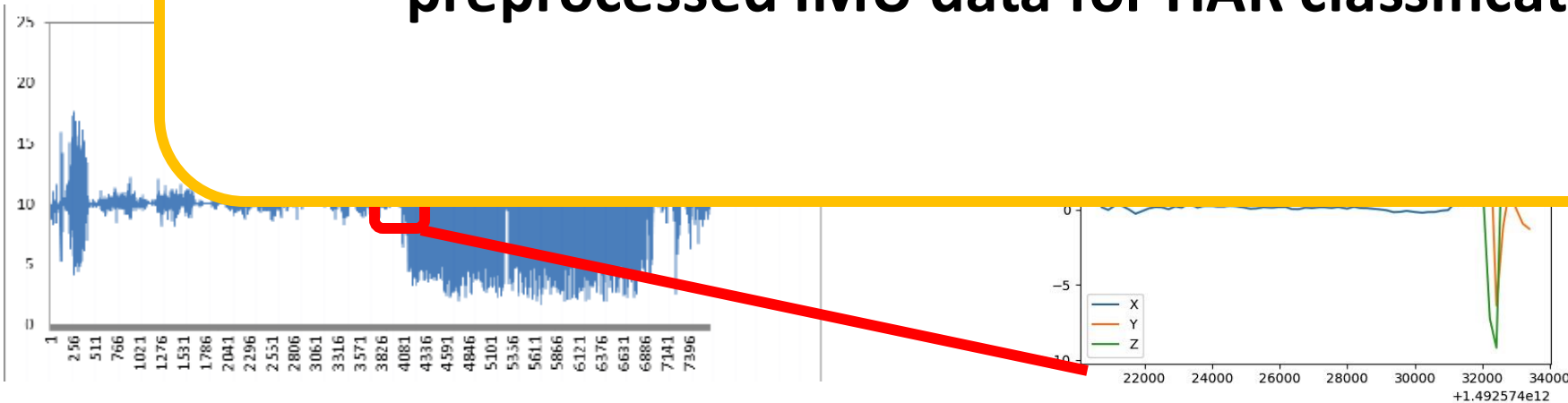


Human Activity Recognition (HAR)

- Can we use vision models for HAR?

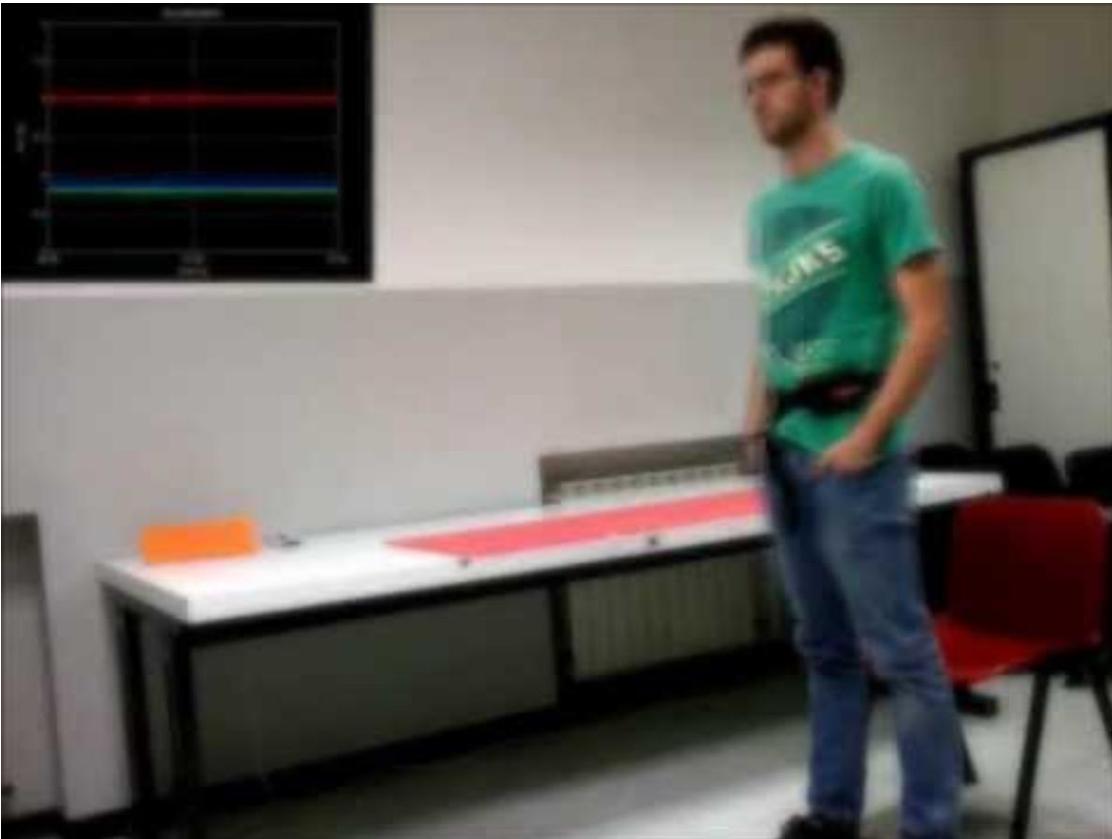
- Take a matrix representation of the input: The image representation of the accelerometer data and pass it to the vision-based model for activity classification?
- May I
- of th
- Also com

A Possible Solution: Use supervised models on the preprocessed IMU data for HAR classification



Deep Learning for HAR

- Represent the raw time-series data (after the necessary preprocessing and filtering) from the IMU as your input
 - No hand-crafted feature engineering is necessary



UCI-HAR dataset, several activity classes:

- Walking
- Walking upstairs
- Walking downstairs
- Sitting
- Standing
- Laying

<https://archive.ics.uci.edu/dataset/240/human+activity+recognition+using+smartphones>

Deep Learning for HAR

- 1D-CNN to predict the feature classes
 - Can learn from the raw time series data directly (accelerometer and gyroscope)
 - Do not require domain expertise for manual feature engineering
- **Input:** three time-series signals (3-DoF representation)
 - Total acceleration (gravitational acceleration + body acceleration)
 - Body acceleration
 - Body gyroscope
- Window the input data – one sample is one window
 - 128 time-steps
- Multi-class classification
 - Adam optimizer, categorical cross entropy loss
 - Tune hyperparameters (kernel size, number of filters) to best fit the model

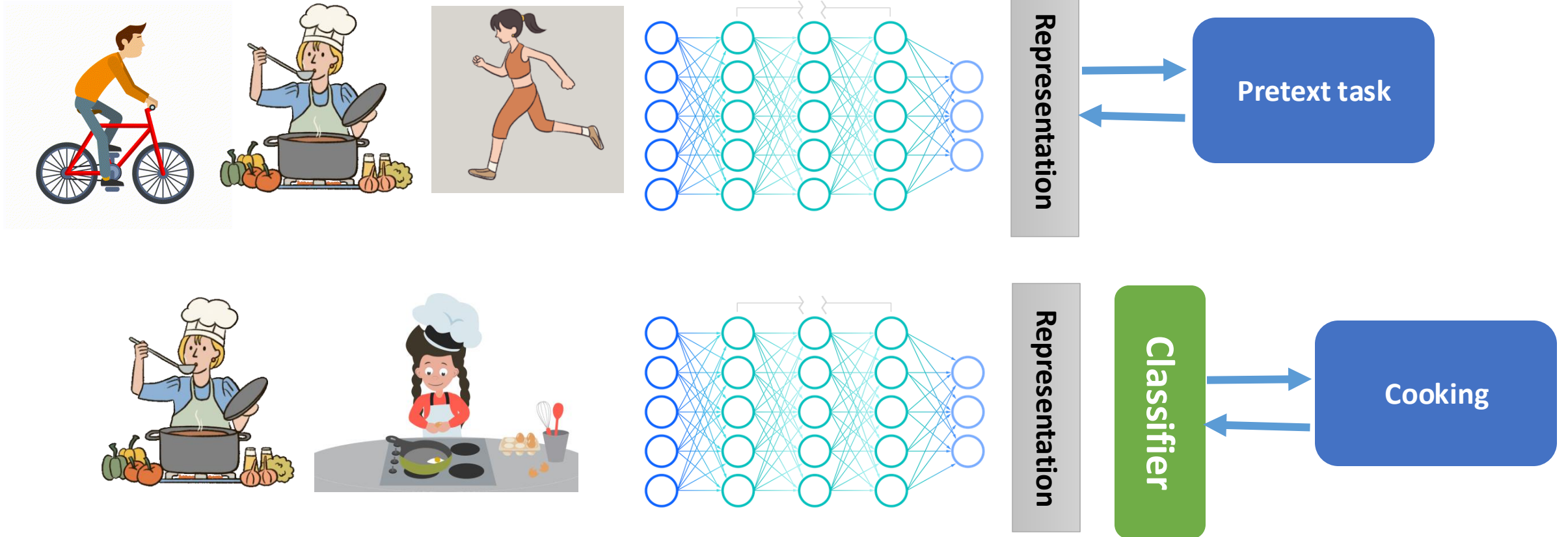
Deep Learning for HAR

- 1D-CNN to predict the feature classes
 - Can learn from the raw time series data directly (accelerometer and gyroscope)
 - Do not require domain expertise for manual feature engineering
- **Input:** three time-series signals (3-DoF representation)
 - Total acceleration (gravitational acceleration + body acceleration)
 - Body acceleration
 - Body gyroscope
- Window the data
 - 128 time-samples
- Multi-class classification
 - Adam optimizer, categorical cross entropy loss
 - Tune hyperparameters (kernel size, number of filters) to best fit the model

Cons: You still need a significant amount of labelled data to train your model

Self-Supervised Learning (SSL) for HAR

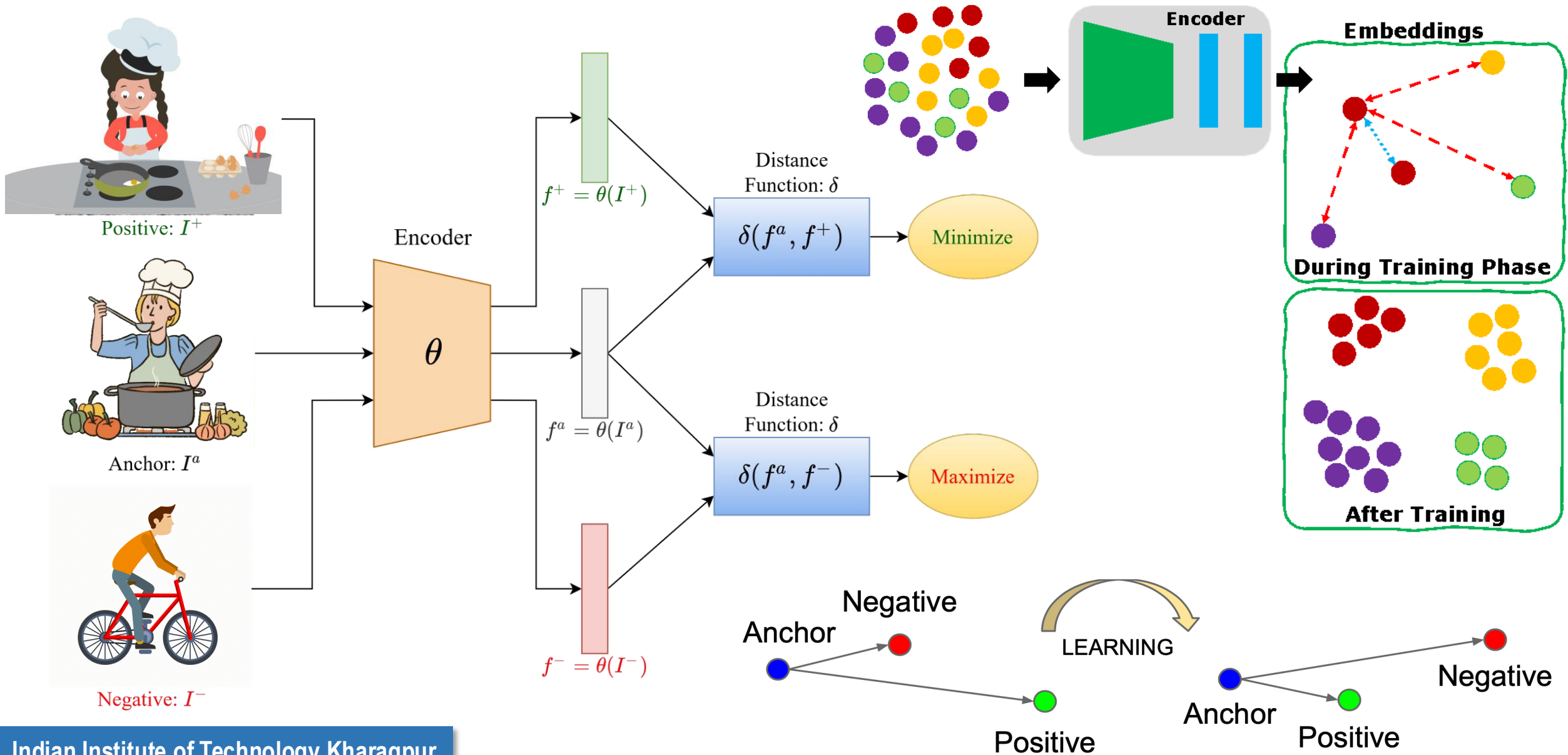
- Self-supervised Learning (SSL)
 - Does not rely on the labelled dataset for supervisory signals
 - Generate implicit labels from the unstructured data



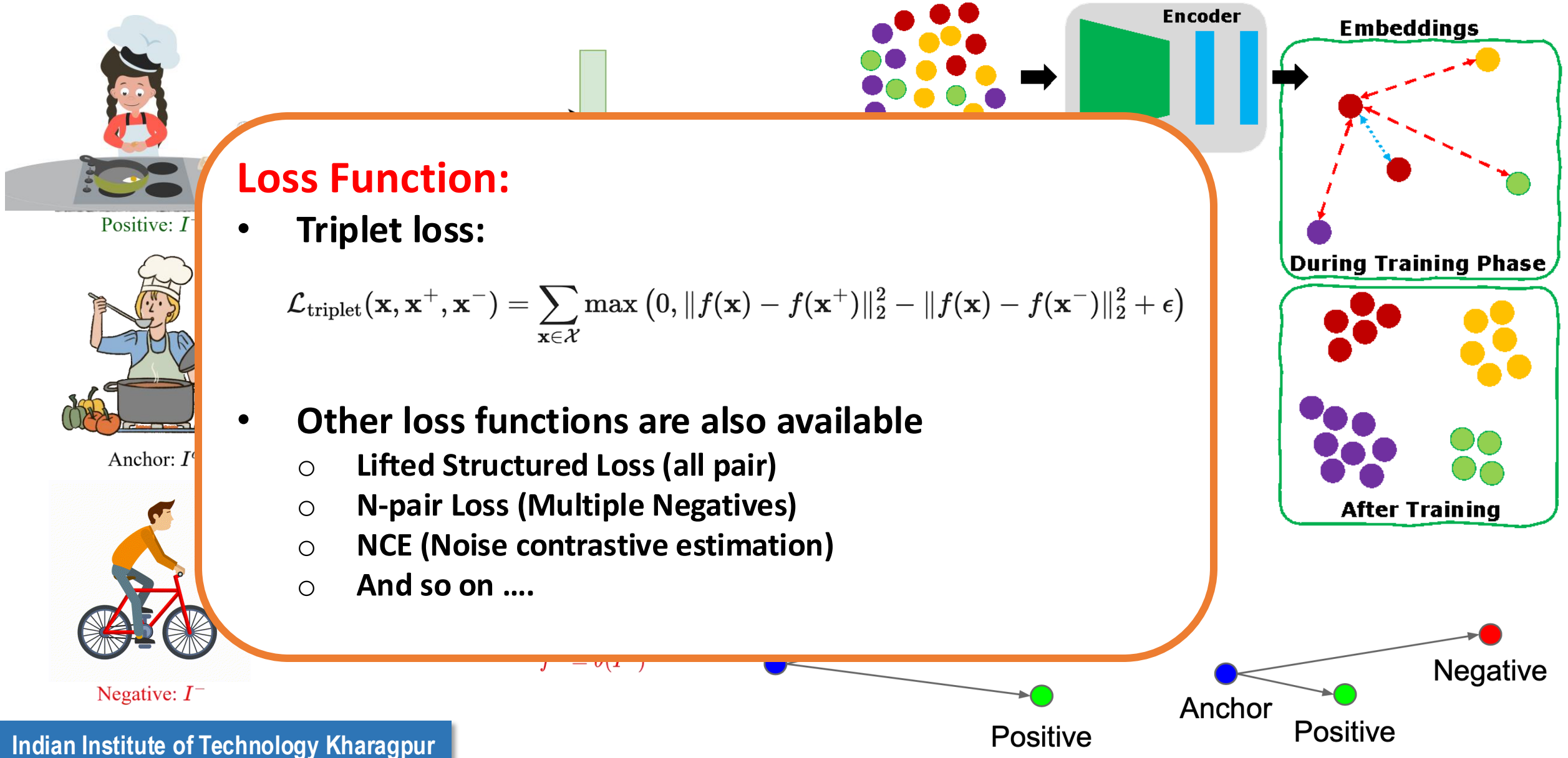
Contrastive Learning

- Pick up an anchor – the target that you want to learn
 - Positive (X^+): Samples that are similar to the anchor (similar semantic meaning)
 - Negative (X^-): Samples that are different from the anchor
- Pull Anchor and Positive embeddings closer and push the Anchor and Negative embeddings further

Contrastive Learning

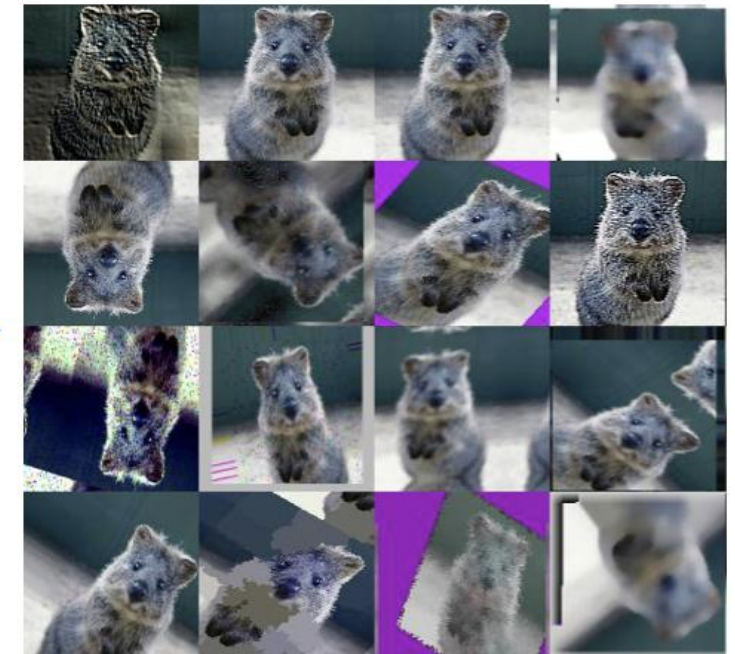


Contrastive Learning



Contrastive Learning

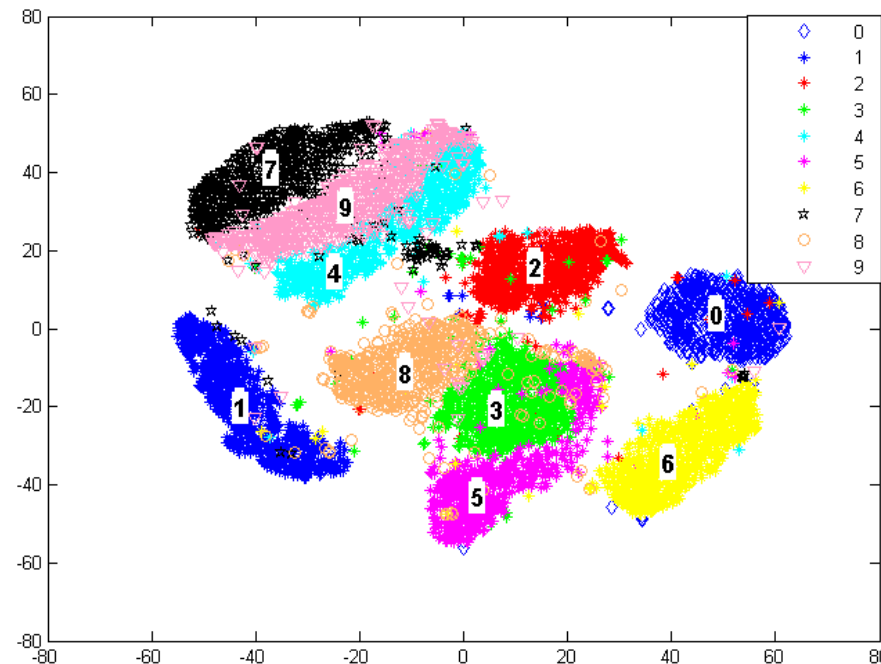
- How do you get the positive samples without labeling?
 - **Positive:** Pick up a few anchors and apply augmentations
 - **Negative:** Everything else
- Contrastive learning is popular in the image domain
 - Augmentation is easy
 - Well defined semantic augmentations that do not change the meaning of the subjects
 - Flip
 - Rotate
 - Color distortion
 - Random crop
 - Zooming
 - Blur, and so on ...



Slide credit:
Prasenjit Karmakar

MNIST Representations

- Learned M dimensional Embedding Vector for which below properties holds
 - Two similar images are embedded closely in the M dimensional space as compared to two dissimilar images, forming clusters.
 - Two such cluster should have minimal to no overlap (GOOD for downstream classification)

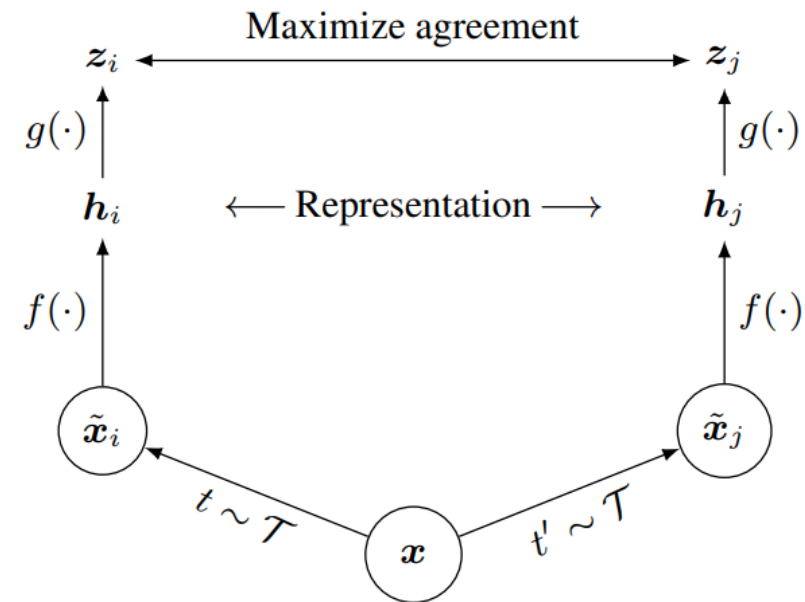


Two dimensional MNIST representations

Slide credit:
Prasenjit Karmakar

SimCLR

- One of the simplest but effective contrast learning framework for vision-based classification

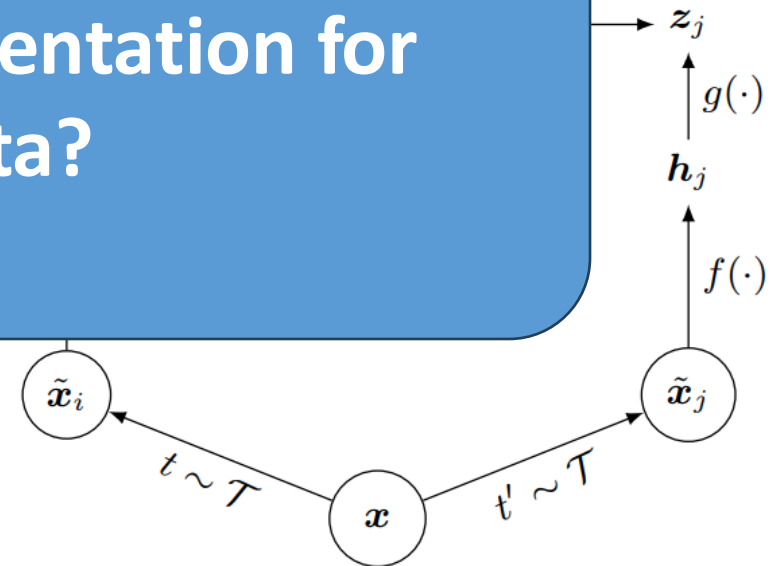


Gif source: <https://medium.com/analytics-vidhya/eli5-a-simple-framework-for-contrastive-learning-of-visual-representations-20d9509d0a12>

SimCLR

- One of the simplest but effective contrast learning framework for

How do we apply data augmentation for physical sensing data?



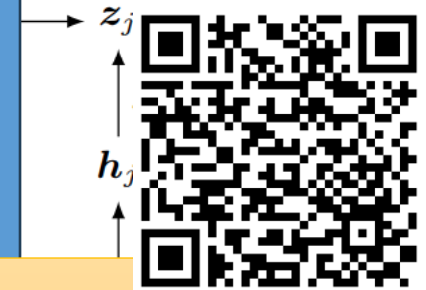
Gif source: <https://medium.com/analytics-vidhya/eli5-a-simple-framework-for-contrastive-learning-of-visual-representations-20d9509d0a12>

SimCLR

- One of the simplest but effective contrast learning framework for

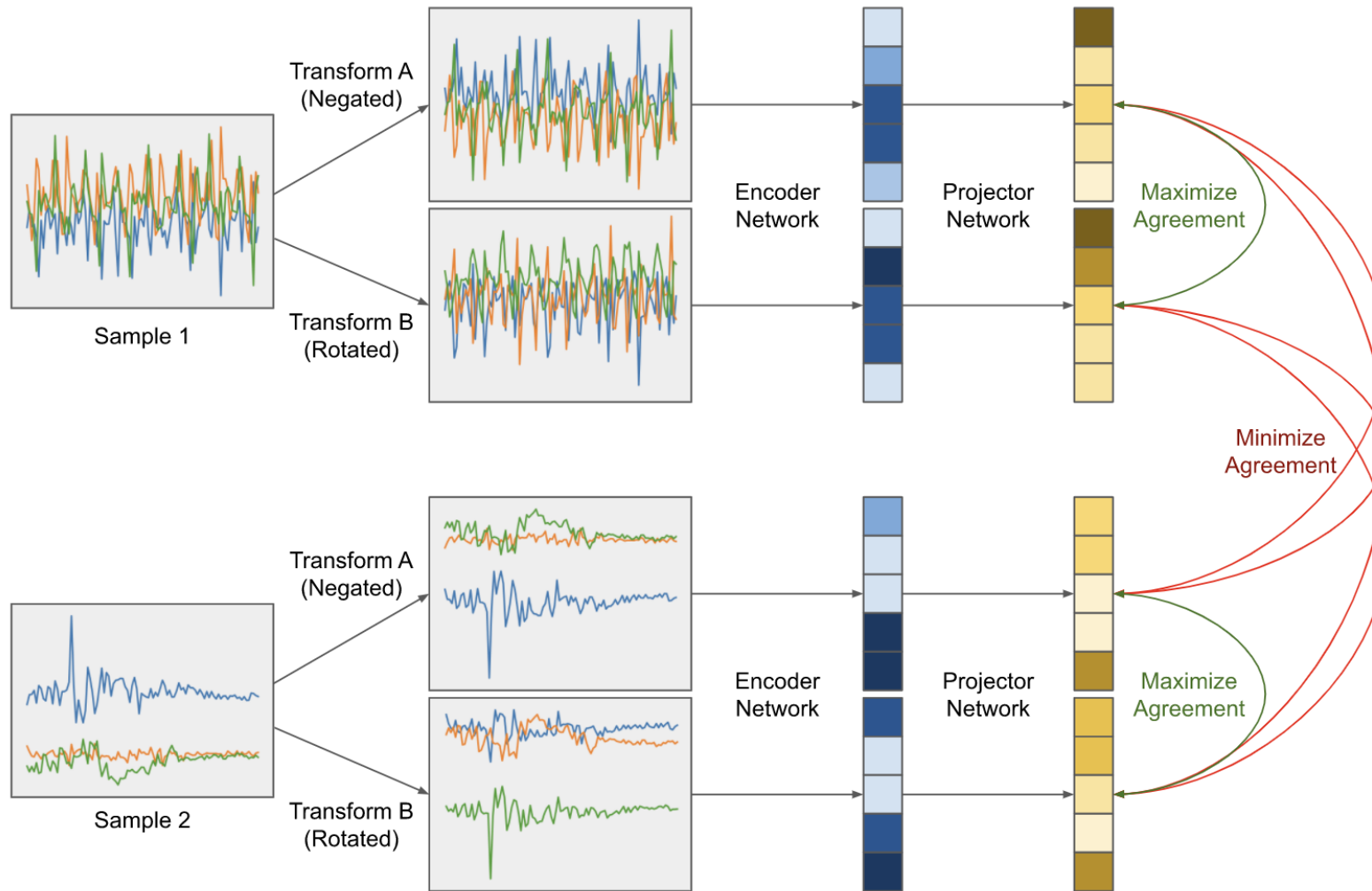
How do we apply data augmentation for physical sensing data?

Apply the laws of physics



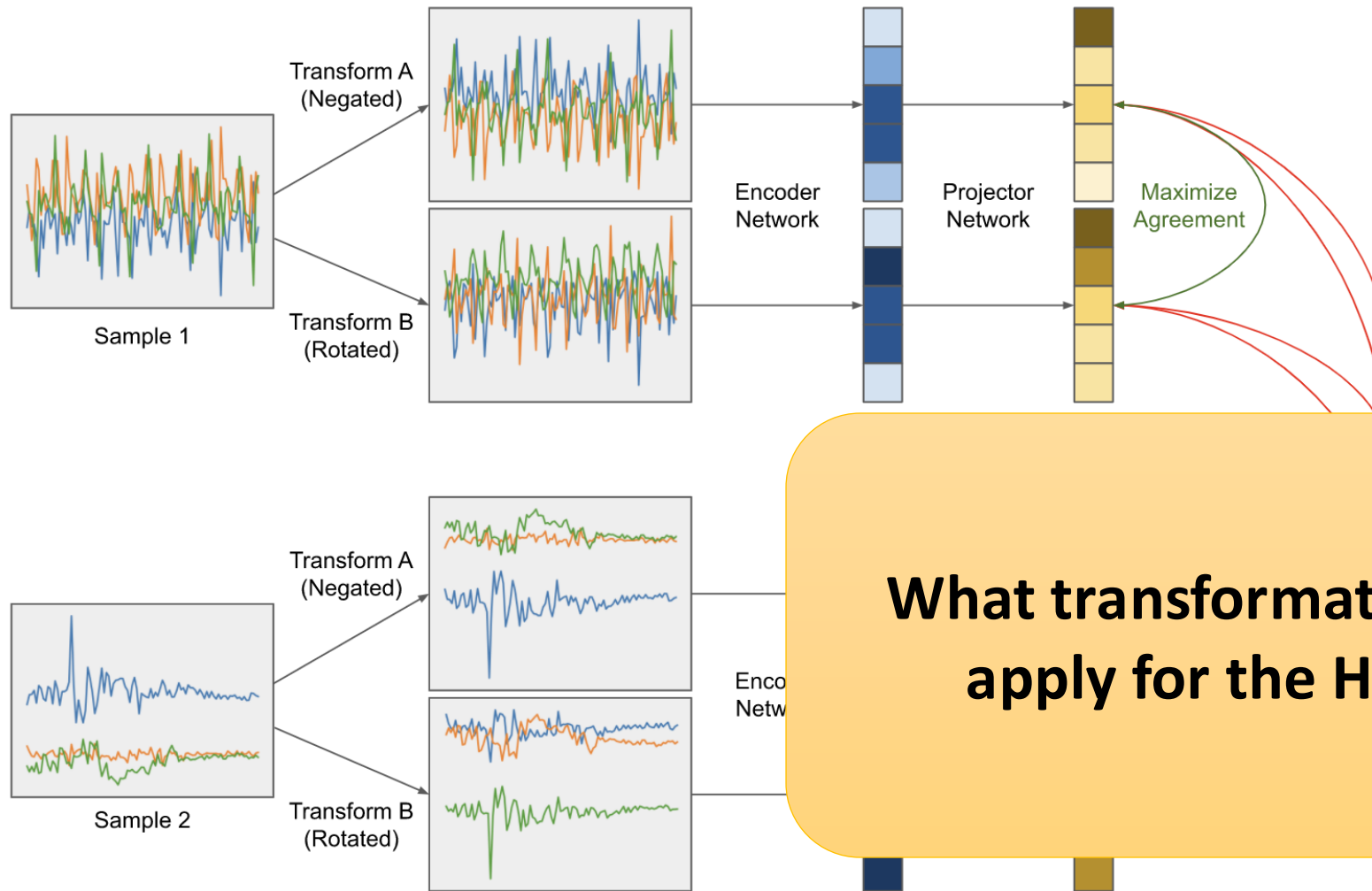
Git
fra
20d9509d0a12

Contrastive Learning for HAR



Adoption of SimCLR framework for HAR

Contrastive Learning for HAR



Adoption of SimCLR framework for HAR

Contrastive Learning for HAR

ColloSSL: Collaborative Self-Supervised Learning for Human Activity Recognition

YASH JAIN*, Georgia Tech, United States of America

CHI IAN TANG*, University of Cambridge, United Kingdom

CHULHONG MIN, Nokia Bell Labs, United Kingdom

FAHIM KAWSAR, Nokia Bell Labs, United Kingdom

AKHIL MATHUR, Nokia Bell Labs, United Kingdom

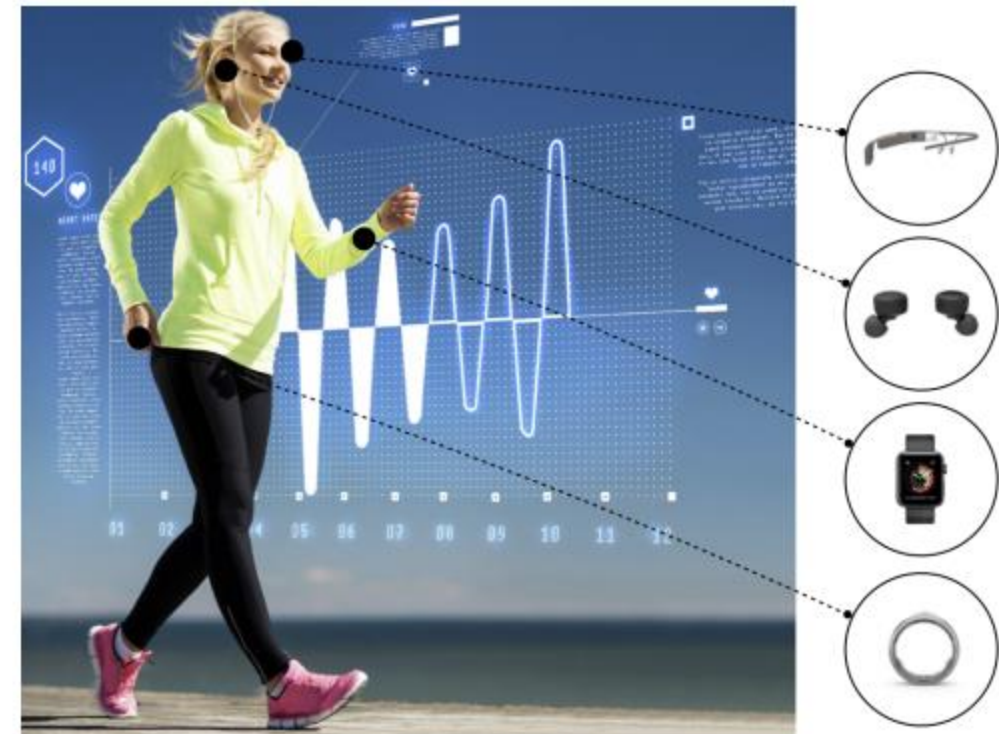


<https://github.com/iantangc/ContrastiveLearningHAR>

IMWUT 2022

Time Synchronous Multi Device Systems (TSMDS)

- Multiple computing/sensor devices are worn on, affixed to or implanted in a person's body
 - All devices observe a physical phenomenon (e.g., a user's locomotion activity) simultaneously and record sensor data in a time-aligned manner
- Assumptions
 - All sensor devices in the multi-device system share the same sensor sampling rate (or that their data can be re-sampled to the same rate)
 - Multiple devices in collect sensor data in a time-aligned manner (minor variations do not impact the performance significantly)



Transformations for Data Augmentation

- Rather than applying *manual transformations* (like rotation), can we utilize *natural transformations* that are already available in the sensor datasets
 - Multiple sensors in a TSMDS system captures the same activity under different view



Transformations for Data Augmentation

- Rather than applying *manual transformations* (like rotation), can we utilize *natural transformations* that are already available in the sensor datasets
 - Multiple sensors in a TSMDS system captures the same activity under different view



Transformations for Data Augmentation

- Rather than applying *manual transformations* (like rotation), can we utilize *natural transformations* that are already available in the sensor datasets
 - Multiple sensors in a TSMDS system captures the same activity under different view
- Accelerometer and Gyro at different body parts capture the translational and the rotational motions of those body parts
 - For an activity, the motions at different body parts are interlinked
 - Think about the motion patterns of hands and legs when you are running!



Transformations for Data Augmentation

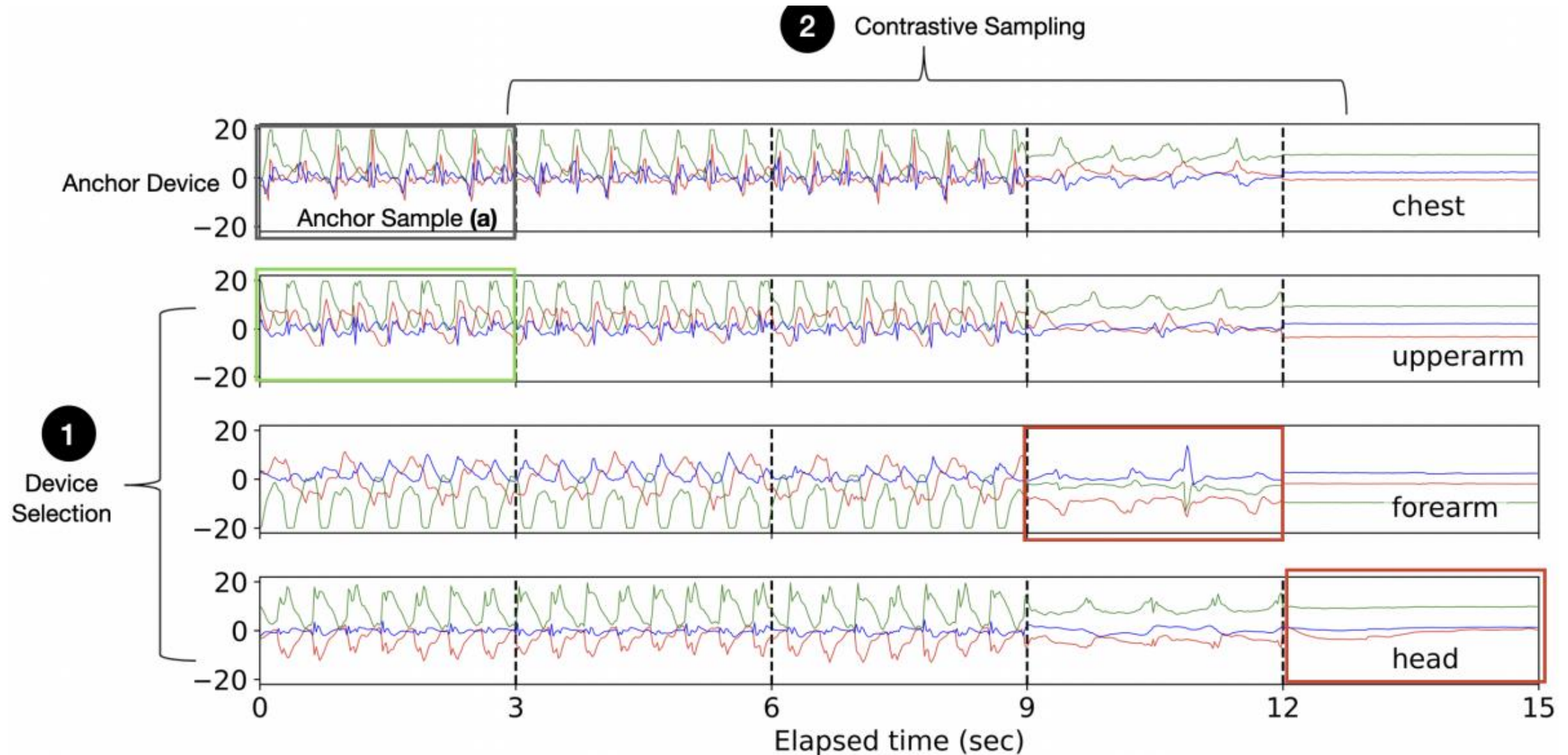
- Rather than applying *manual transformations* (like rotation), can we utilize *natural transformations* that are already available in the sensor datasets
 - Multiple sensors in a TSMDs system captures the same activity under different view
- Accelerometer and Gyro at different body parts capture the translational and the rotational motions of those body parts

Use such natural transformations to define a pretext task and perform contrastive learning

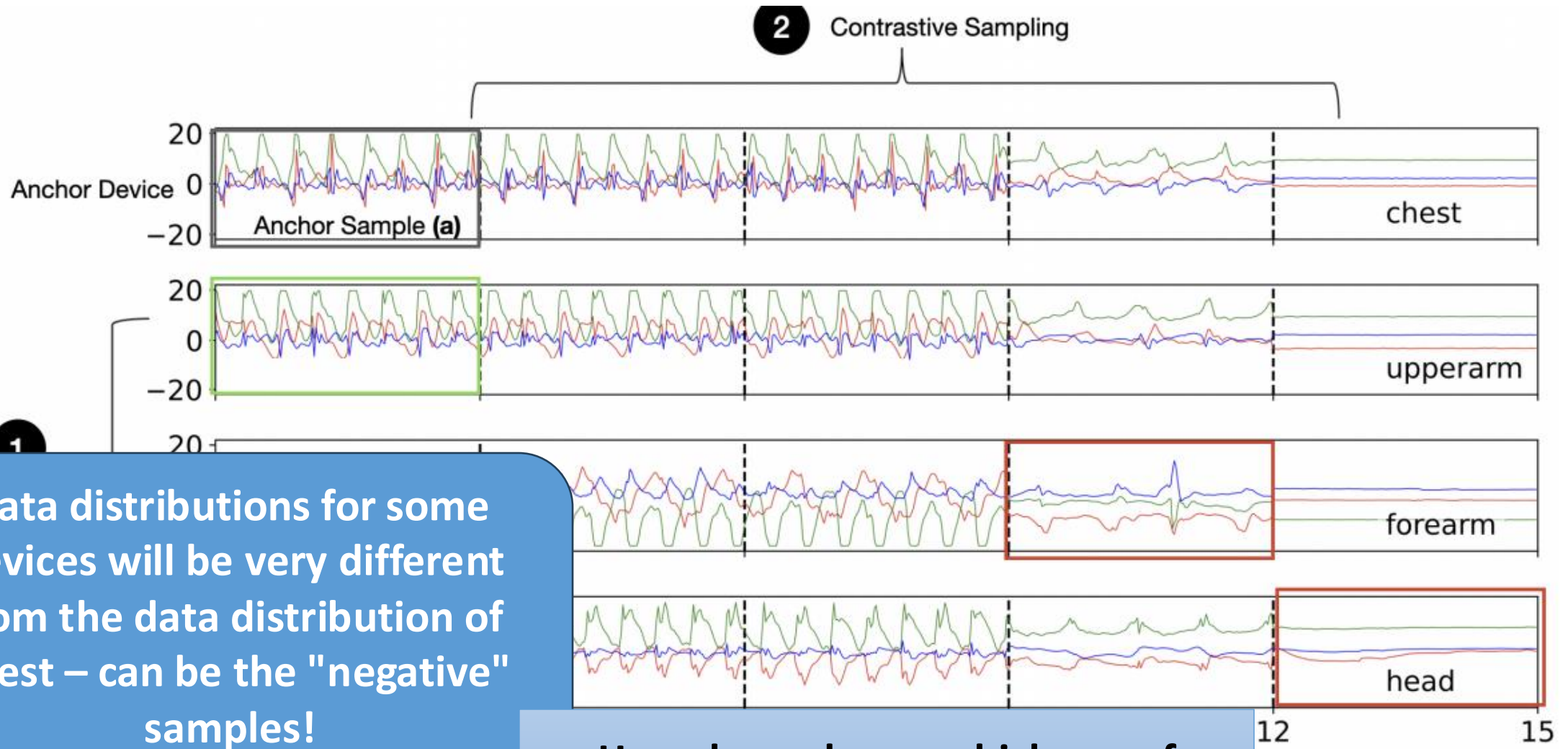
Body parts are interlinked
and legs when



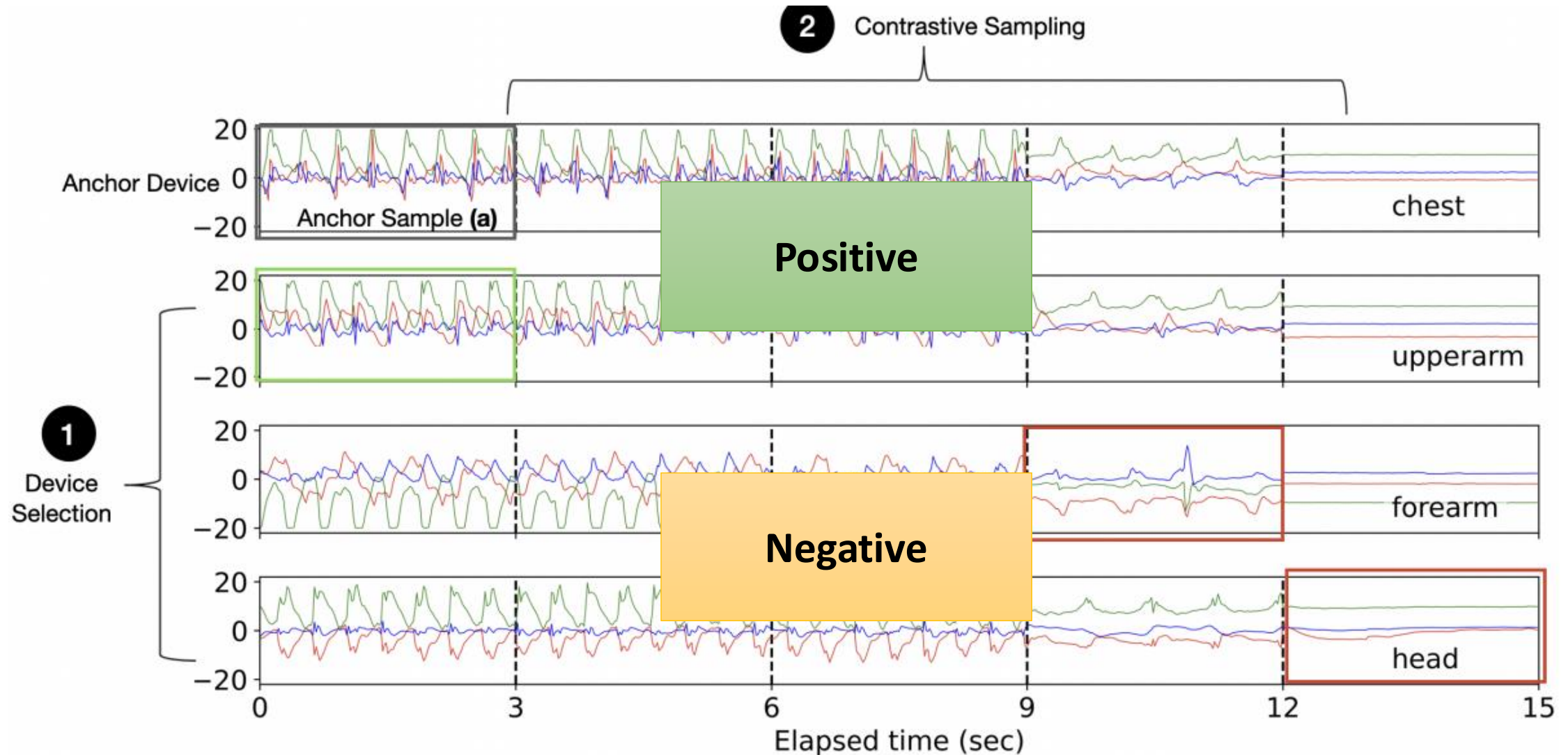
Challenge 1: Selecting Positive and Negative Samples



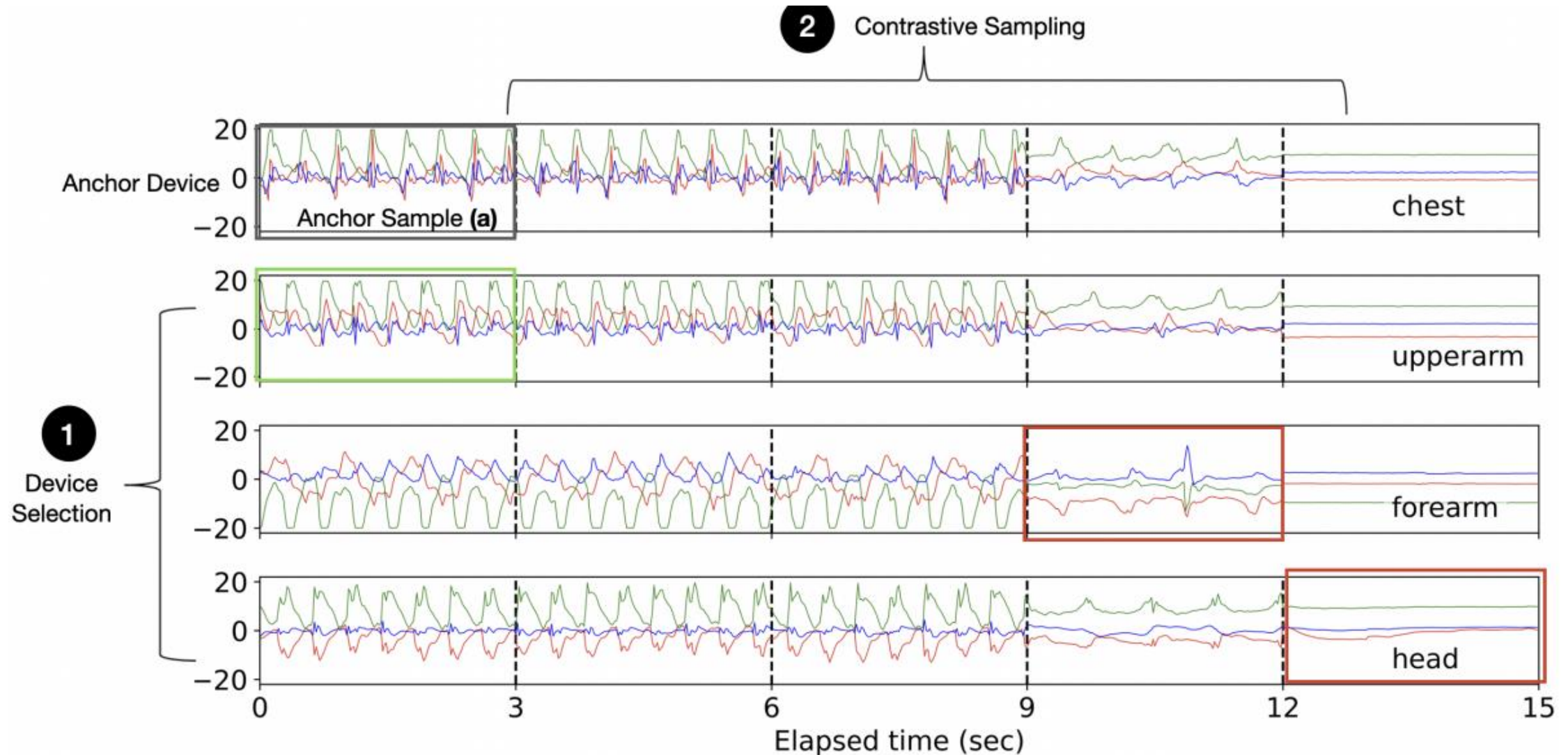
Challenge 1: Selecting Positive and Negative Samples



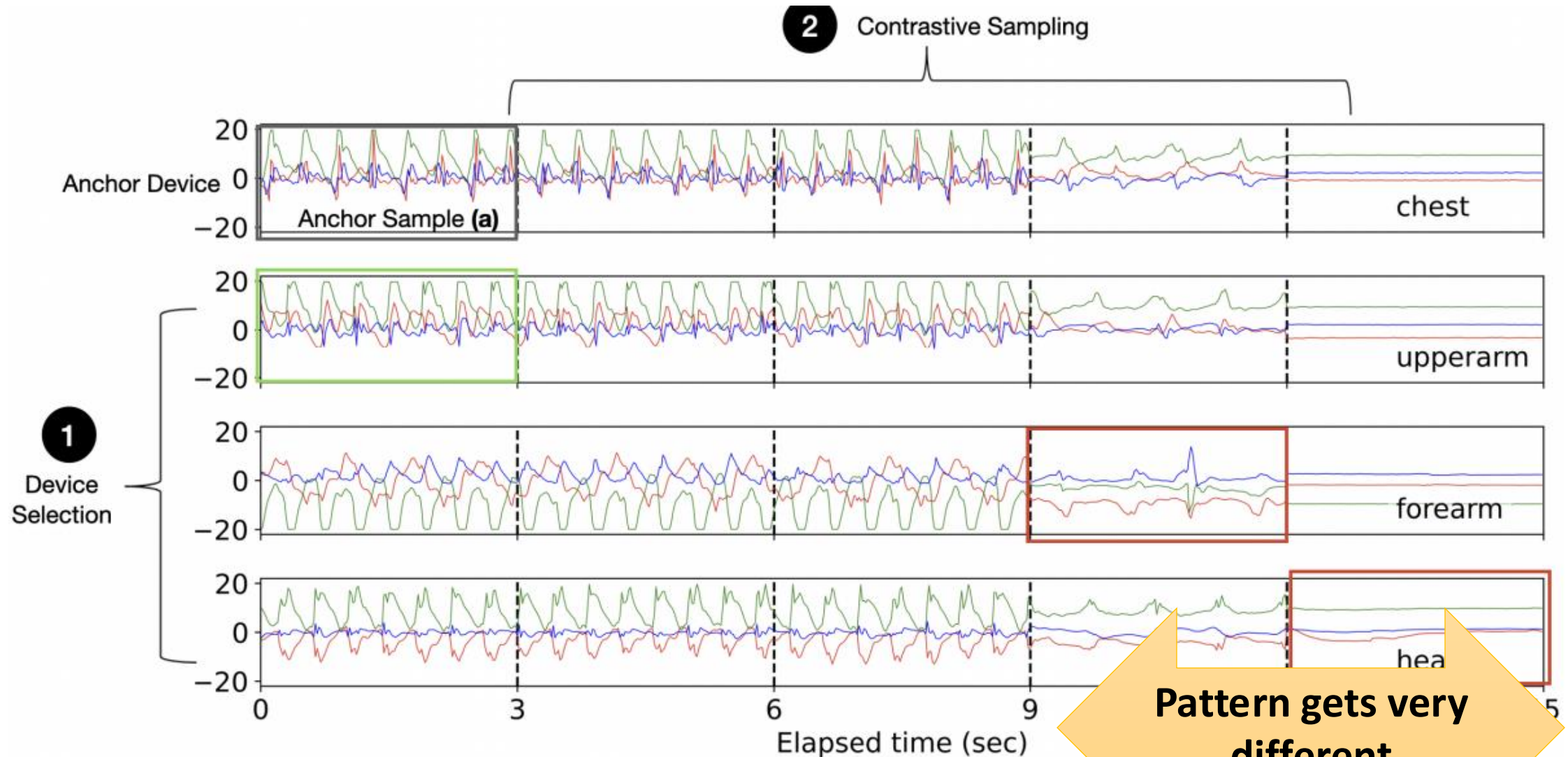
Challenge 1: Selecting Positive and Negative Samples



Challenge 2: Which Samples Can be Used in CL?



Challenge 2: Which Samples Can be Used in CL?



Collaborative Self-supervised Learning

- We have D devices with time-aligned but unlabeled sensor datasets $\{X_i\}_{i=1}^D$
- The dataset is pre-segmented in T number of windows
 - Each dataset X_i contains T windows $\{x_i^1, x_i^2, \dots, x_i^T\}$
 - Each sensor sample x_i^j is the 6-DoF IMU data
- Let $D^* \in D$ be an anchor device for which we want to train HAR prediction
- Let $L^* = \{(x_*^1, y_*^1), \dots, (x_*^m, y_*^m)\}$ be a small pre-segmented labeled dataset from the anchor device with m windows ($m \ll T$)
 - X_* are the sensor samples, y_* are the labels
- **Objectives:**
 - Use the unlabeled dataset $\{X_i\}_{i=1}^D$ to train a feature extractor $f(.)$
 - Use the pretrained $f(.)$ to obtain the feature embeddings for the labeled anchor samples and subsequently train an HAR classifier $g(.)$ that maps feature embeddings to the labels

Solution Steps

- Initialize the feature extractor $f(.)$ with random weights
- Sample a batch B of time-aligned unlabeled data $\{x_i^1, \dots, x_i^B\}_{i=1}^D$ from all the D devices
- **Device Selection:** Decide which of the remaining devices will provide *positive samples* and *negative samples* for CL
- **Contrastive Sampling:** For each anchor sample, which samples from the batches of positive and negative devices should be used for CL

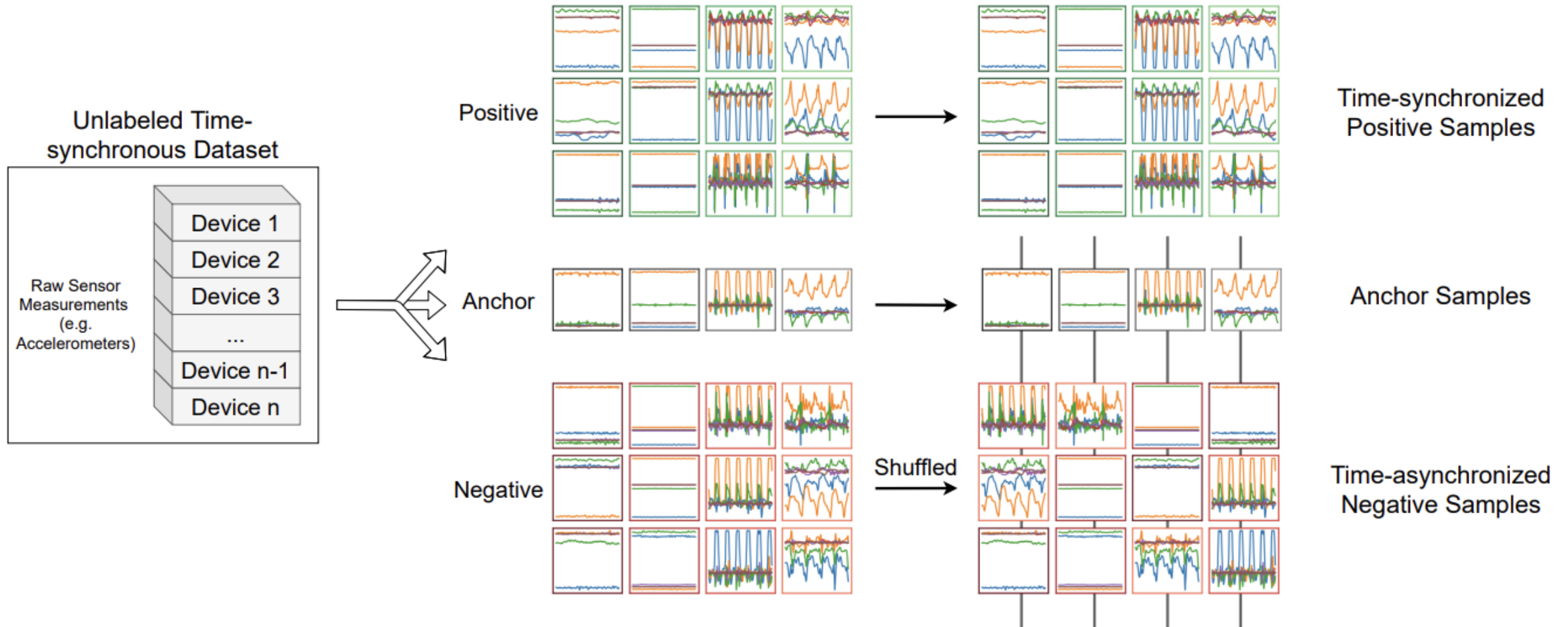
Solution Steps

1

Device Selection

2

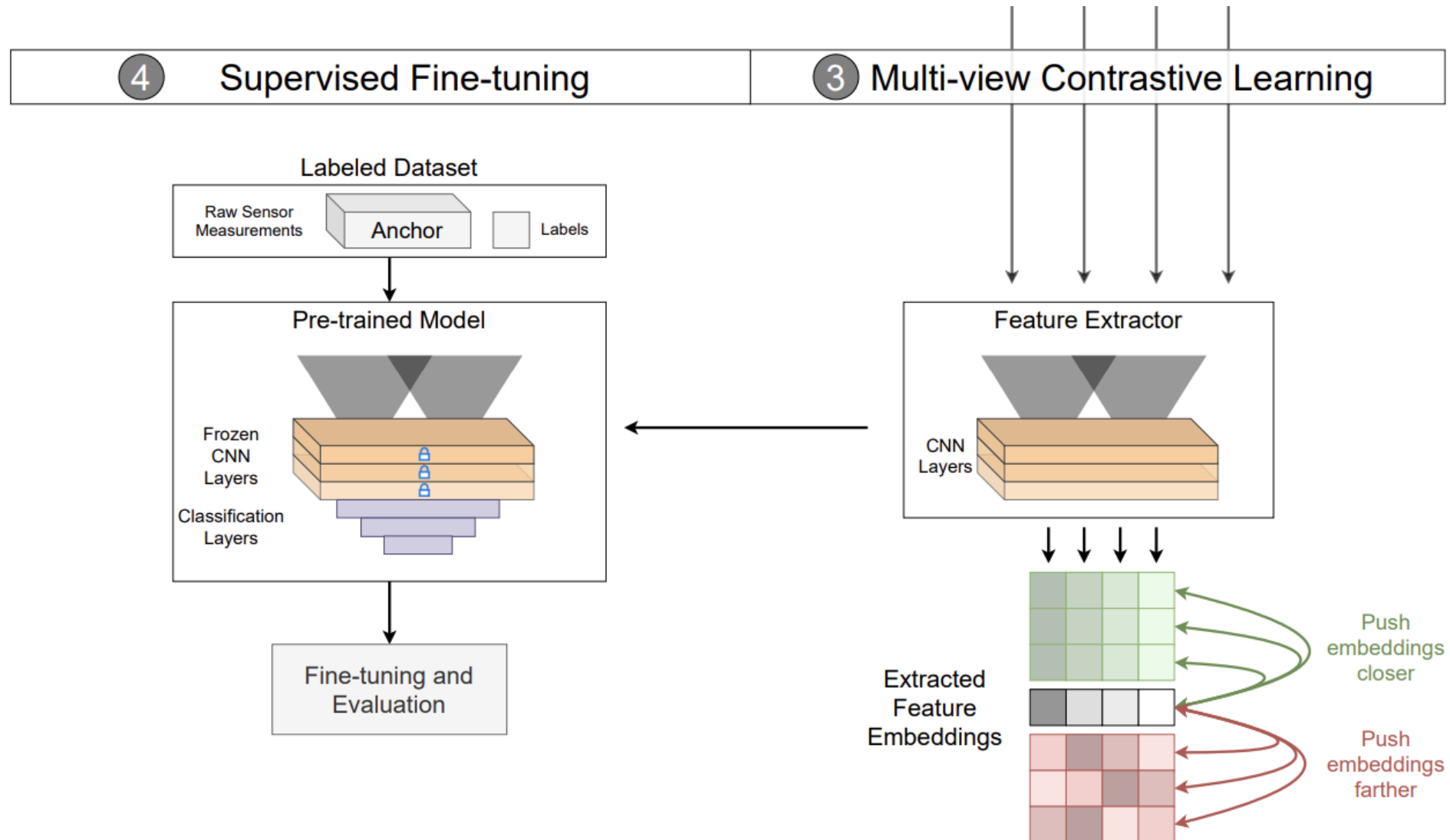
Contrastive Sampling



Solution Steps

- **Multi-view Contrastive Learning:** The anchor, positive and negative sample(s) are fed to the feature extractor $f(\cdot)$ to generate feature embeddings
 - Use 1D-CNN as the feature extractor
 - Compute *Multi-view Contrastive Loss* to push the positive samples towards the anchor and the negative embeddings far from the anchor
- Repeat the above steps until the contrastive loss converges (and we compute all the anchor samples for the batch)
- **Supervised Fine Tuning:** Train a HAR classifier using $f(\cdot)$ and the labeled dataset from the anchor device

Solution Steps

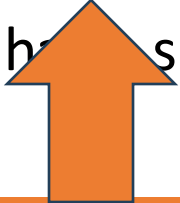


Device Selection

- 'Good' positive samples (x^+):
 - x^+ should belong to the same label/class as the anchor sample x
 - x^+ should come from a device whose data distribution has a small divergence from that of the anchor device.

Device Selection

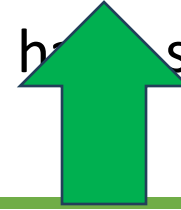
- 'Good' positive samples (x^+):
 - x^+ should belong to the same label/class as the anchor sample x
 - x^+ should come from a device whose data distribution has small divergence from that of the anchor device.



We do not have ground truth data, so how do we ensure this condition?

Device Selection

- 'Good' positive samples (x^+):
 - x^+ should belong to the same label/class as the anchor sample x
 - x^+ should come from a device whose data distribution has small divergence from that of the anchor device.



The Catch: Devices are time-aligned in TSMDS.
So, all the devices observe the same class
label at every time window

Device Selection

- 'Good' positive samples (x^+):
 - x^+ should belong to the same label/class as the anchor sample x
 - x^+ should come from a device whose data distribution has a small divergence from that of the anchor device.
- 'Good' negative samples (x^-)
 - x^- should be a true negative (belongs to a different class)
 - The most informative negative samples are those whose embeddings are initially near to the anchor embeddings, and f needs to push them far apart.
 - f gets a strong supervisory signal from the data and more useful gradients during training if the embeddings are initially near to the anchor
 - Otherwise (if the embeddings are already far apart), f will receive a weaker supervisor signal: May affect its convergence

Device Selection

- 'Good' positive samples (x^+):
 - x^+ should belong to the same class as the anchor sample x
 - x^+ should come from a device with a small divergence from that of the anchor device.
- 'Good' negative samples (x^-)
 - x^- should be a true negative (belongs to a different class)
 - The most informative negative samples are those whose embeddings are initially near to the anchor embeddings, and f needs to push them far apart.
 - f gets a strong supervisory signal from the data and more useful gradients during training if the embeddings are initially near to the anchor
 - Otherwise (if the embeddings are already far apart), f will receive a weaker supervisor signal: May affect its convergence

**Strictly enforcing this
may not be possible**



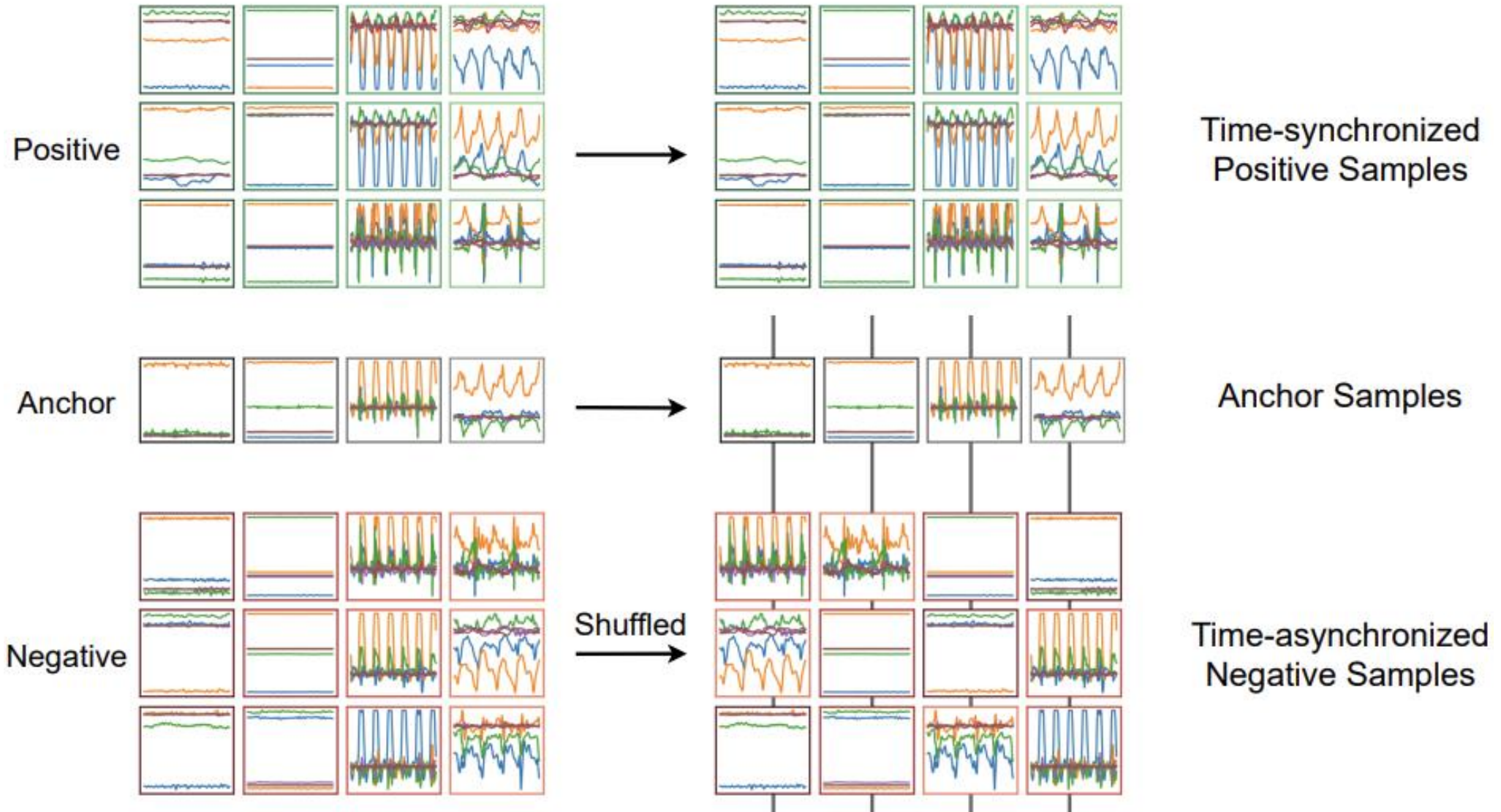
Device Selection

- Increase the likelihood of selecting 'good' positive and negative samples
 - Compute pairwise Maximum Mean Discrepancy (MMD) between the data samples from the anchor and other devices
 - **MMD**: Distance between the feature means -> Higher MMD implies a larger distance between the two distributions
- Device Selection Policy:
 - **Closest Positive**: Least MMD distance from the anchor is chosen as the positive (empirically observed that one positive device gives the best performance)
 - **Weighted Negatives**: All from the candidate set, but the feature contributions are weighted by the inverse of the MMD from anchor (satisfies the second requirement – negatives closer to the anchor gets more weight, also ensures the first requirement by enabling other devices to supercede the training when one negative device belongs to the same class as of the anchor)

Contrastive Sampling

- Decides which data samples should be picked from each device for contrastive training
- Sampling policy
 - Synchronous positive samples
 - Asynchronous negative samples
- **Synchronous positive samples:** Time-aligned positive counterparts from the positive device, corresponding to the anchors
- **Asynchronous negative samples:** We need the samples from a different class – samples which are not time-synchronized with the positive class are "*more likely*" to be the samples from a different class

Contrastive Sampling



Multi-view Contrastive Loss

- Extension of standard contrastive loss for multiple positive and negative samples:

$$\mathcal{L}_{MCL} = -\log \frac{\sum_{i=0}^{|D^+|} \exp(\text{sim}(z^*, z_i^+) / \tau)}{\left(\sum_{i=0}^{|D^+|} \exp(\text{sim}(z^*, z_i^+) / \tau) + \sum_{j=0}^{|D^-|} w_j \exp(\text{sim}(z^*, z_j^-) / \tau) \right)}$$

- $\text{sim}(\cdot)$ indicates cosine similarity
- The loss function is minimized for each batch of data using stochastic gradient descent
 - Increases the cosine similarity between the anchor and positive samples (push them closer in the feature space)

ColloSSL Performance (Macro F1 Score)

Dataset (anchor)	Supervised-single	Random	Supervised-multi	AutoEncoder-single	AutoEncoder-multi	Multi-task SSL	ColloSSL
RealWorld							
forearm	0.732 (100%)	0.253 (50%)*	0.495 (100%)*	0.723 (100%)*	0.739 (75%)	0.734 (50%)	0.767 (25%)
head	0.643 (100%)	0.211 (25%)*	0.537 (100%)*	0.647 (100%)	0.646 (25%)	0.670 (25%)	0.690 (10%)
shin	0.781 (100%)	0.375 (100%)*	0.628 (100%)*	0.784 (10%)	0.765 (75%)*	0.81 (10%)	0.81 (10%)
chest	0.715 (100%)	0.228 (50%)*	0.650 (100%)*	0.478 (75%)*	0.720 (25%)	0.722 (10%)	0.716 (25%)
thigh	0.701 (100%)	0.283 (100%)*	0.586 (100%)*	0.695 (75%)*	0.656 (25%)*	0.675 (75%)*	0.690 (25%)*
upper arm	0.726 (100%)	0.268 (75%)*	0.595 (100%)*	0.739 (75%)	0.708 (100%)*	0.753 (10%)	0.740 (25%)
waist	0.745 (100%)	0.297 (25%)*	0.674 (100%)*	0.582 (75%)*	0.770 (10%)	0.778 (10%)	0.781 (10%)
Opportunity							
back	0.439 (100%)	0.164 (50%)*	0.253 (25%)*	0.446 (10%)	0.445 (50%)	0.380 (25%)*	0.556 (10%)
lla	0.370 (100%)	0.197 (100%)*	0.398 (25%)	0.386 (25%)	0.375 (25%)	0.374 (100%)	0.516 (10%)
left shoe	0.391 (100%)	0.164 (10%)*	0.396 (75%)	0.282 (100%)*	0.172 (25%)*	0.164 (100%)*	0.416 (25%)
right shoe	0.378 (100%)	0.164 (10%)*	0.392 (25%)	0.265 (100%)*	0.166 (50%)*	0.183 (100%)*	0.402 (10%)
rua	0.416 (100%)	0.164 (10%)*	0.293 (100%)*	0.447 (10%)	0.375 (10%)*	0.277 (10%)*	0.538 (10%)
PAMAP2 - Locomotion							
ankle	0.731 (100%)	0.609 (50%)*	0.589 (10%)*	0.651 (10%)*	0.770 (10%)	0.774 (50%)	0.784 (100%)
chest	0.654 (100%)	0.295 (50%)*	0.738 (10%)	0.669 (75%)	0.655 (10%)	0.730 (10%)	0.741 (10%)
hand	0.723 (100%)	0.496 (25%)*	0.731 (25%)	0.723 (100%)	0.750 (10%)	0.791 (10%)	0.740 (10%)
PAMAP2 - ADL							
ankle	0.550 (100%)	0.262 (100%)*	0.548 (100%)*	0.56 (25%)	0.489 (100%)*	0.567 (50%)	0.578 (25%)
chest	0.640 (100%)	0.156 (100%)*	0.64 (50%)	0.623 (25%)	0.607 (75%)*	0.615 (100%)*	0.651 (50%)
hand	0.575 (100%)	0.208 (50%)*	0.585 (25%)	0.577 (75%)	0.586 (50%)	0.596 (50%)	0.617 (25%)

The number in the bracket indicates amount of data used

Take Aways

- IMU can provide significant insights about human activities
 - Motion models can well capture various gestures in different body parts
 - One of the fundamental sensing modality used in almost every smart device
- Statistical estimators/filters help denoising IMU data and predicting gestures
- Obtained labeled data is costly for ubicomp applications
 - Supervised methods are good if sufficient labels are available (but they are scarce)
 - Unsupervised or semi-supervised/self-supervised methods may give reasonable performance with significantly less amount of data



Happy Learning!

Some resources
related to this topic

