RESOURCE-CONSTRAINED MACHINE LEARNING FOR UBIQUITOUS COMPUTING APPLICATIONS

Gautham Krishna Gudur

gauthamkrishna.gudur@gmail.com

@gauthamkrishna_

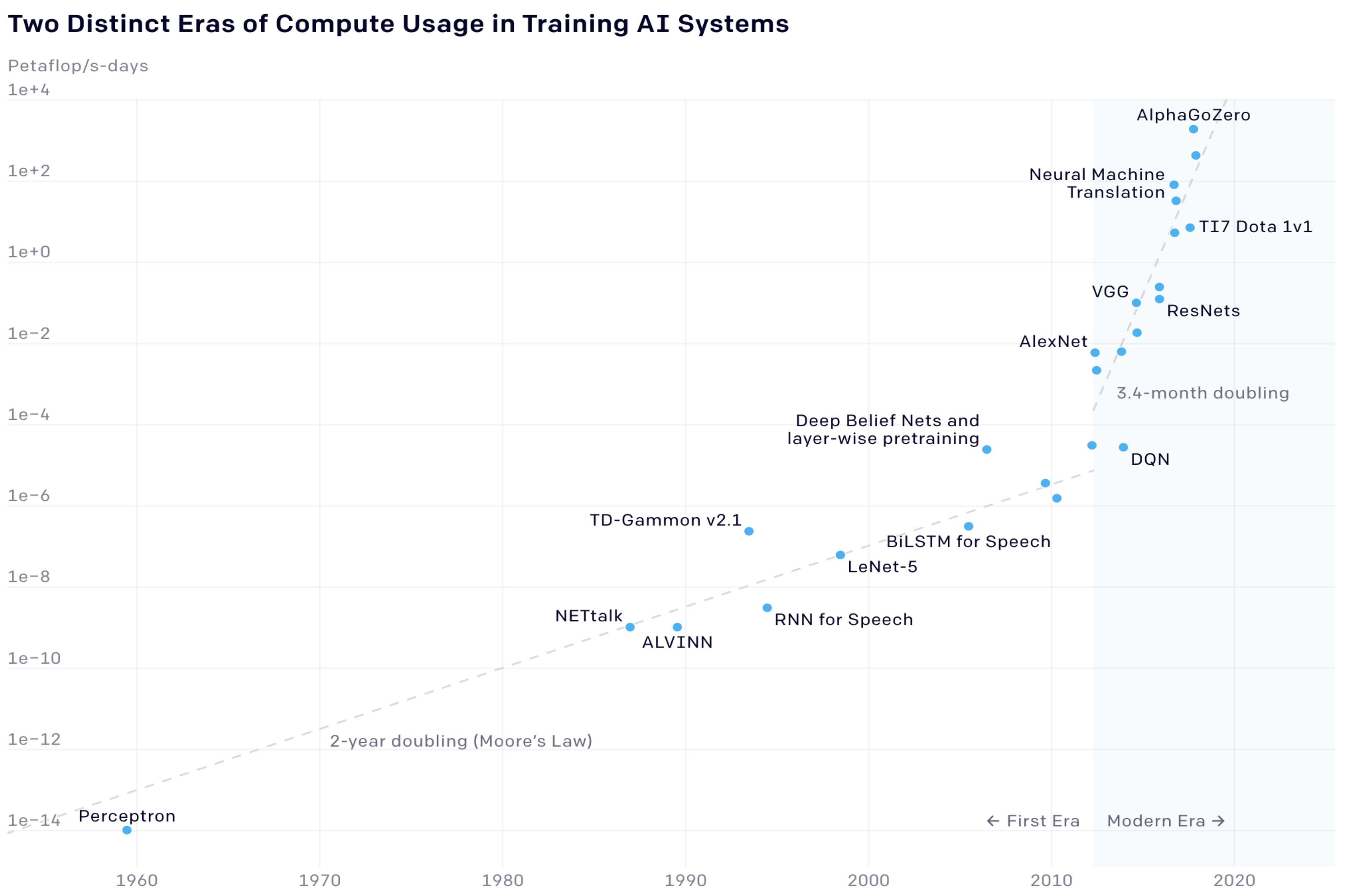


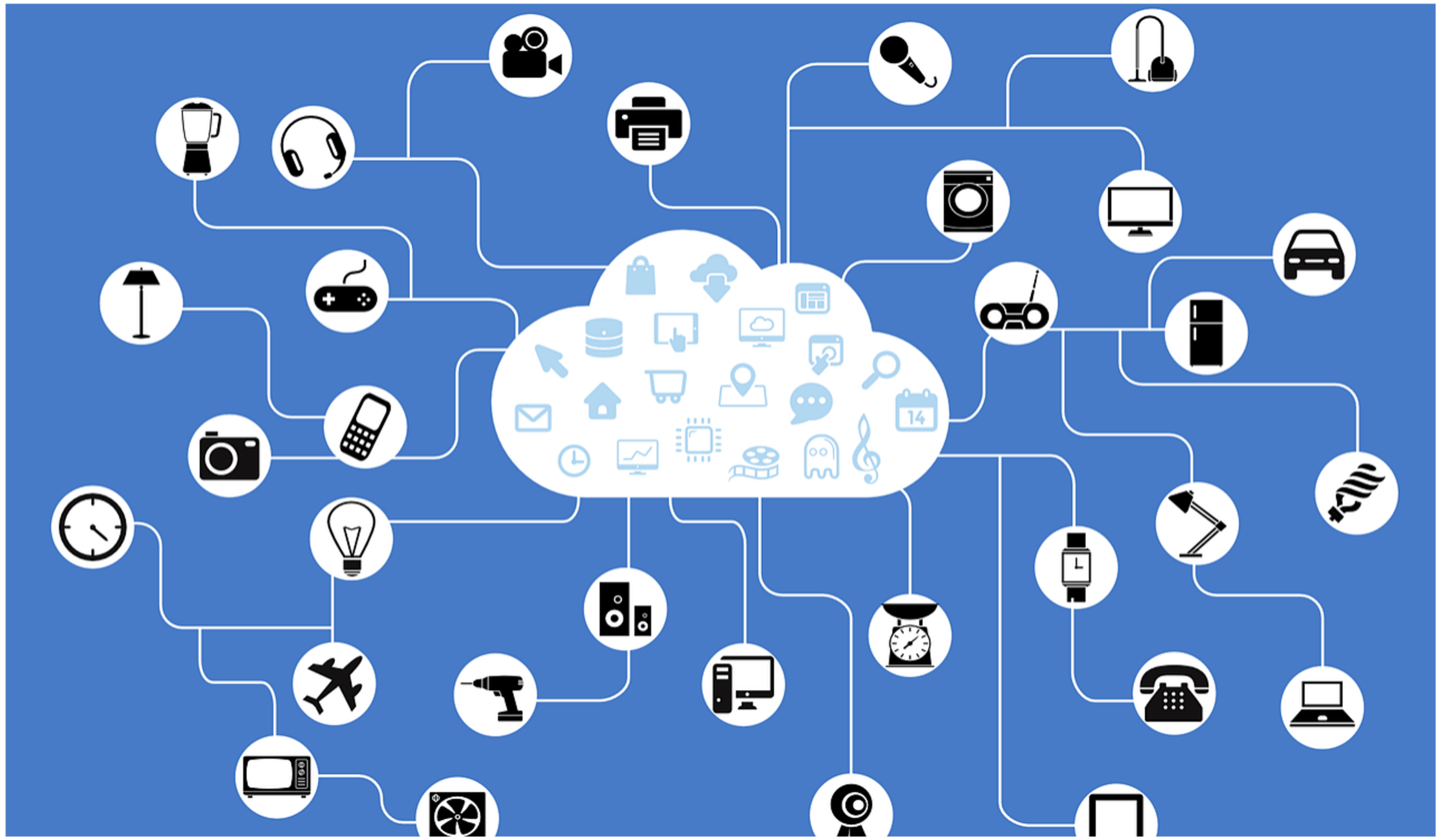
Fig. taken from OpenAI blog

UBIQUITOUS COMPUTING (UbiComp)

Computing that appears anytime and everywhere.

Also widely called **Pervasive Computing**, **Ambient Intelligence**, etc.

Encompasses **Mobile/Wearable Computing**, **Context-Aware Computing** and the likes



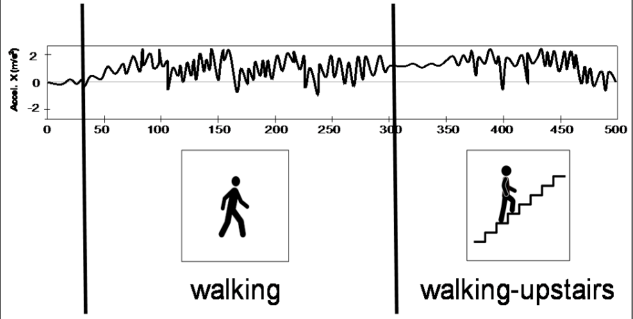
WEARABLE/ MOBI-QUITOUS COMPUTING

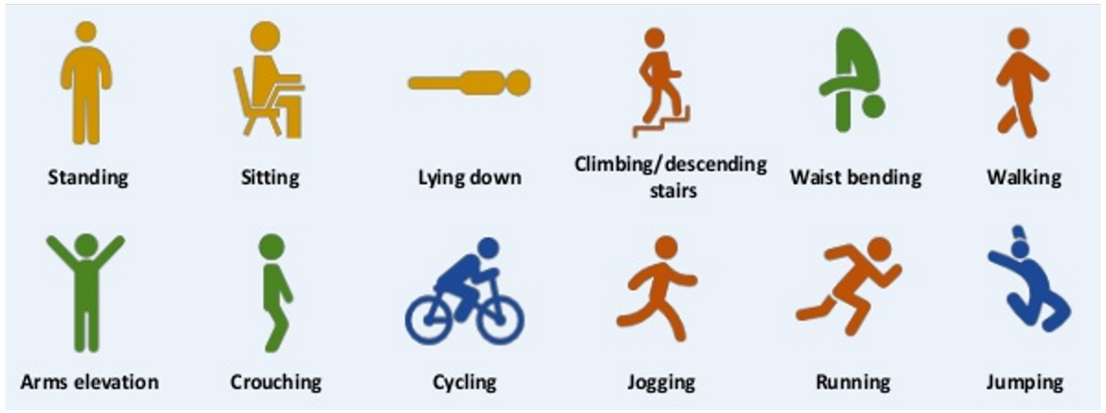
- Expansive growth of usage of mobile phones, smartwatches across various users.
- Significant research in the field of ubiquitous & mobile/wearable computing.
- Data from sensors embedded in mobiles/wearables conveniently provide a way to extract contextual, behavioral information of users.

Applications particularly gaining importance in such fields are:

- **Pervasive Healthcare**
- **Physical Activity Monitoring**
- **Fitness Tracking**
- **Emotion Recognition**
- **Gesture Recognition**

and more …





RESOURCE

Resource is a very broad term

- **Hardware** – Constrained edge devices like RPi, Arduino, FPGA
- **Communication Overhead** – Privacy Preserving ML, Federated Learning, Distribute Learning
- **Data** – Limited Data (Few Shot Learning), Limited Labels
- **Model** – Classic ML or squeezed DL models
- **Feature Extractor** – Automatic or hand-picked
- **Memory Overhead** – Compression techniques
- **Inference Engine** – Quantization, Pruning
- **Retraining** – Active Learning, Online Learning
- **Semi-Supervised, Unsupervised Learning**

  and many more …

Heterogeneities exist in everything!

IT IS VERY IMPORTANT TO BE RESOURCE-AWARE!!

AGENDA

- Quick walk-through of different applications/scenarios for marrying machine learning with resource-aware ubiquitous computing.
- Address challenges of resources at all levels that come up.
- To each problem, its own challenge. Hard to generalize to universal solutions apply across all scenarios and applications.

Few UbiComp/ML projects:

- Dynamic Gesture Recognition System using Machine Learning
- Incremental Learning on Constrained Devices for Human Activity Recognition
- On-Device Bayesian Active Learning
- On-Device Vision-Based Bus Stop Recognition System
- Resource-Constrained Federated Learning for Heterogeneous Labels and Models

Dynamic Gesture Recognition using Machine Learning

Gautham Krishna et al., A Generic Multi-Modal Dynamic Gesture Recognition System using Machine Learning, IEEE FICC '18

GESTURE RECOGNITION APPLICATIONS

- Developing aids for the hearing-impaired using Sign Language interpretation
- Virtual Gaming
- Smart Home Environments
- Socially-Assistive Robotics
- Affective Computing

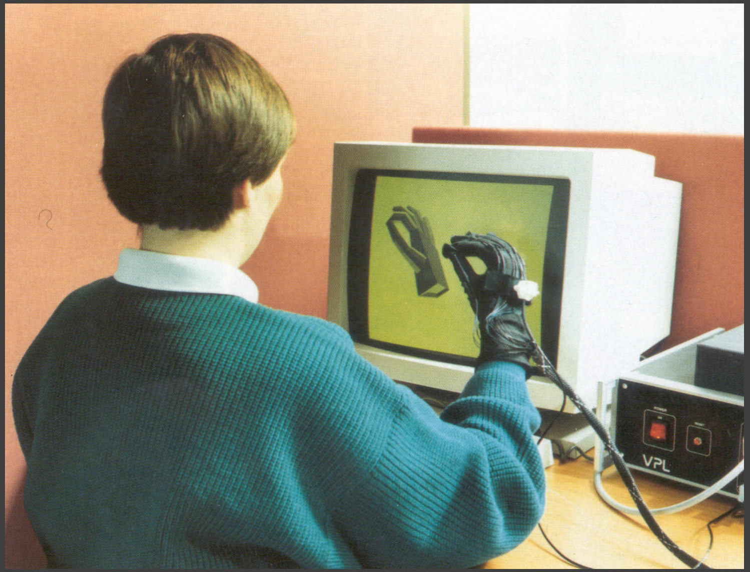
GESTURE RECOGNITION APPROACHES

Vision Based Approach

- Camera for tracking movements
- Higher Computational Overheads

Sensor (Haptic) Based Approach

- Cost-effective and Computationally Efficient
- Dependent on Environmental Stimuli
- Examples: Accelerometer, Gyroscope, Magnetometer



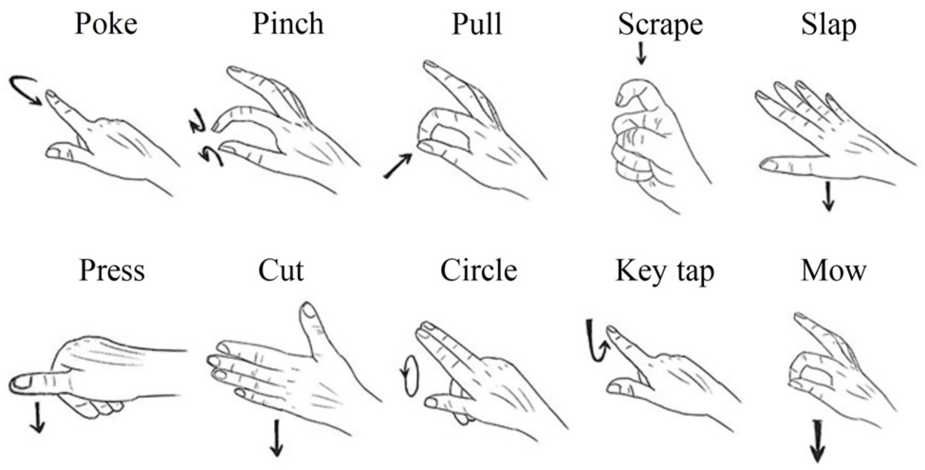
GESTURE RECOGNITION TYPES

Static Gesture Recognition

- Uniquely characterized to static Start and End points
- Analogous to an Image

Dynamic Gesture Recognition

- Requires the entire sequence of Gesture sample
- Analogous to Video



CHALLENGES IN CONVENTIONAL GESTURE RECOGNITION SYSTEMS

- Requires instrumented and multifarious sensors
- Failure to generalize across multiple users (modally inflexible)
- Gesture specific feature extraction
- Failure to handle disparate speeds of same gestures signed by various users
- Application specificity and deployment in Low-cost platforms

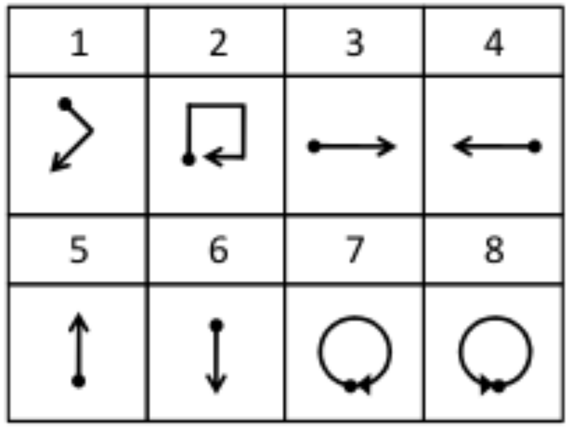
GOALS OF OUR PROPOSED SYSTEM

- Utilize only accelerometer-based datasets (containing g-values) – Sensor Minimization
- Handle disparate speeds across gestures by leveraging unique features
- Incorporate multi-modality across users (**plug and play**)
- Provide flexibility to emphasize between classification accuracy and classification time
- Deploy on a low-cost platform

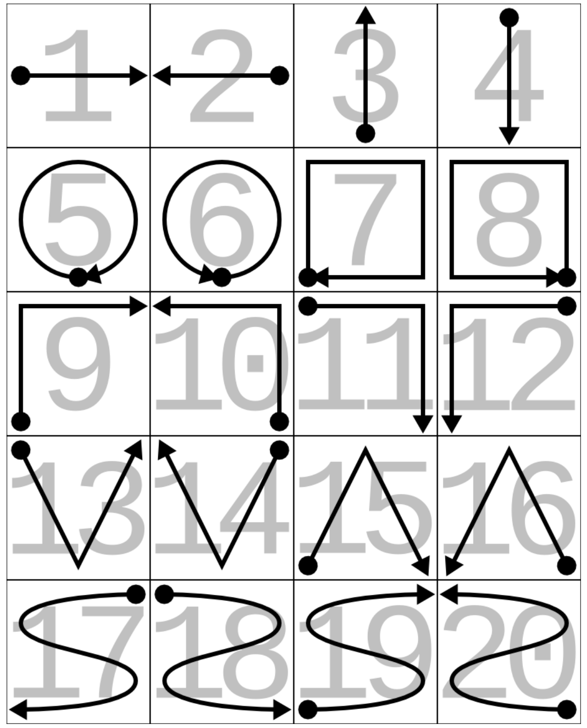
GESTURE DATASETS

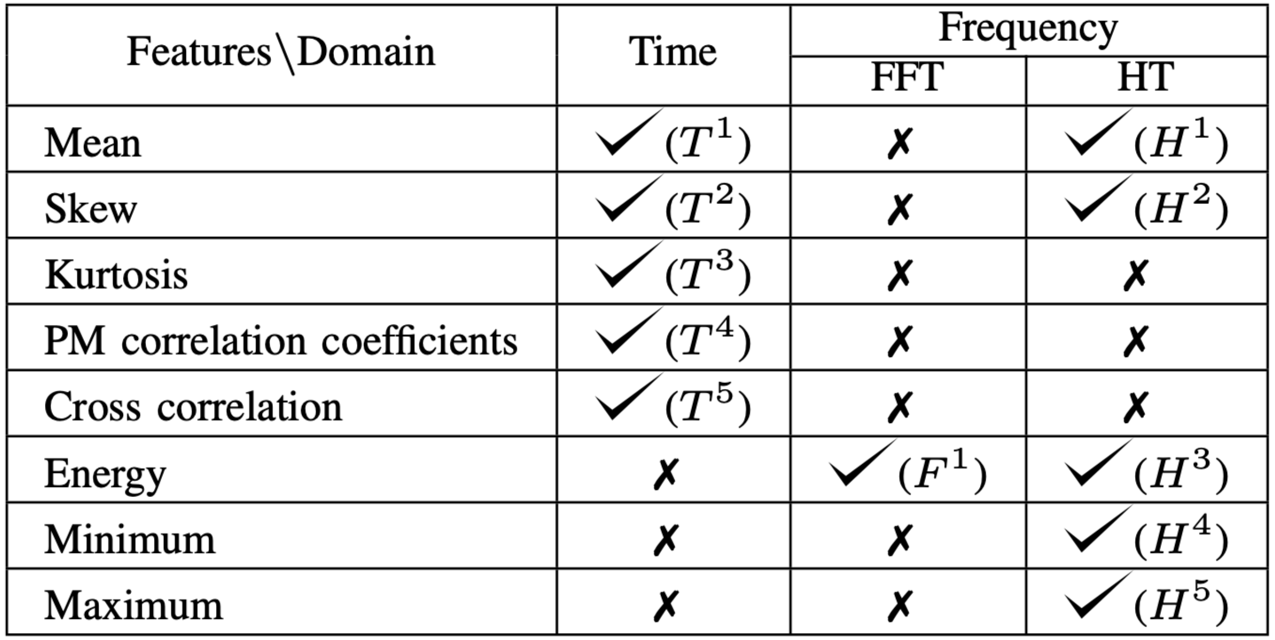
Dataset	Users (U)	Gestures (N_G)	Samples per Gesture (S_G)	Days (N_D)	N_{GS}
uWave (D_u)	8	8	10	7	4480
Sony (D_S)	8	20	20	-	3200

uWave Gestures



Sony Gestures





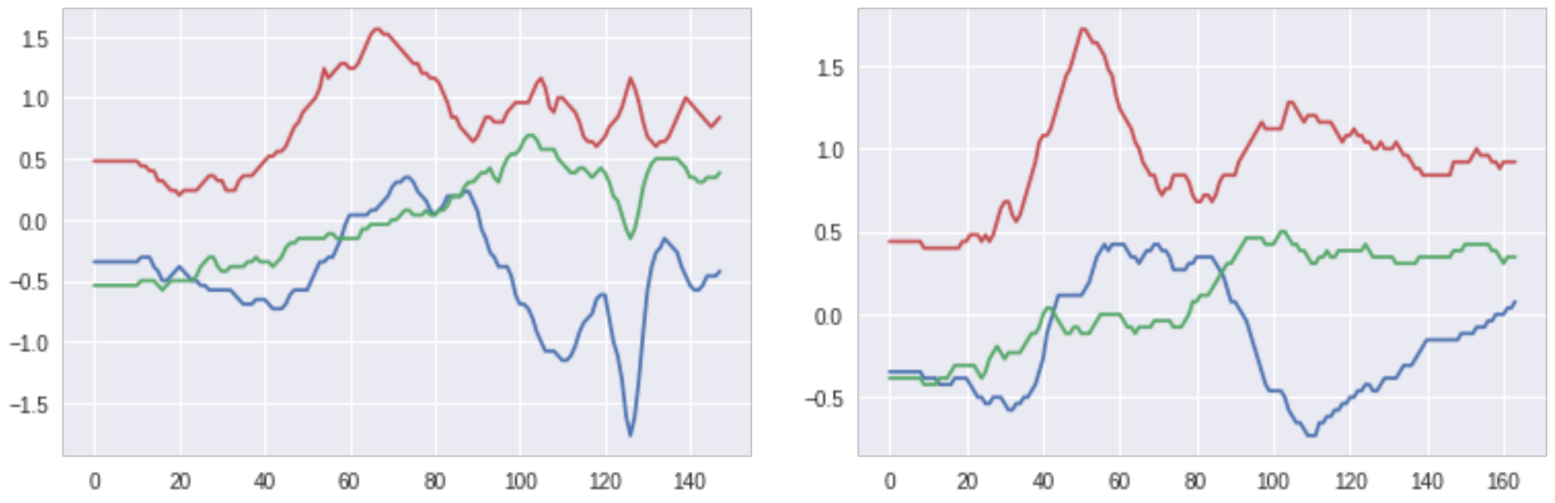
Features\Domain	Time	Frequency	
		FFT	HT
Mean	✓ (T^1)	✗	✓ (H^1)
Skew	✓ (T^2)	✗	✓ (H^2)
Kurtosis	✓ (T^3)	✗	✗
PM correlation coefficients	✓ (T^4)	✗	✗
Cross correlation	✓ (T^5)	✗	✗
Energy	✗	✓ (F^1)	✓ (H^3)
Minimum	✗	✗	✓ (H^4)
Maximum	✗	✗	✓ (H^5)

- We characterize a set of unique features that accurately represent a gesture signed by any user.
- Two transforms: ***Fast Fourier Transform (FFT)*** and ***Hilbert Transform (HT)***, are utilized to convert from time domain to frequency domain.
- Final set of features arrived by recursive feature elimination across all domains and transforms.

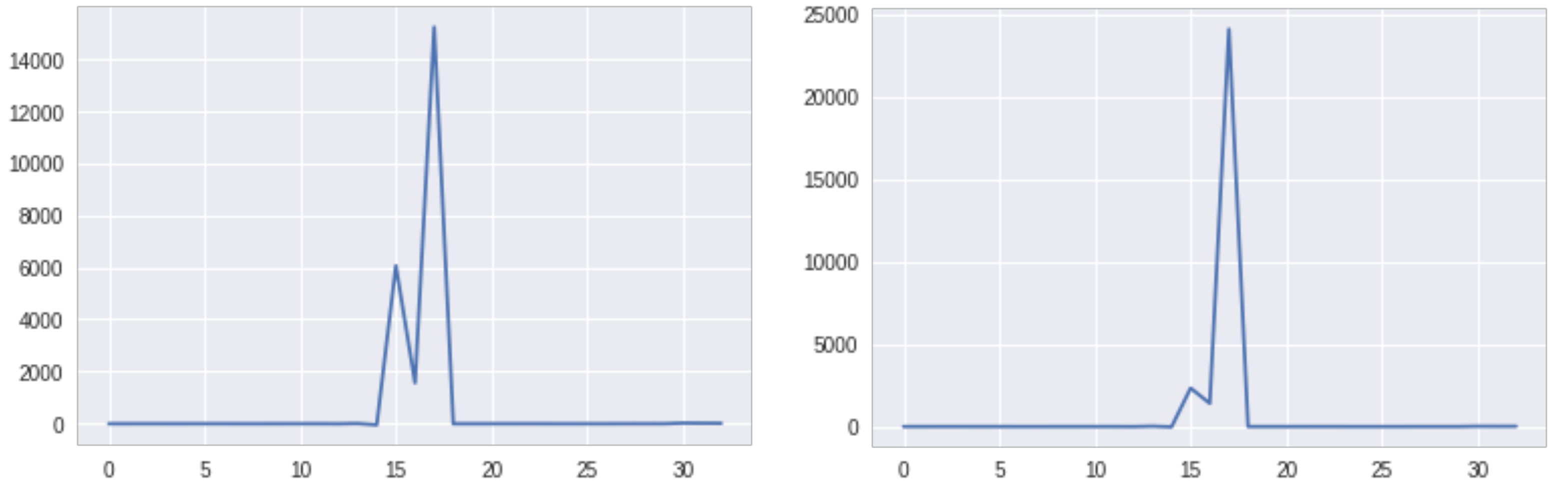
FEATURE EXTRACTION

FEATURE EXTRACTION

INPUT



FEATURE VECTOR



EXPERIMENT

- ML classifiers were chosen over traditional algorithms like Dynamic Time Warping (DTW), as generalizing a look-up table (template) for each gesture is computationally inefficient and hard to establish in the user-independent paradigm.
- The *ML* classification algorithms used here are,
  - Extremely Randomized Trees (Extra Trees)
  - Random Forest
  - Gradient Boosting
  - Bagging
  - Decision Trees
  - Naive Bayes
  - Ridge Classifier

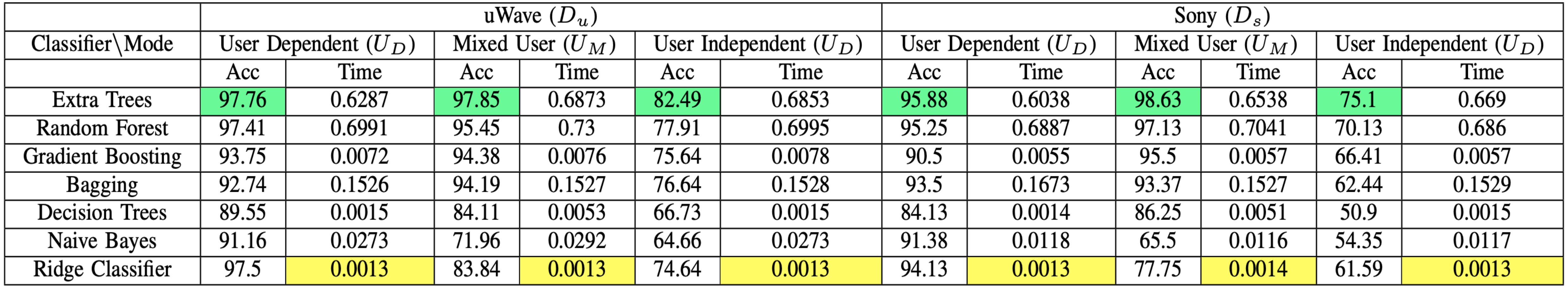
END-USER MODELING

- The end-user is provided the choice of any one of three proposed modes of operation:
  - ***User Dependent (UD):*** Estimator of how well the system performs when the train-test split is between the gestures of a single user.
  - ***Mixed User (UM):*** Representative of the complete set of gestures across all participants.
  - ***User Independent (UI):*** Employs a stratified k-fold cross validation technique which corresponds to training on several users and testing on the rest.

EXPERIMENT

- The seven classifiers are simulated on a single board computing platform – **Raspberry Pi Zero**.
- Raspberry Pi Zero priced at 5$, makes it a low-cost alternative to conventional computing modules.

RESULTS



	uWave (D_u)						Sony (D_s)					
Classifier\Mode	User Dependent (U_D)		Mixed User (U_M)		User Independent (U_D)		User Dependent (U_D)		Mixed User (U_M)		User Independent (U_D)	
	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time
Extra Trees	97.76	0.6287	97.85	0.6873	82.49	0.6853	95.88	0.6038	98.63	0.6538	75.1	0.669
Random Forest	97.41	0.6991	95.45	0.73	77.91	0.6995	95.25	0.6887	97.13	0.7041	70.13	0.686
Gradient Boosting	93.75	0.0072	94.38	0.0076	75.64	0.0078	90.5	0.0055	95.5	0.0057	66.41	0.0057
Bagging	92.74	0.1526	94.19	0.1527	76.64	0.1528	93.5	0.1673	93.37	0.1527	62.44	0.1529
Decision Trees	89.55	0.0015	84.11	0.0053	66.73	0.0015	84.13	0.0014	86.25	0.0051	50.9	0.0015
Naive Bayes	91.16	0.0273	71.96	0.0292	64.66	0.0273	91.38	0.0118	65.5	0.0116	54.35	0.0117
Ridge Classifier	97.5	0.0013	83.84	0.0013	74.64	0.0013	94.13	0.0013	77.75	0.0014	61.59	0.0013

Accuracy (in %) and Time (in seconds)

HARNet: Towards On-Device Incremental Learning using Deep Ensembles on Constrained Devices

Prahalathan et al., HARNet: Towards On-Device Incremental Learning using Deep Ensembles on Constrained Devices, EMDL ’18@ACM MobiSys ‘18

DEEP LEARNING FOR HUMAN ACTIVITY RECOGNITION

- ALLEVIATES THE PROBLEM OF CRAFTING SHALLOW HAND-PICKED FEATURES
- AUTOMATICALLY EXTRACTS DISCRIMINATIVE FEATURES
- DOES NOT REQUIRE EXTENSIVE DOMAIN KNOWLEDGE
- ENHANCES SCALABILITY AND GENERALIZABILITY

PROMINENT CHALLENGES IN ON-DEVICE HAR

1. On-Device Incremental Learning

- Model updation incrementally
- Facilitation of User Adaptability
- Complex deep architectures generally have high computational overheads, hence difficult to update models on-device

PROMINENT CHALLENGES IN ON-DEVICE HAR

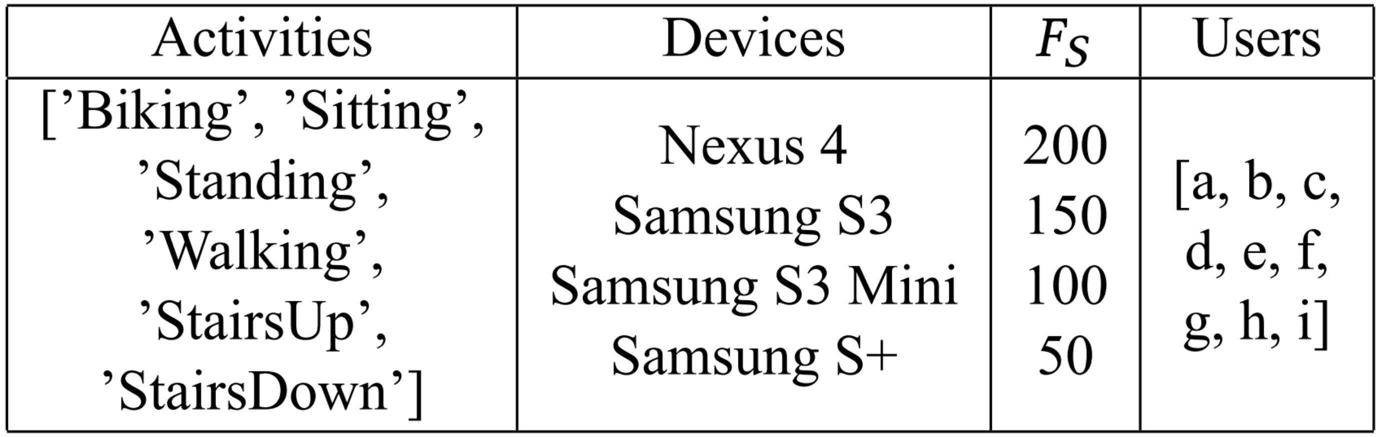
2. Heterogeneity

- Sampling rates, their instability due to different OS types, CPU load conditions and varied user characteristics among others
- Performance across various users and mobile phones in real-world is generally sub-optimal due to the aforementioned factors

GOALS OF OUR PROPOSED SYSTEM

- Develop a generic HAR model in heterogeneous conditions which supports **Incremental Learning**, and **resource-friendly**
- Systematic minimization of resources
- Effective training and deployment on a Mobile/Embedded platform, whilst achieving on-par accuracies compared to state-of-the art models
- Facilitate **User Adaptability**

HHAR DATASET



Activities	Devices	F_S	Users
['Biking', 'Sitting', 'Standing', 'Walking', 'StairsUp', 'StairsDown']	Nexus 4 Samsung S3 Samsung S3 Mini Samsung S+	200 150 100 50	[a, b, c, d, e, f, g, h, i]

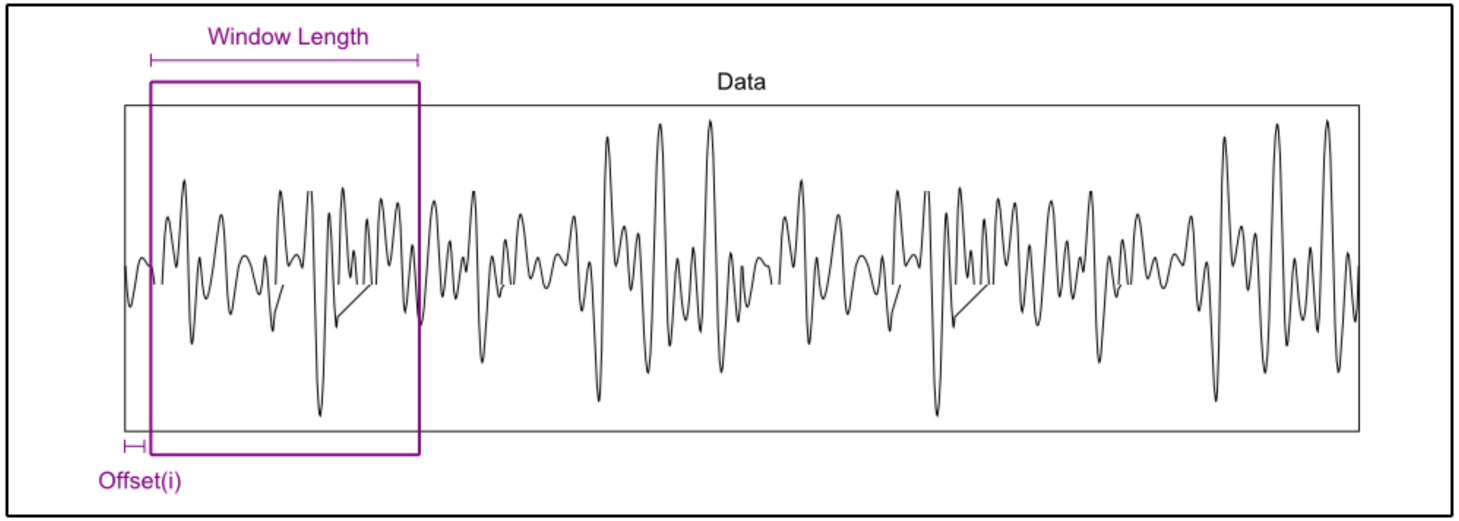
Allan et al., Smart Devices are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition, SenSys '15

PRE-PROCESSING

- To handle varying sampling rates
- Obtain a rich representation of the signal components
- **Windowing & Decimation**
- **Discrete Wavelet Transform (DWT)**

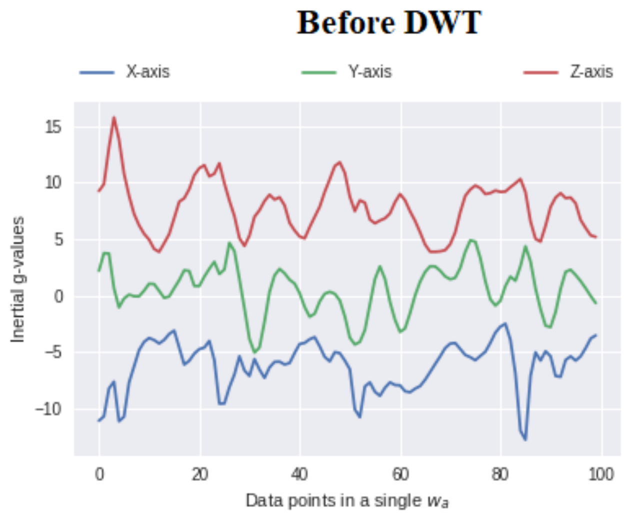
WINDOWING & DECIMATION

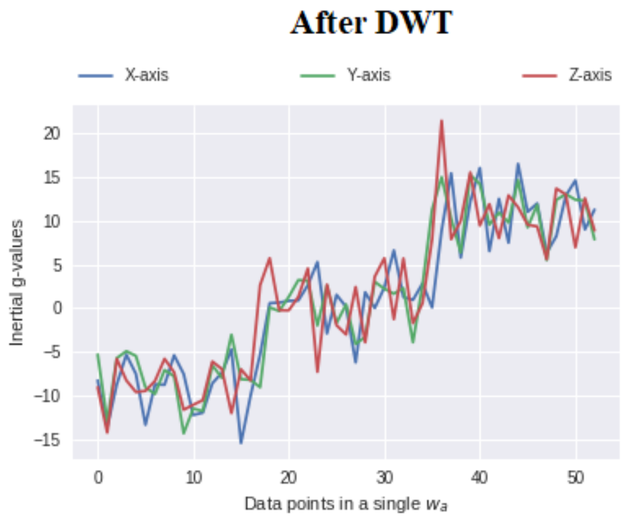
- Split raw inertial data into non-overlapping two-second activity windows
- Result in disparate length windows due to varying sampling rates across phones
- Hence, **Decimation** – a Down-sampling technique is performed (to the lowest frequency) to ensure uniformity in window lengths
- A maximum of ~75% data reduction is observed for smartphones with the highest sampling frequency



DISCRETE WAVELET TRANSFORM (DWT)

- Better representation of raw inertial signals
- Captures well-defined temporal characteristics in frequency domain
- Approximation co-effs (low frequency components) only used, discarded Detail co-effs – obtaining a smoothened version
- Results in compression of data up to ~50%





MODEL

Intra-Axial Dependencies:

- ***Conv-1D:*** Kernel extracts characteristics from each axis individually. Two-layer stacked network (8 and 16 filters each) with 2x2 receptive field size, with BatchNorm and a 2x2 Max-pool layer.
- ***LSTM:*** Capturing temporal information in time-series data. Two-layer stacked LSTM network with 32 and 20 output cells each with a Hyperbolic Tangent (tanh) activation function.
- ***LSTM -> Conv-1D:*** Combining both local characteristics and temporal information. A single LSTM layer with 32 output cells followed by a convolutional Conv-1D layer of 8 filters with kernel size of 2 with BatchNorm and a 2x2 Max-pool layer.

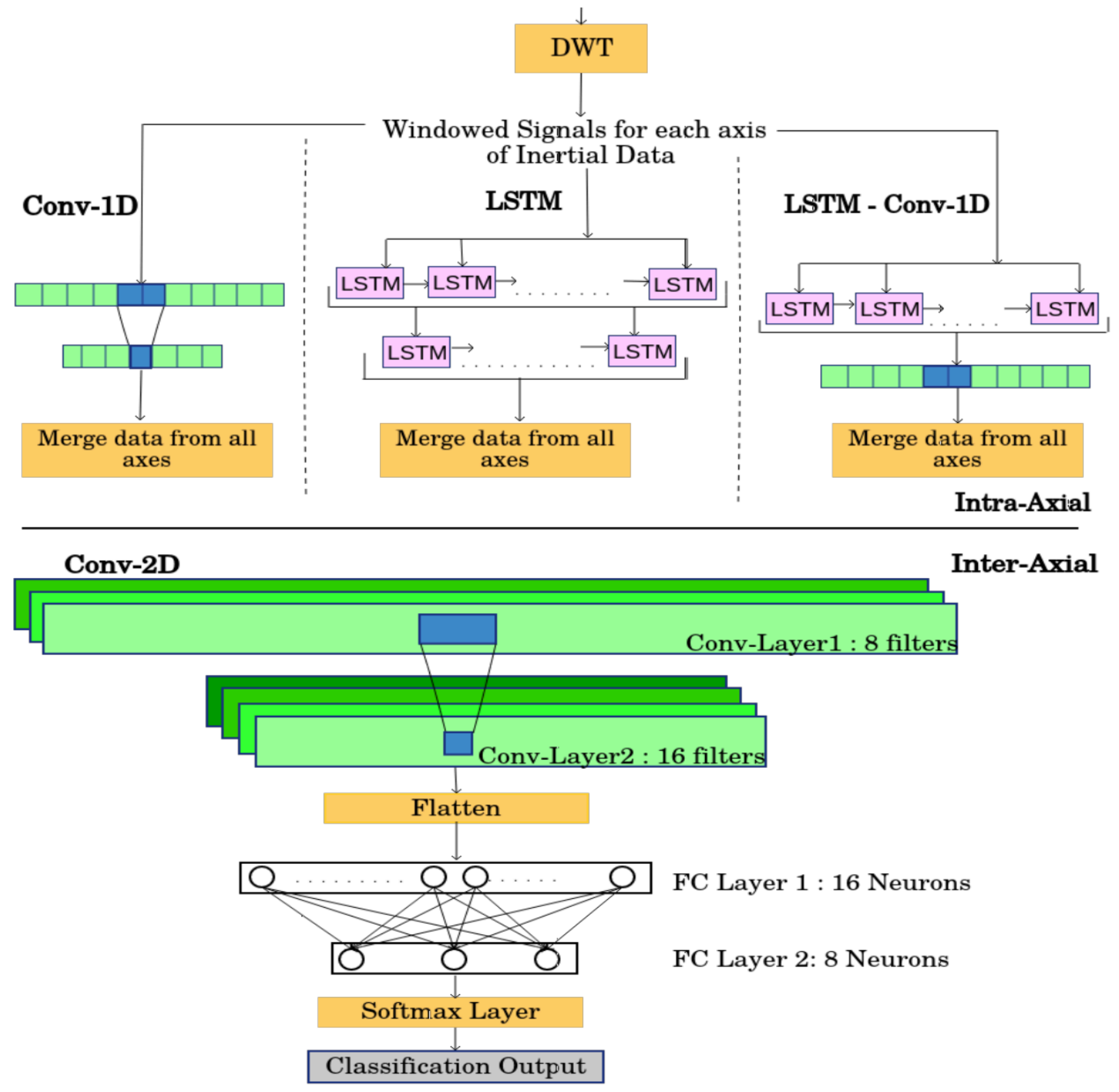
Inter-Axial Dependencies:

- ***Conv-2D:*** Capturing the interactions between data from three axes, thereby learning discriminative features across spatial dimensions.
- Two-layer stacked network (8 and 16 filters each) with 3x3 receptive field size with BatchNorm and a Max-pool layer of size 3x2

The intra-axial and inter-axial models are stacked together for effective extraction of patterns.

MODEL

- Two fully-connected (FC) layers of 16 and 8 neurons each, with Rectified Linear Unit (ReLU) activation functions are used.
- Dropout is used as a regularization mechanism after each FC with a probability of 0.25.
- Softmax (negative log likelihood) probability estimations are used for classification of activities.
- Adam optimizer is used to minimize the Categorical cross-entropy classification loss.



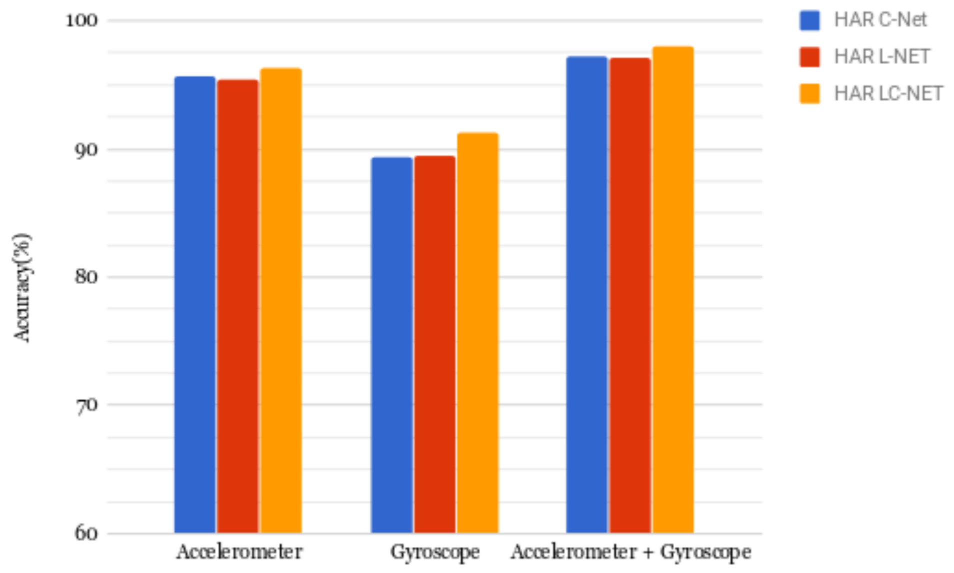
HARNet
VARIANTS

- Three variants of architectures:
  - ***HAR-CNet***: Conv-1D -> Conv-2D
  - ***HAR-LNet***: LSTM -> Conv-2D
  - ***HAR-LCNet***: LSTM -> Conv-1D -> Conv-2D

- Performance evaluation using three modes:
  - Mixed Mode
  - Device-Independent Mode
  - User-Independent Mode

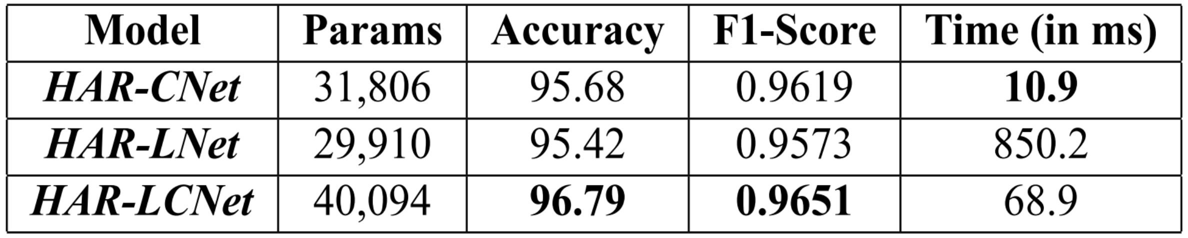
SENSOR MINIMIZATION

- Accuracies obtained from accelerometer + gyroscope are only ~1.5% higher than those of accelerometer alone across all three variants.
- Hence, only accelerometer data is considered, thereby resulting in ~50% data reduction.



MODEL PERFORMANCE

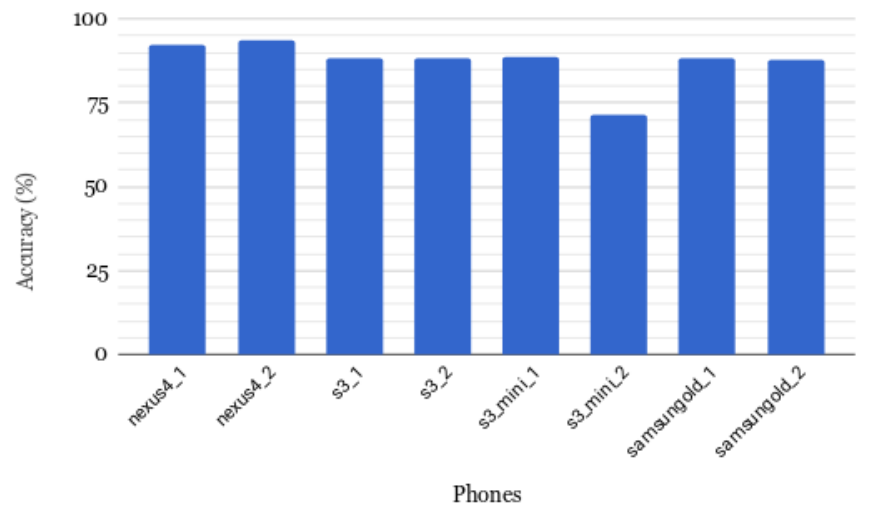
- *HAR-CNet* is ~7x faster than the next-best performing model, *HAR-LCNet* in classification time with just ~1% difference in accuracy and F1 scores.
- Hence, we consider ***HAR-CNet*** as our final model, considering the computations done on embedded/mobile platforms.



Model	Params	Accuracy	F1-Score	Time (in ms)
HAR-CNet	31,806	95.68	0.9619	**10.9**
HAR-LNet	29,910	95.42	0.9573	850.2
HAR-LCNet	40,094	**96.79**	**0.9651**	68.9

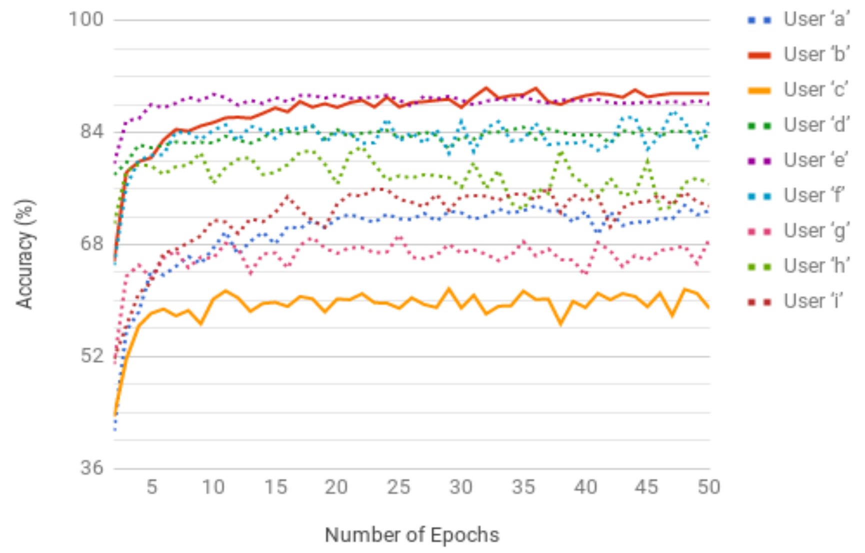
DEVICE-INDEPENDENT MODE

- To evaluate the model's generalizing capabilities across various heterogeneous devices, a Leave-One-Device-Out cross validation technique was used.
- The cross-val accuracy and F1-score was observed to be 89.5% and 0.887 respectively.



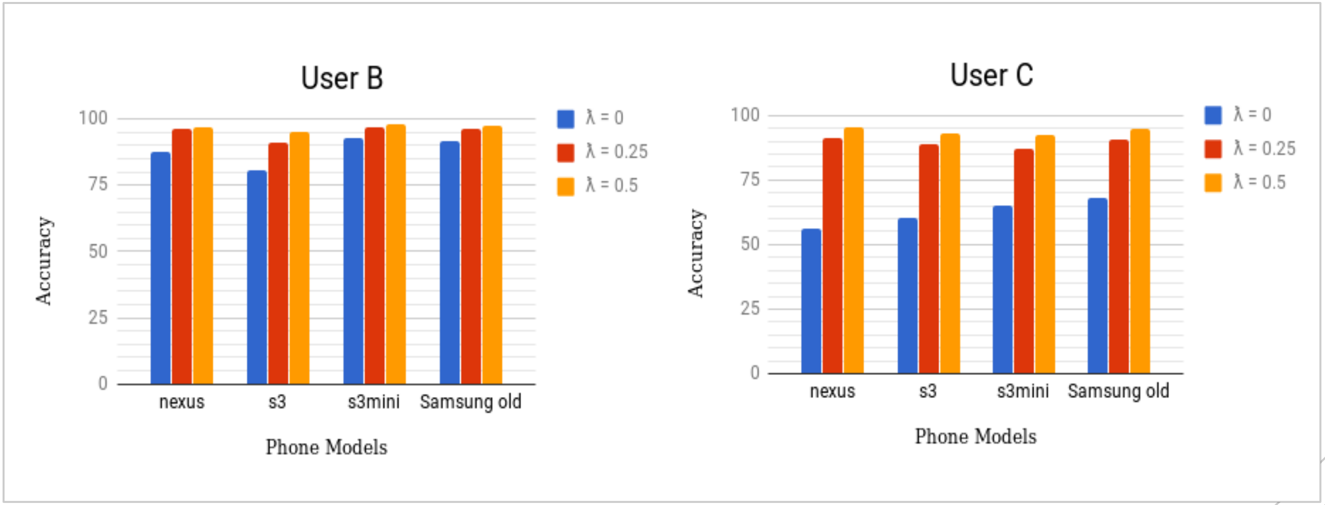
USER-INDEPENDENT MODE

- Stratified k-fold Leave-One-User-Out (testing on previously unseen users) cross validation.
- User ‘c’ achieves least accuracy which could attributed to physical build, posture and execution of activities. We hence perform Incremental Learning to enhance the accuracies of worst-performing users.



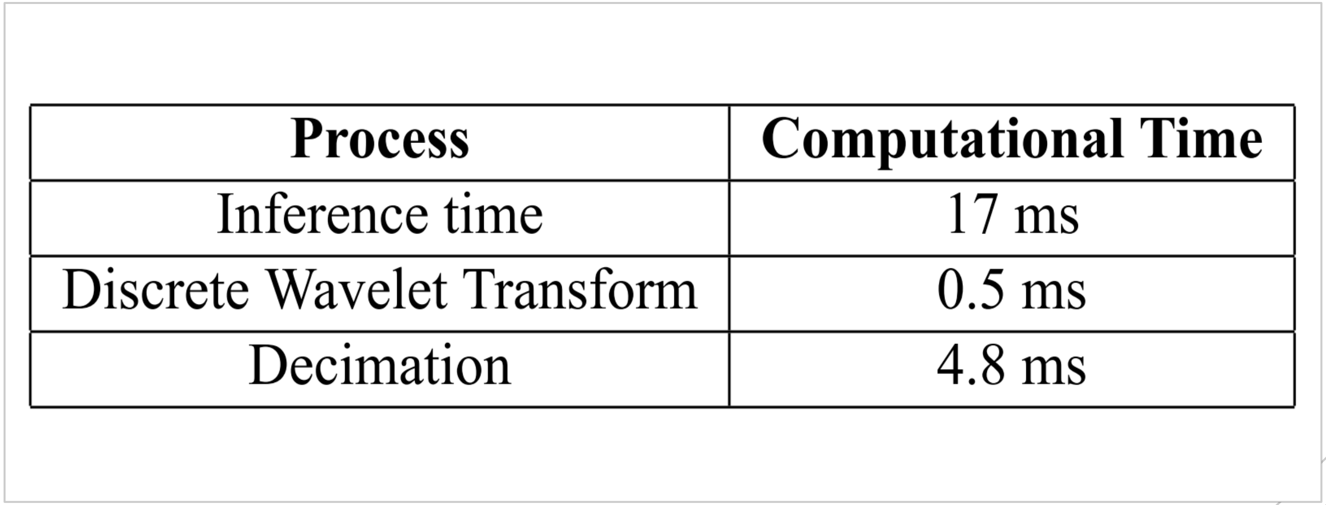
ON-DEVICE INCREMENTAL LEARNING

- Raspberry Pi 2 (similar H/W, S/W with predominant contemporary wearables), with the trained model weights being stocked.
- The portion of unseen users is governed by **adaptation factor λ**.
- Initially, with just λ=0.25, for worst-performing user 'c', accuracy substantially increases by ~35%.



ON-DEVICE INCREMENTAL LEARNING

- For a particular user in Incremental Learning, the model adapts to the user's recent behavioral pattern, thus leading to higher accuracies.
- 3 seconds per epoch.
- model size – ~0.5 MB.



Process	Computational Time
Inference time	17 ms
Discrete Wavelet Transform	0.5 ms
Decimation	4.8 ms

ActiveHARNet: Towards On-Device Deep Bayesian Active Learning for Human Activity Recognition

Gautham Krishna et al., ActiveHARNet: Towards On-Device Deep Bayesian Active Learning for Human Activity Recognition, EMDL ’19@ACM MobiSys ‘19

PROMINENT CHALLENGES IN ON-DEVICE HAR

Label Acquisition

- Real-time acquisition of labels (ground truthing) is hard
- Labelling load on oracle (user) needs to be reduced

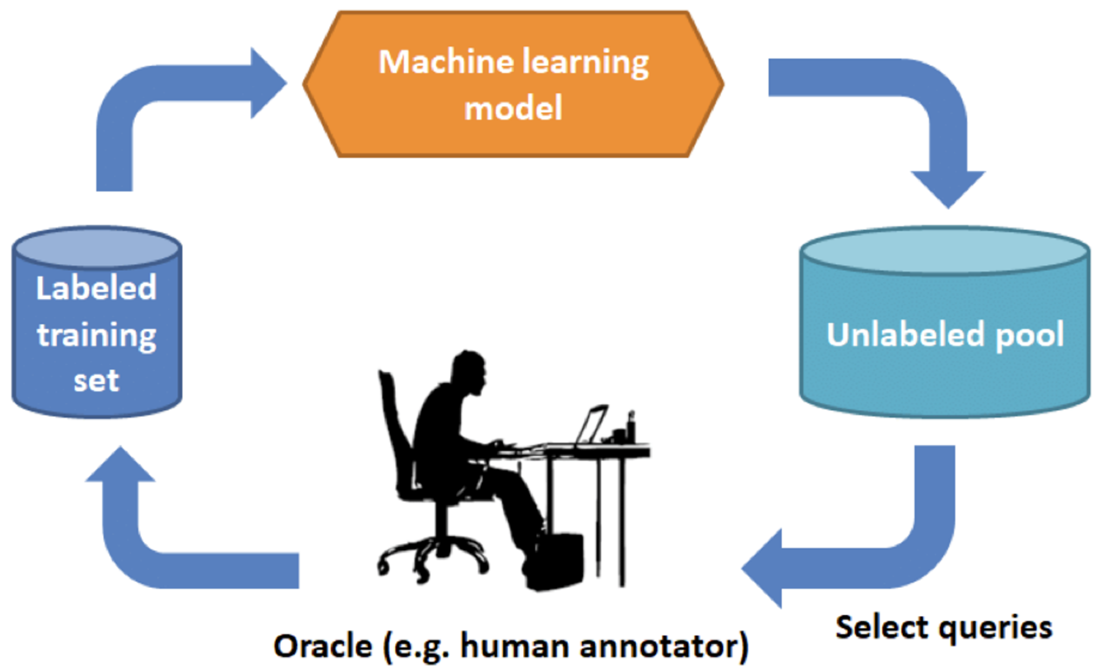
GOALS OF OUR PROPOSED SYSTEM

- A generic HAR model which handles **Incremental Learning** on wearables, and is **resource-friendly**
- **Active Learning**, which queries the oracle only necessary (most-informative) labels on-device
- Facilitate **User Adaptability**
- Test the generalizing Incremental Active Learning capabilities together on **HAR** and **Fall Detection** tasks

GOALS OF OUR PROPOSED SYSTEM

But,
Why Active Learning?

- A generic HAR model which handles **Incremental Learning** on wearables, and is **resource-friendly**
- **Active Learning**, which queries the oracle only necessary (most-informative) labels on-device
- Facilitate **User Adaptability**
- Test the generalizing Incremental Active Learning capabilities together on **HAR** and **Fall Detection** tasks



- A big challenge in many applications is obtaining labelled data.
- **Active Learning (AL), over unsupervised techniques, can be used predominantly to substantiate the confidence on the queried data points.**
- Instead of labeling hundreds of activities, an ideal system should query few labels in each activity.

ACTIVE LEARNING

BAYESIAN NEURAL NETS (BNNs)

- Offer a probabilistic interpretation to deep learning models.
- Incorporate Gaussian prior (probability distributions p(ω)) over our model parameters ω.
- Can possess and **model uncertainty information**.

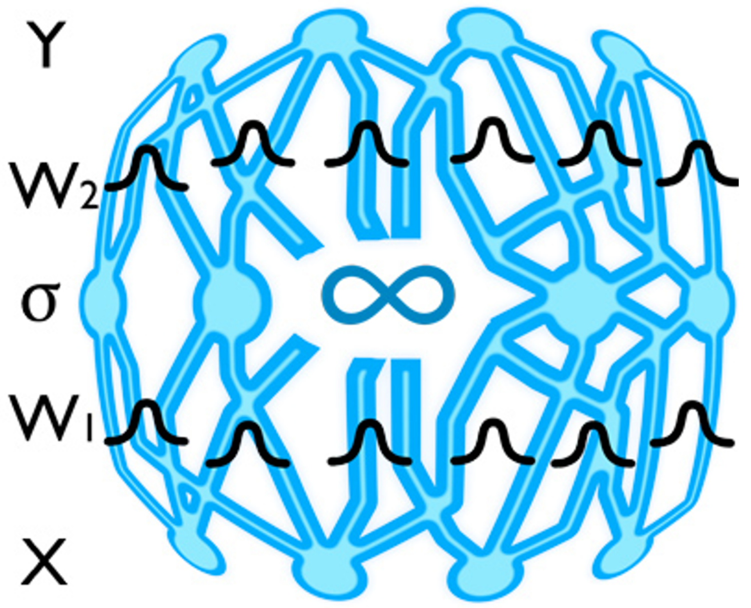


Fig. taken from Prof. Yarin Gal’s blog

MODELING UNCERTAINTIES USING DROPOUT

- **Dropout** - a stochastic regularization technique can perform approximate inference over a deep Gaussian process
- Learns the model posterior uncertainties **without high computational complexities** over few stochastic iterations at both train/test times
- Termed Monte-Carlo Dropout **(MC-Dropout)**
- Equivalent to performing Variational Inference
- $p(y* | x*, D_{train}) = \int p(y* | x*, \omega)\, p(\omega | D_{train})\, d\omega$

Yarin Gal et al., Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, ICML ‘16

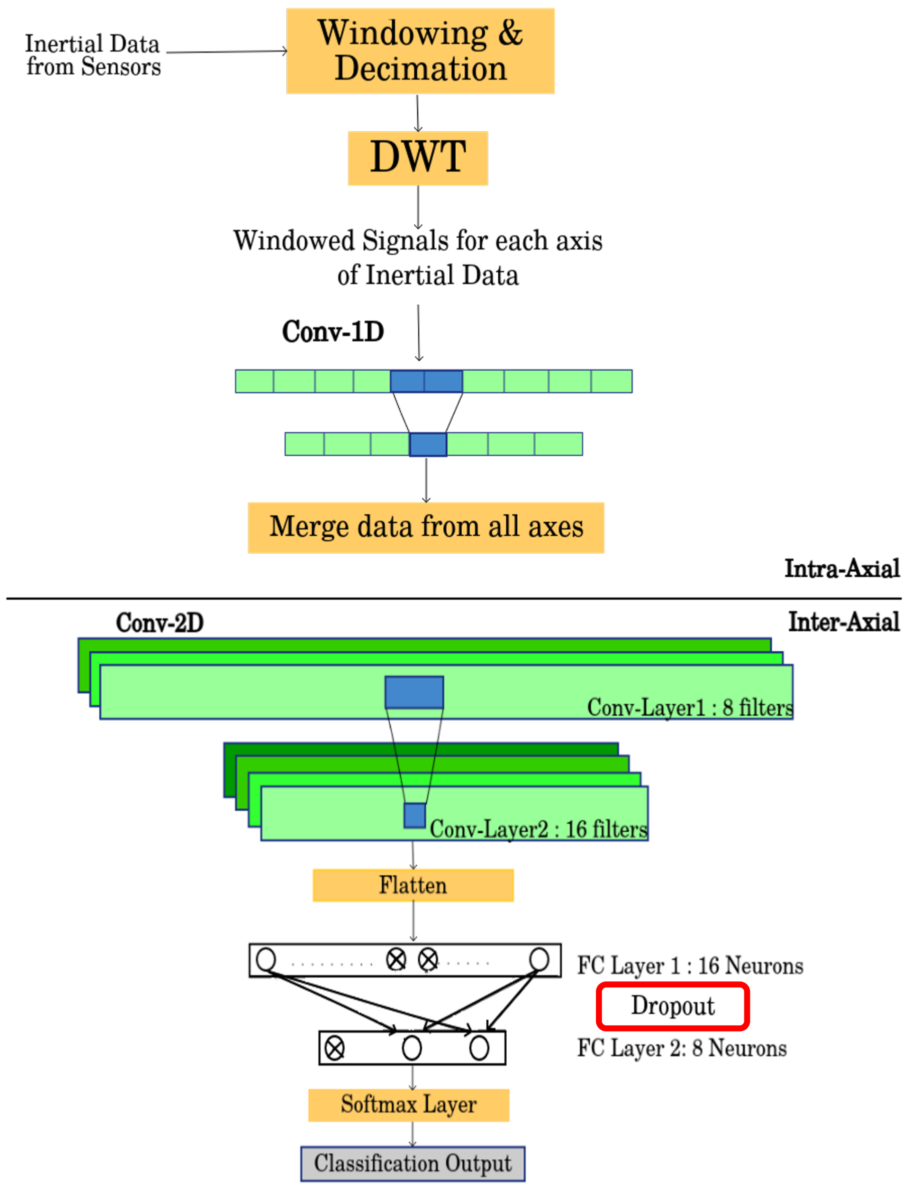
MODELING UNCERTAINTIES USING DROPOUT

- **Dropout** - a stochastic regularization technique can perform approximate inference over a deep Gaussian process
- Learns the model posterior uncertainties **without high computational complexities** over few stochastic iterations at both train/test times
- Termed Monte-Carlo Dropout **(MC-Dropout)**
- Equivalent to performing Variational Inference
- $p(y* | x*, D_{train}) = \int p(y* | x*, \omega) \underbrace{\boxed{p(\omega | D_{train})}}_{\boxed{\text{Posterior}}} d\omega$

Yarin Gal et al., Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, ICML '16

BAYESIAN HARNet ARCHITECTURE

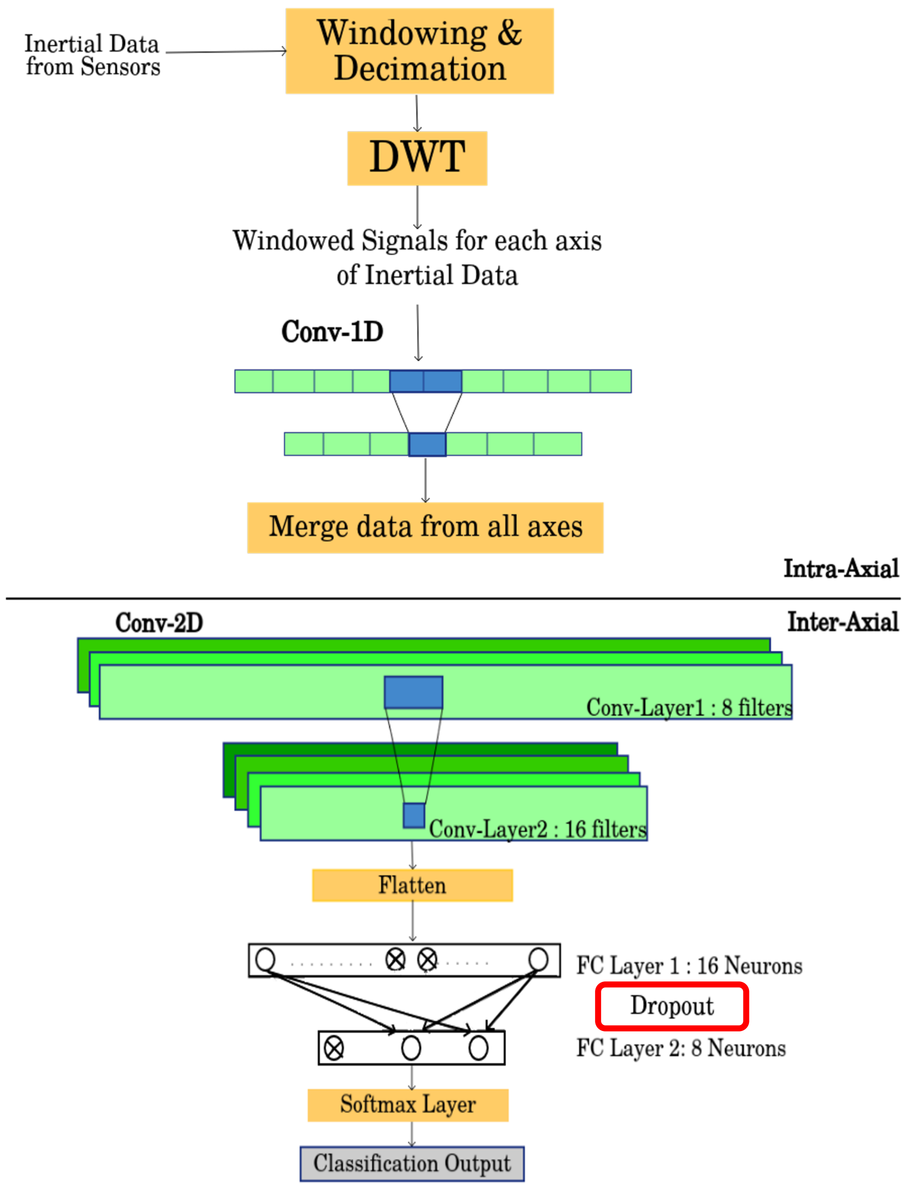
- Utilize *HARNet* architecture and treat it as a Bayesian Neural Net (with Dropout).
- Intra-Axial and Inter-Axial dependencies exploited using stacked Conv-1D and Conv-2D architectures.
- Pre-processing techniques – Windowing, Decimation (down-sampling) and Discrete Wavelet Transform (DWT).
- Conv-1D to extract characteristics within each axis (X, Y, Z of accelerometer data).
- Conv-2D to capture interactions between data from three axes, thereby learning discriminative features across spatial dimensions.



BAYESIAN HARNet ARCHITECTURE

- Two stacked Conv-1D layers with 8 & 16 filters each size 2, BatchNorm, Max-Pool size 2 (Intra-axial)
- Two stacked Conv-2D layers with 8 & 16 filters each size 3x3, BatchNorm, Max-Pool size 3x2 (Inter-axial)
- Two Fully-Connected Layers with 16 & 8 neurons each and ReLU activations.
- Dropout drop probability of 0.3.
- Softmax Layer to estimate probability scores
- Categorical cross-entropy loss with Adam Optimizer

Refer paper for more details:



Gautham Krishna Gudur er al., ActiveHARNet: Towards On-Device Deep Bayesian Active Learning for Human Activity Recognition, EMDL '19

ACQUISITION FUNCTIONS

- Uncertainty measures from Bayesian *HARNet* need to be quantified
- Arriving at most efficient set of data points (select k from n) to query from D_{pool}

ACQUISITION FUNCTIONS

Given incoming data point x and unknown label y with data D and parameters ω,

- *Max Entropy:* Maximize predictive entropy

  $H[y|x,D] := -\sum_c p(y = c|x,D) \log p(y = c|x,D)\ c$

- *BALD (Bayesian Active Learning by Disagreement):* Maximize mutual information between predictions and model posterior

  $I[y,\omega|x,D] = H[y|x,D] - E_{p(\omega|D)} H[y|x,\omega]$

- Maximize *Variation Ratios:*

  $\text{variation-ratio}[x] := 1 - \max p(y|x,D)\ y$

- *Random Acquisitions:* Select data points from pool uniformly at random.

DATASETS USED

Heterogeneous Human Activity Recognition (HHAR) Smartwatch Dataset

Smart Devices are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition, SenSys '15

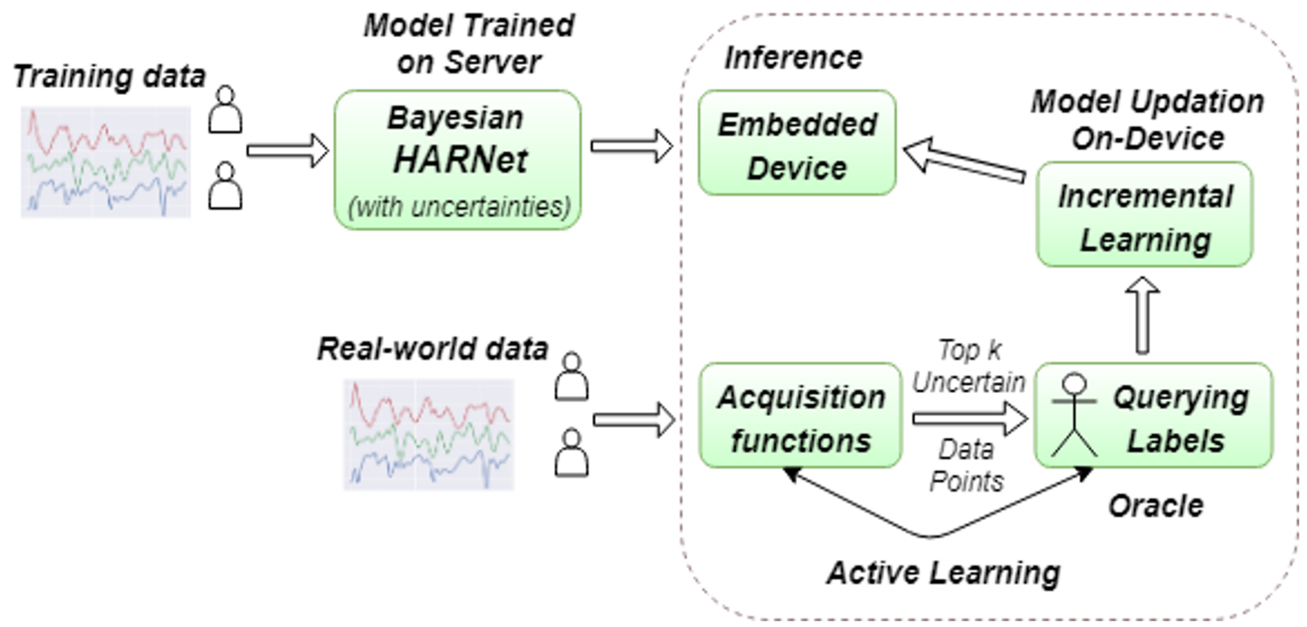
- Utilizing accelerometer data from different wearables - two LG G smartwatches and two Samsung Galaxy Gears across nine users performing six activities: Biking, Sitting, Standing, Walking, Stairs-Up, Stairs-Down in real- time heterogeneous conditions.

Notch Wrist-worn Fall Detection Dataset

Smartfall: A smartwatch-based fall detection system using deep learning, Sensors '18

- Utilizing wrist-worn accelerometer data from an off-the-shelf Notch sensor by seven volunteers across various age groups performing simulated falls and activities (activities are termed as not-falls) .

ActiveHARNet ARCHITECTURE



Gautham Krishna Gudur et al. ActiveHARNet: Towards On-Device Deep Bayesian Active Learning for Human Activity Recognition, EMDL '19

- User-Independent Incremental Active Learning is experimented on Raspberry Pi 2 (similar H/W, S/W with predominant contemporary wearables), with the trained model weights being stocked.
- The number of acquisition pool windows used for incremental active training can be governed by the **acquisition adaptation factor η ∈ [0, 1]**.

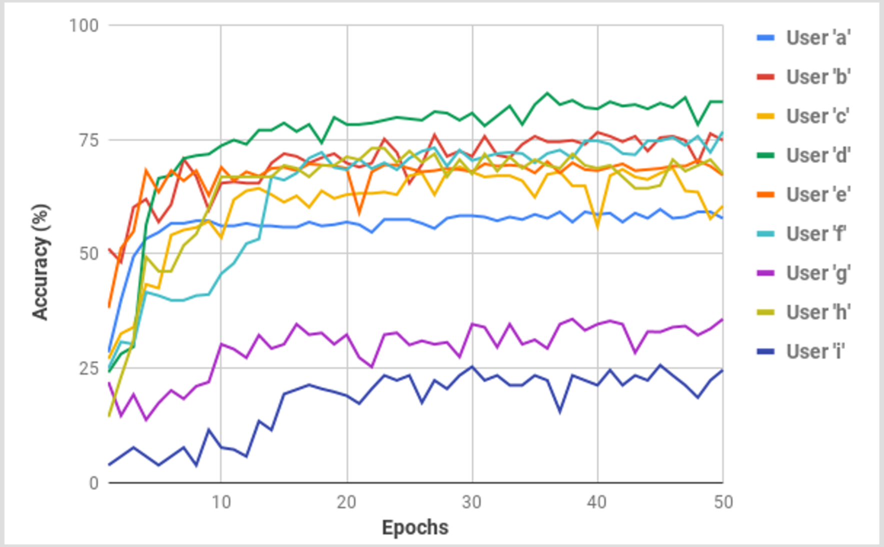
BASELINE EFFICIENCIES using Bayesian *HARNet*

- A stratified k-fold *Leave-User-Out* (testing on previously unseen users) cross validation technique was used for evaluating User Adaptability.
- Unseen user data split into test and pool data, pool treated as real-world data, test is untouched.

HHAR

User 'd' – 84%; User 'g' – 36%; User 'i' - 25%

Average – 61%



Notch

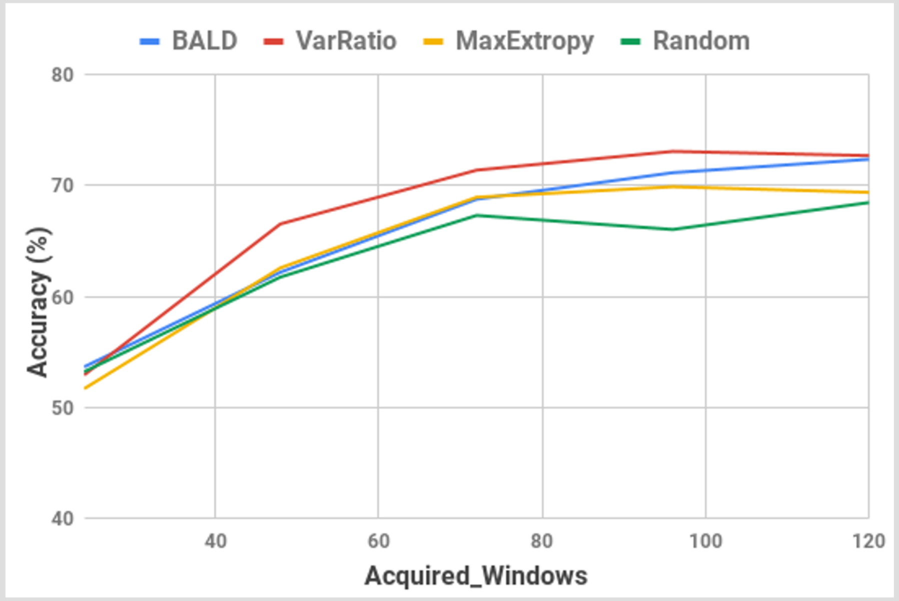
Average f1 – 0.927

f1-score used since fall is a very rare-class

	User 1	User 2	User 3	User 4	User 5	User 6	User 7
f1-score	0.9326	0.9214	0.9357	0.9372	0.9195	0.9229	0.9248
Accuracy	97.02	94.44	94.05	95.36	94.08	94.59	94.65

ActiveHARNet on HHAR

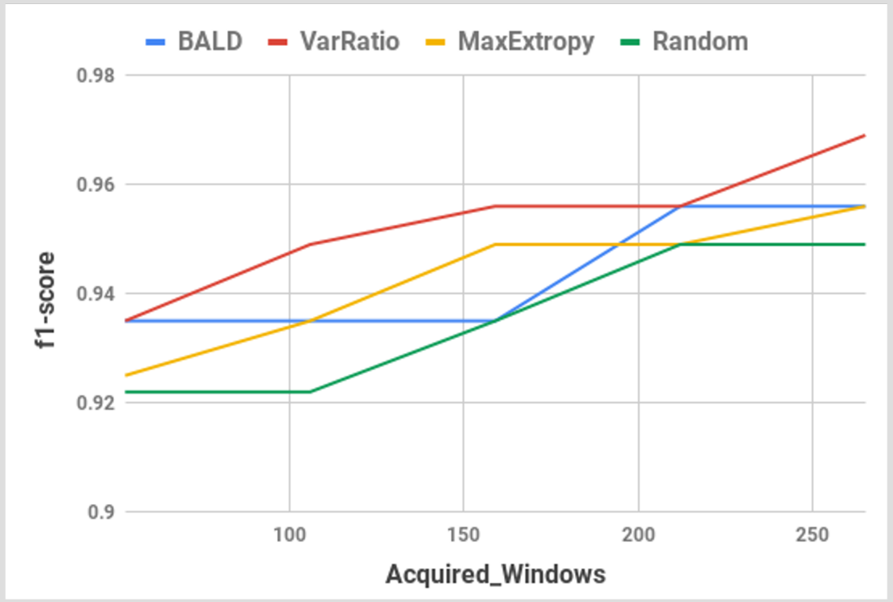
- Variation Ratios (VR) acquisition function performs the best.
- User ‘i’ (least performing) – accuracy increase from 25% - 70% with just ~60 pool points. ~49% (η=0.49 - 60 pool points) of total 123 data points gives this 45% accuracy increase. With all 123 data points (100% - η=1.0), gives 73% accuracy.
- All users average: 61% (η=0) to 86% (η=1) for VR. η=0.4 gives near-equal 85.87%.



η	User a	User b	User c	User d	User e	User f	User g	User h	User i	Avg.
0.0	57.83	74.86	60.5	83.79	67.25	76.77	35.78	67.5	24.66	61
0.2	83.52	89.76	75.7	91.95	81.53	79.79	73.39	78.75	52.92	78.59
0.4	89.15	91.72	80.85	92.3	85.05	84.57	76.23	81	66.53	83.05
0.6	91.55	92.18	82.26	93.26	87.92	86.96	77.15	83.5	71.38	85.13
0.8	92.64	93.24	82.28	93.56	87.52	88.07	78.58	82.6	73.07	85.73
1.0	92.72	93.16	85.06	93.64	89.95	87.96	76.23	81.375	72.69	85.87

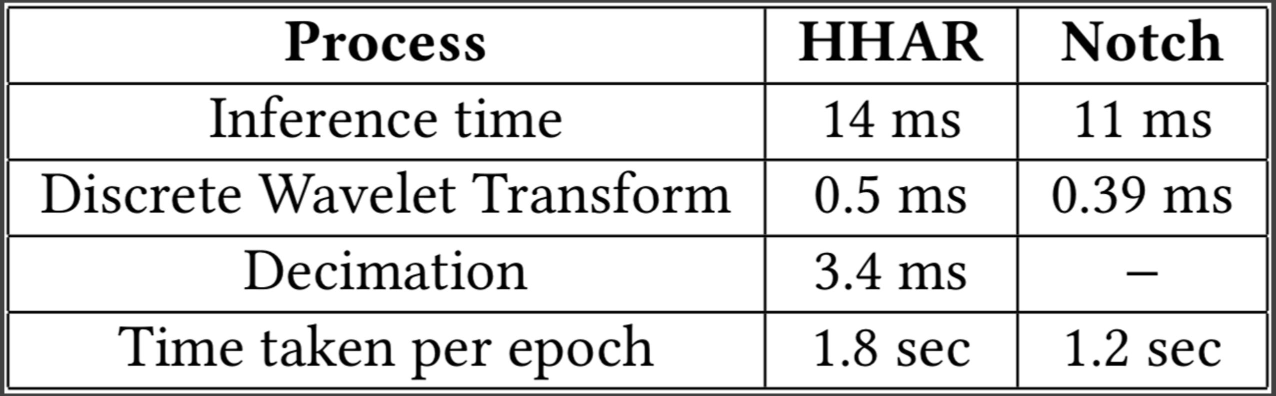
ActiveHARNet on Notch

- Variation Ratios (VR) acquisition function again performs the best here.
- User 5 (least performing) – f1-score increases from ~0.92 - 0.95 with just 150 pool points (η=0.4). With all 265 data points (100% - η=1.0), gives 0.969 f1-score.
- All users average: 0.928 (η=0) to 0.943 (η=0.4) and to 0.948 (η=0.6) for VR.



η	**User 1**	**User 2**	**User 3**	**User 4**	**User 5**	**User 6**	**User 7**	**Avg.**
0.0	0.932	0.921	0.936	0.937	0.92	0.923	0.925	0.928
0.2	0.938	0.924	0.945	0.947	0.935	0.932	0.925	0.935
0.4	0.943	0.929	0.961	0.952	0.949	0.932	0.932	0.943
0.6	0.949	0.929	0.965	0.952	0.956	0.945	0.936	0.948
0.8	0.943	0.937	0.968	0.965	0.956	0.953	0.942	0.952
1.0	0.952	0.937	0.965	0.956	0.969	0.945	0.936	0.951

INCREMENTAL ACTIVE LEARNING



Process	HHAR	Notch
Inference time	14 ms	11 ms
Discrete Wavelet Transform	0.5 ms	0.39 ms
Decimation	3.4 ms	–
Time taken per epoch	1.8 sec	1.2 sec

- HHAR takes a model size of 315 kB, Notch takes 180kB.
- T=10 stochastic dropout iterations (1.4 sec per iteration) were used, hence total ~ 14 seconds for Bayesian Active Learning.
- Number of data points collected for each acquisition iteration can be bounded based on **time** or **acquired count (number of data points)** criterion.
- **Time** is proposed as preferred metric, i.e., periodic updates at fixed intervals – since oracle would only be able to remember recent trends in activities.
- Cannot guarantee users to perform activities within given time frame, hence thresholding based on count of data points is not recommended.

A Vision-based Deep On-Device Intelligent Bus Stop Recognition System

Gautham Krishna et al., A Vision-based Deep On-Device Intelligent Bus Stop Recognition System, PURBA ’19@ACM UbiComp ‘19

BUS STOP RECOGNITION

Intelligent Public Transportation Systems - cornerstone to any smart city, particularly ***Autonomous Vehicles***.

Over 47% of people use buses as preferred public transport mode in the United States.

~76% of buses have automatic bus stop announcements.

In India, buses used to take over 90% of public transport in Indian cities.

BUS STOP RECOGNITION IN INDIA

- Second largest road network in the world.
- Conventionally, bus conductor intimates (*whistles*) when a bus stop arrives and announces its location aloud.
- Driver halts the bus.
- New sophisticated buses consist of pre-defined queues – the sequence of bus stops are pre-loaded.
- Easier alternative to the conductor’s manual announcement of bus stops.

CHALLENGES IN AUTOMATIC BUS STOP RECOGNITION

Really difficult to identify the arrival of a bus stop on-the-fly for buses to appropriately halt and notify its passengers.

Global Positioning System (GPS) look-up can be used for bus stops identification, however latency issues in the network.

Hard to localize, identify and halt the vehicle right in front of the bus stop using GPS.

CHALLENGES IN AUTOMATIC BUS STOP RECOGNITION IN INDIA

The landscapes and surroundings of bus stops dynamically change and evolve over time.

Rural and sub-urban Indian bus stops predominantly do not have bounded or localized spaces/lanes or even proper bus stops

Bus routes are periodically revised and repurposed based on demand and traffic patterns.

Necessary to not only know the sequence of bus stops, but also intelligently perceive the location of bus stop in order to halt at right location.

GOALS OF THE PROPOSED SYSTEM

To employ novel vision-based techniques to recognize bus stops on-device and eliminate the latency of a GIS/GPS

look-up

The model should be able to handle new and existing bus stops

Incremental Learning/Data Augmentation

Existing Bus Stops

To incrementally handle the ever-changing surroundings of existing bus stops

New Bus Stops

To incorporate new bus stops with minimal/no human intervention, without performance degradation due to class imbalance

To automatically handle real-time ground truthing/labeling of bus stops

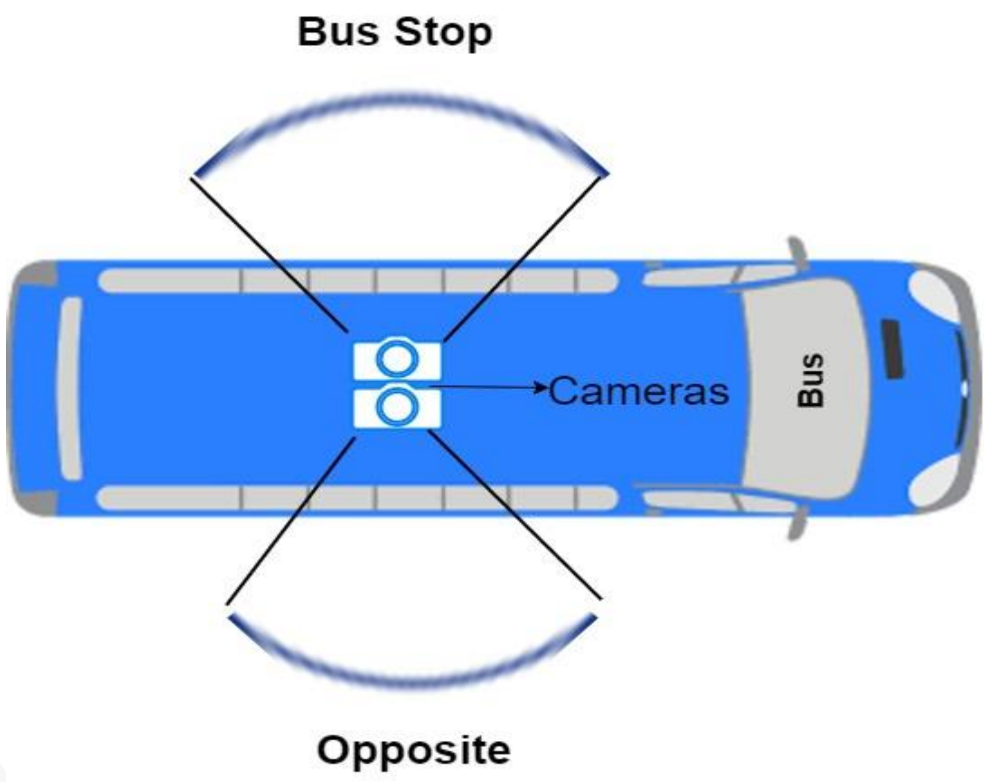
Crowdsourcing/Active Learning

REAL-TIME DESIGN AND INFERENCE

- Unnecessary overheads in capturing images during the whole route during classification of bus stop (inference).
- Hence, the images are captured and classified only when speed of the bus is below a certain ideal threshold.
- The real-time speed can be acquired from the speedometer of the bus.
- The ideal minimum threshold (10 km/hr for instance) is subject to locality and traffic conditions.
- We propose two different classifiers – day and night classifier (differentiated using a light sensor).

DATASET

- The images of the bus stops were collected in and around the city of Chennai, India.
- Images were acquired using two 5 MP cameras placed in opposite directions.
- 8 bus stops during the day – 5 public urban and 3 rural bus stops.
- Images from 3 urban bus stops were also collected during the night.
- 90 images in each bus stop, 45 in each direction were collected.



EXAMPLES FROM THE DATASET

(BAYESIAN) CONVOLUTIONAL NEURAL NETWORKS

CNNs are powerful mechanisms for distinctive spatial representations and offer automatic, effective feature learning capabilities.

The acquired bus stops in real-time requires identification of discriminative features to handle evolving landscape changes.

This essentially is a scene classification problem, which CNNs are known to effectively handle.

Bayesian (Convolutional) Neural Networks are probability distributions (Gaussian priors) instead of point estimates, and this helps in modelling uncertainties.

MODELING UNCERTAINTIES USING DROPOUT

- **Dropout** - a stochastic regularization technique can perform approximate inference over a deep Gaussian process
- Learns the model posterior uncertainties **without high computational complexities** over few stochastic iterations at both train/test times
- Termed Monte-Carlo Dropout **(MC-Dropout)**
- Equivalent to performing Variational Inference
- $p(y* | x*, D_{train}) = \int p(y* | x*, \omega)\, p(\omega | D_{train})\, d\omega$

Yarin Gal et al., Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, ICML '16

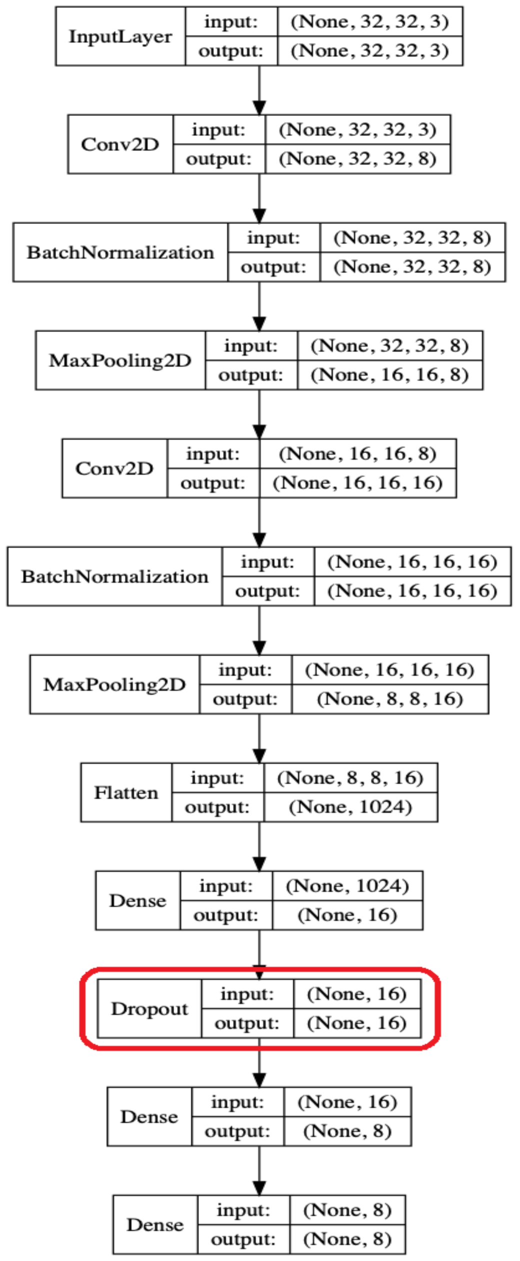
MODELING UNCERTAINTIES USING DROPOUT

- **Dropout** - a stochastic regularization technique can perform approximate inference over a deep Gaussian process
- Learns the model posterior uncertainties **without high computational complexities** over few stochastic iterations at both train/test times
- Termed Monte-Carlo Dropout **(MC-Dropout)**
- Equivalent to performing Variational Inference
- $p(y* | x*, D_{train}) = \int p(y* | x*, \omega) \underbrace{\boxed{p(\omega | D_{train})}}_{\boxed{\text{Posterior}}} d\omega$

Yarin Gal et al., Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, ICML '16

Bayesian CNN Architecture

- The images were resized to 32 × 32 × 3 and normalized (divided RGB pixel values by 255 for easier model convergence).
- Utilize the CNN architecture, and treat it as a Bayesian Neural Net (with Dropout).
- We utilize 2 stacked CNN layers with BatchNorm and MaxPool layers between each layers, followed by two Fully-Connected Dense layers, with dropout of probability 0.3 between each FC layer.
- This is followed by a Linear Softmax layer, governed by the categorical cross-entropy loss.


InputLayer | input: (None, 32, 32, 3) | output: (None, 32, 32, 3)
Conv2D | input: (None, 32, 32, 3) | output: (None, 32, 32, 8)
BatchNormalization | input: (None, 32, 32, 8) | output: (None, 32, 32, 8)
MaxPooling2D | input: (None, 32, 32, 8) | output: (None, 16, 16, 8)
Conv2D | input: (None, 16, 16, 8) | output: (None, 16, 16, 16)
BatchNormalization | input: (None, 16, 16, 16) | output: (None, 16, 16, 16)
MaxPooling2D | input: (None, 16, 16, 16) | output: (None, 8, 8, 16)
Flatten | input: (None, 8, 8, 16) | output: (None, 1024)
Dense | input: (None, 1024) | output: (None, 16)
Dropout | input: (None, 16) | output: (None, 16)
Dense | input: (None, 16) | output: (None, 8)
Dense | input: (None, 8) | output: (None, 8)

INCREMENTAL LEARNING

- Used to update existing model with recent bus stop images which might have evolved with landscape changes.
- Will emphasize on learning the most recent and salient features of that bus stop.
- The inherent bias towards the model updated with the recently acquired images is favorable.

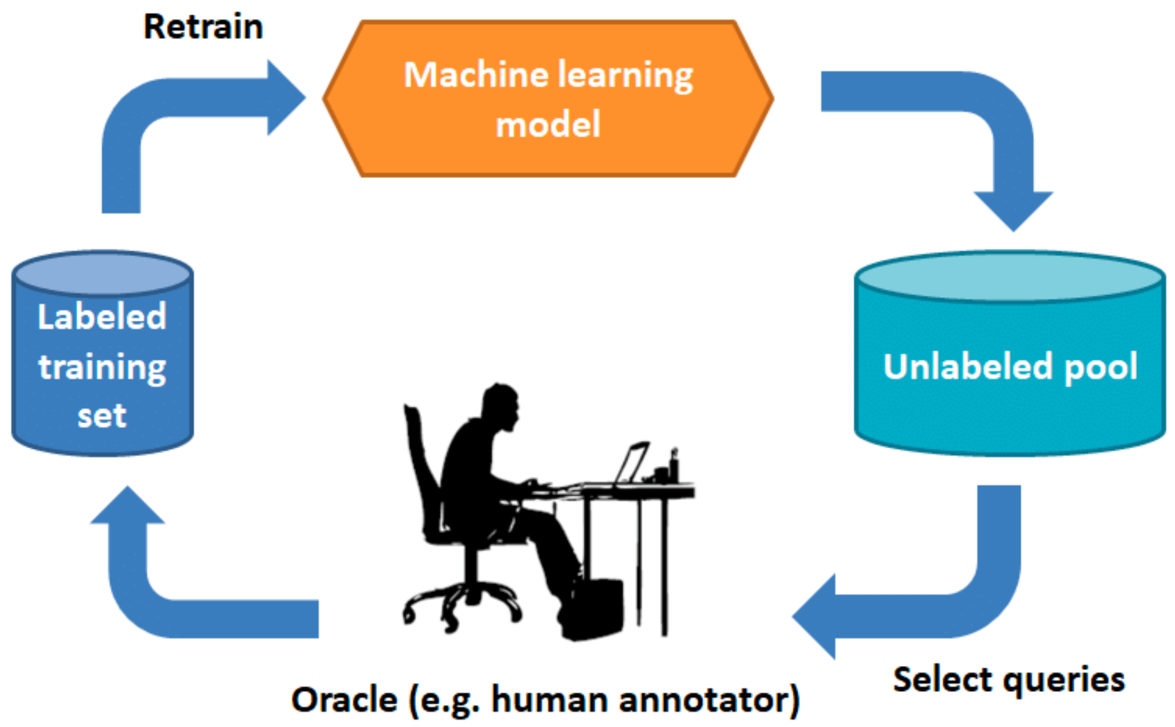
But, Why?

INCREMENTAL LEARNING

- Used to update existing model with recent bus stop images which might have evolved with landscape changes.
- Will emphasize on learning the most recent and salient features of that bus stop.
- The inherent bias towards the model updated with the recently acquired images is favorable.

But, Why?

- In a single bus stop, only the recently acquired data is sufficient to make accurate predictions, since the information from newer images are added periodically to the model.
- Hence, the memory and neural footprint of older images is not necessary.



- A big challenge in many real-time applications is obtaining labelled data.
- Active Learning/Crowdsourcing, over unsupervised techniques, is used predominantly to substantiate the confidence on the queried data points.
- Instead of labelling hundreds of bus stop images, an ideal system should query few labels in each bus stop.
- For instance, Google Waze (complementary to Google Maps).

ACTIVE LEARNING

ACQUISITION FUNCTIONS

- Uncertainty measures from Bayesian *HARNet* need to be quantified
- Arriving at most efficient set of data points (select k from n) to query from D_{pool}

ACQUISITION FUNCTIONS

Given incoming data point x and unknown label y with data D and parameters ω,

- *Max Entropy:* Maximize predictive entropy

  $H[y|x,D] := -\sum_c p(y = c|x,D) \log p(y = c|x,D)$ c

- *BALD (Bayesian Active Learning by Disagreement):* Maximize mutual information between predictions and model posterior

  $I[y,\omega|x,D] = H[y|x,D] - E_{p(\omega|D)} H[y|x,\omega]$

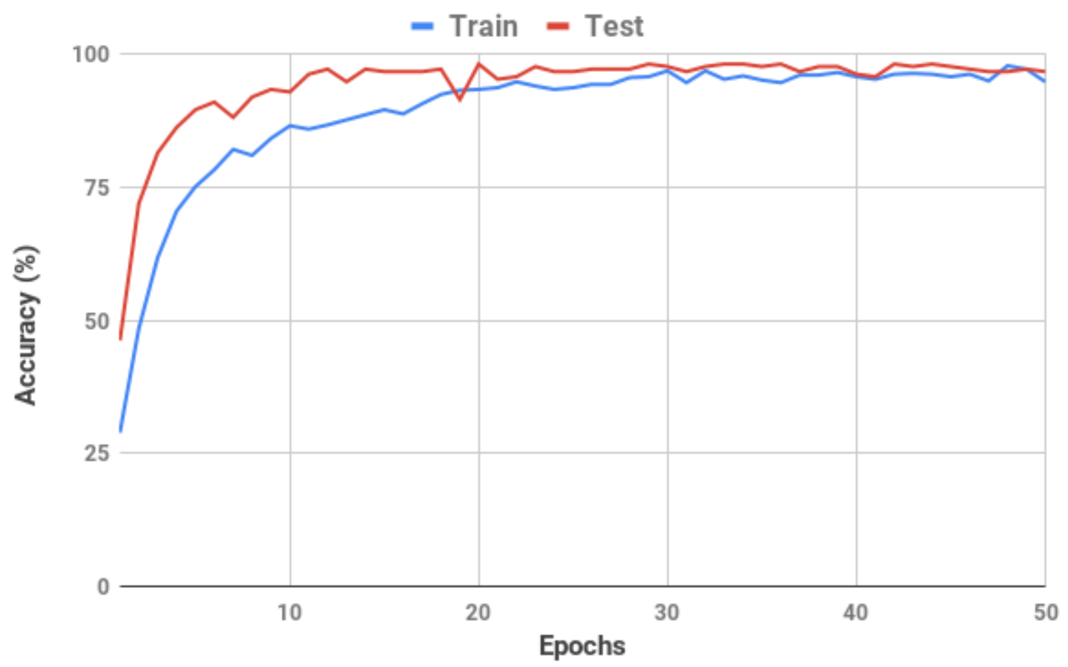
- Maximize *Variation Ratios:*

  $\text{variation-ratio}[x] := 1 - \max p(y|x,D)$ y

- *Random Acquisitions:* Select data points from pool uniformly at random.

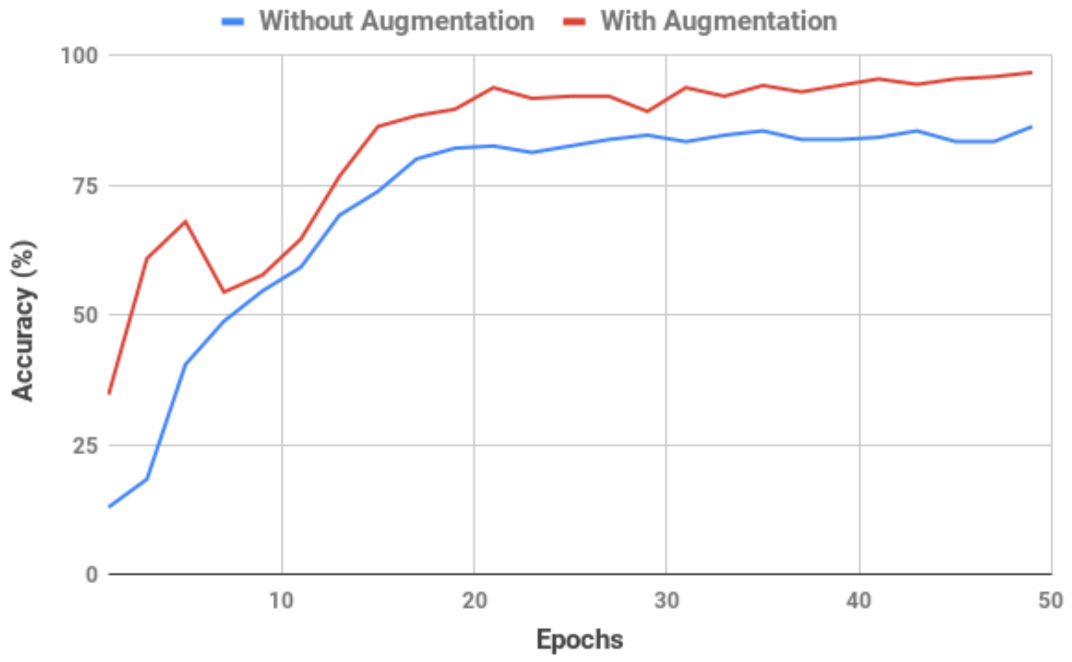
DATA AUGMENTATION

- When a new bus stop is added, it becomes a necessity for the model to scale and handle the incoming data of the new class.
- The acquired images from the new bus stop is predominantly lesser than that of existing classes which results in *class imbalance*.
- Techniques like zoom, shear and rotation (small angle) used for generating new images, which almost resemble the images acquired from a real-time camera to ensure stratified training across all classes.



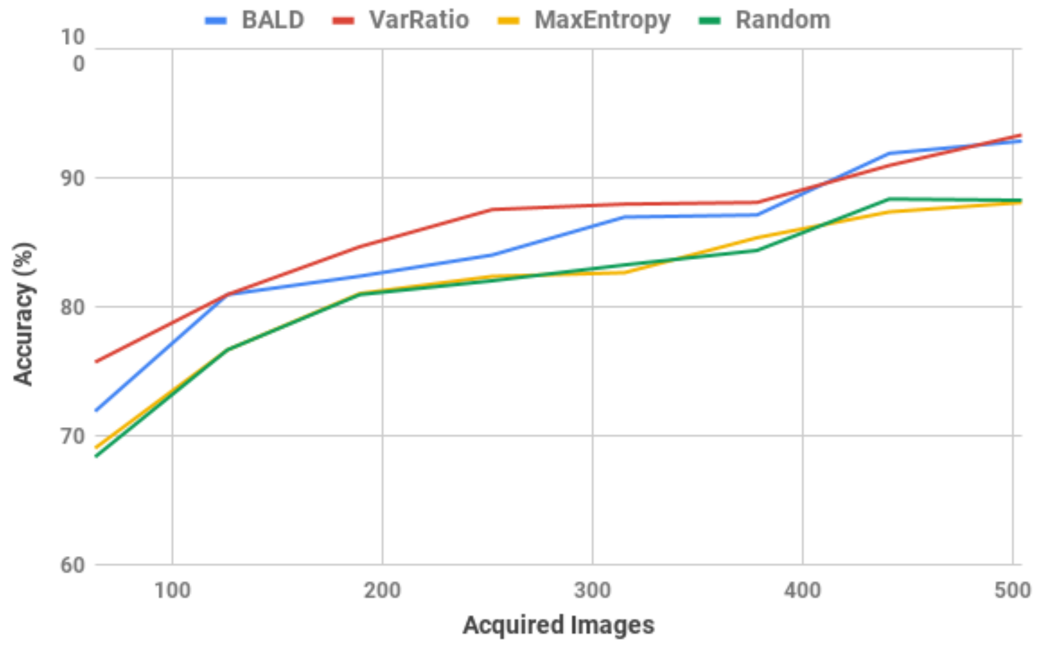
- The system is deployed on a Raspberry Pi 2 in real-time, and the stocked model weights are updated on-device in an incremental manner.
- Initially, we train the model with only 7 bus stops (B1, B2, … B7) and call them existing classes, while the 8th class (B8) is treated as the new unseen bus stop – illustrates scalability.
- Training and testing – high accuracies of ~97% and ~96% respectively.

EXPERIMENT AND BASELINE RESULTS



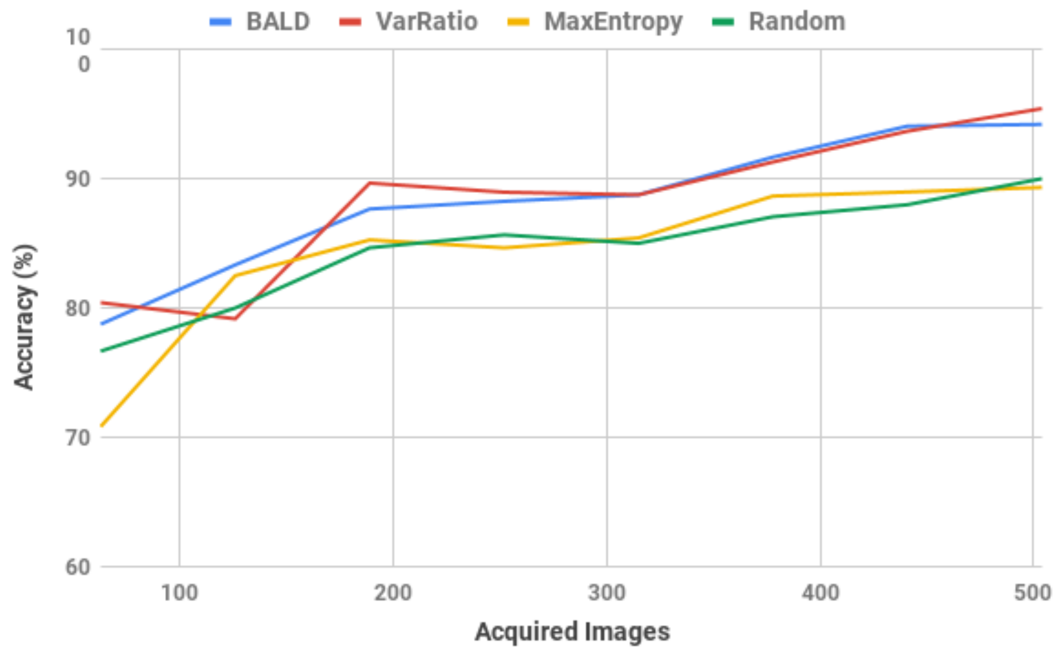
- Initially, we assume only 4 images were collected from either side of the new bus stop.
- With just the 8 data points, accuracy – 86.25%.
- However, the model when augmented with new B8 images sufficiently overcome class imbalance.
- Achieves an accuracy of 96.7% which is a ~10.5% increase in accuracy.
- Effectiveness of data augmentation strategies for new classes – ensures scalability of bus stops.

RESULTS WITH & WITHOUT AUGMENTATION



- The training data with existing 7 classes (B1, B2….B7) are split into pool (D_{pool}) and train (D_{train}).
- The initial accuracy with just 20% of train data is observed to be 64.28%.
- Variation Ratios (VR) performs the best, achieving ~88% with just less than 250 data points (less than 50% of total D_{pool}).

INCREMENTAL ACTIVE LEARNING EXISTING CLASSES



- Similar training mechanism after data augmentation with (B1, B2….B8).
- Variation Ratios again performs the best again, with a classification accuracy of ~90% with just ~180 images (~37% of total D_{pool}).
- A good trade-off between accuracy and images actively acquired.
- After the first acquisition iteration, would typically require very few actively queried data points to achieve on-par classification accuracies of ~96%.

INCREMENTAL ACTIVE LEARNING AUGMENTED CLASSES

INTELLIGENT INFERENCE

- Ideology is to instill ***biomimicry*** -- just like a human brain towards human-like behavior – not like conventional classification.
- Model acquires and classifies multiple iterative bus stop images on demand, until it can assure a confidence of at least α – ratio of mode of predicted classes to n.
- Typically, the threshold for α is set to a majority among the classified ($\alpha > 0.5$ for 2 images, $\alpha \geq 0.67$ for 3 images, and so on).
- The number of images captured and classified during inference (n), is initially set to 2 and capped at 10 – ($2 < n < 10$).
- The proposed inference mechanism would steer the model towards near-100% accuracy.
- Termed misclassification only when $n > 10$, however even after numerous trials, did not falter with maximum value of n reaching 5.

INCREMENTAL ACTIVE LEARNING

Process	Computational Time
Inference time	11 ms
Incremental Learning per epoch	~1.7ms
Dropout iteration	~1.2ms

- The ConvNet takes a model size of 266 kB.
- T=10 stochastic dropout iterations (1.2 sec per iteration) were used.
- Can be customized depending on the locality and bus usage characteristics, like periodic per trip update, per day/night update, etc.
- Can be seamlessly integrated with vision-based autonomous vehicles.

Resource-Constrained Federated Learning with Heterogeneous Labels and Models

Gautham Krishna et al., Resource-Constrained Federated Learning with Heterogeneous Labels and Models, AIoT ’20@ACM KDD ‘20

LEARNING FROM MULTIPLE DEVICES ON THE EDGE

Collaborative and Distributed Machine Learning is now possible more than ever to help best utilize the information learnt from multiple IoT devices.

Practical Challenges

Privacy Concerns about sharing sensitive data to the cloud from local user devices	***Low Latency*** between cloud and local devices

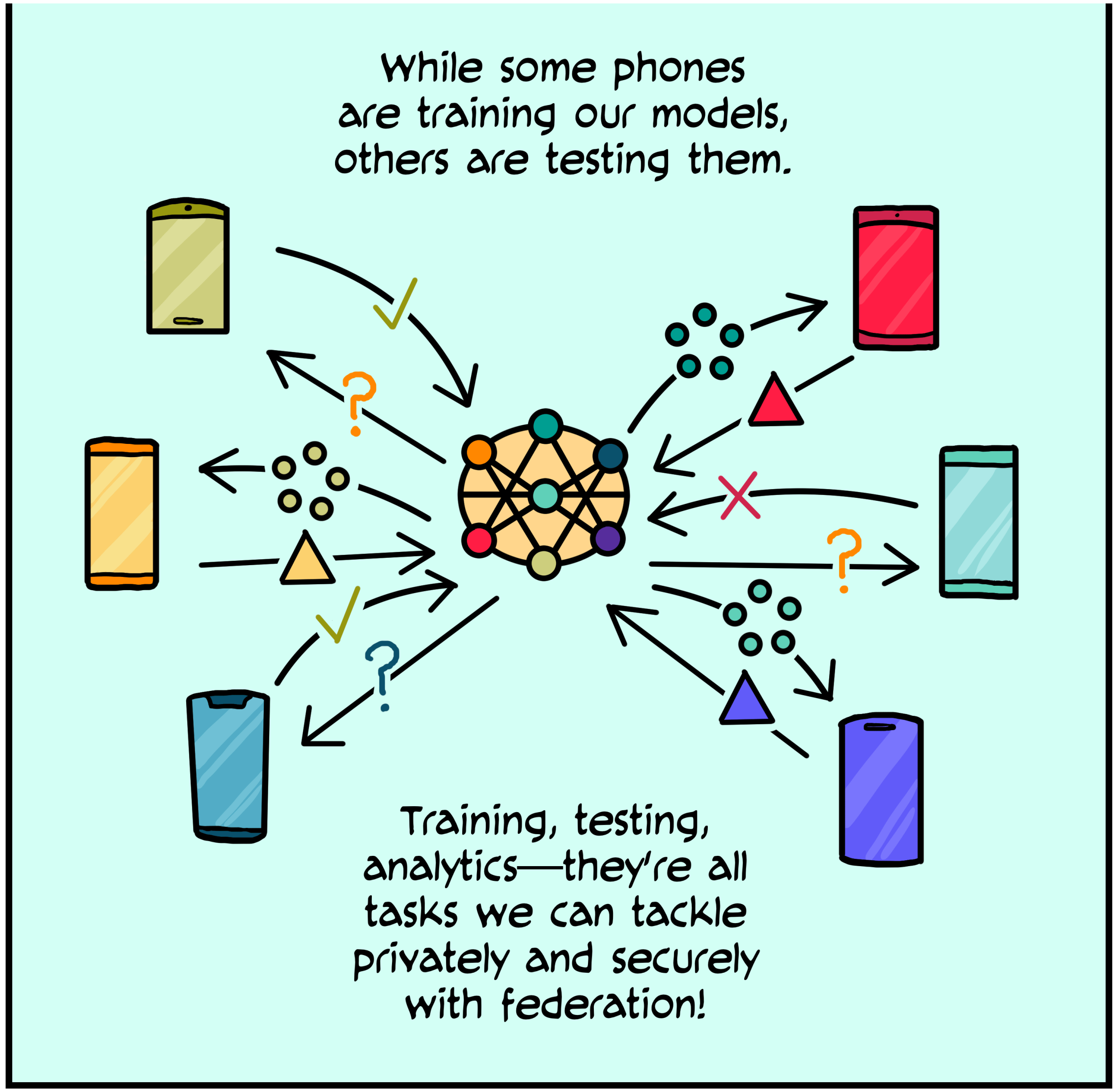
FEDERATED LEARNING

Algorithms are trained across a federation of multiple decentralized devices.

Effectively train a global/centralized model without compromising on sensitive data of various users.

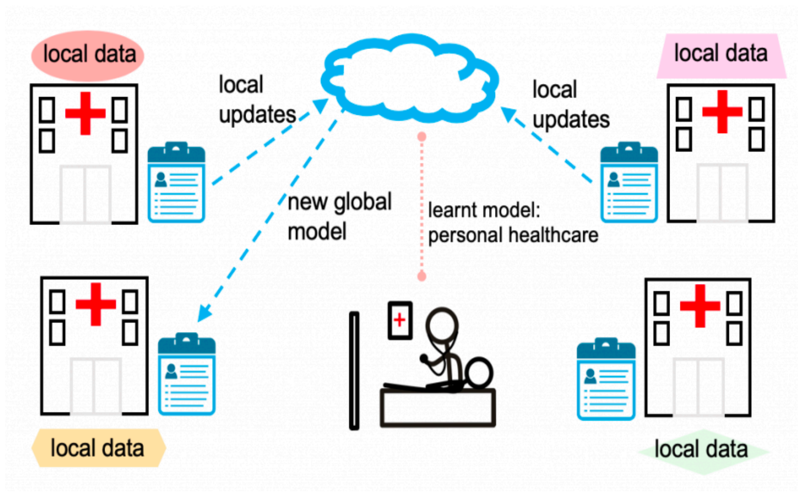
Transfer of model weights and updates from local devices to cloud, rather than conventional sharing of data.

***More Personalization; Minimal Latency; Privacy Preserving*.**


While some phones are training our models, others are testing them.
Training, testing, analytics—they're all tasks we can tackle privately and securely with federation!

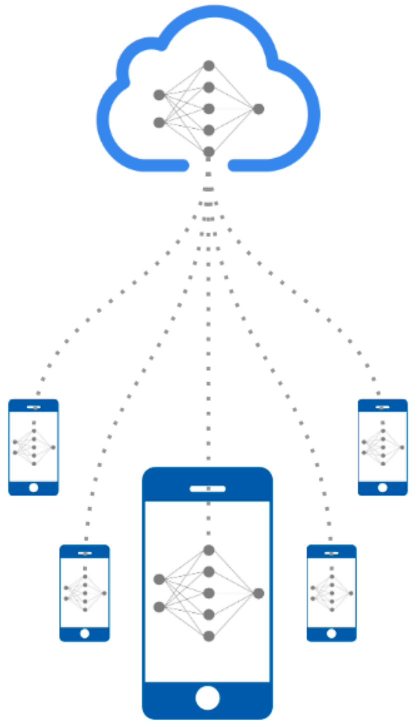
Picture taken from federated.withgoogle.com

HEALTHCARE

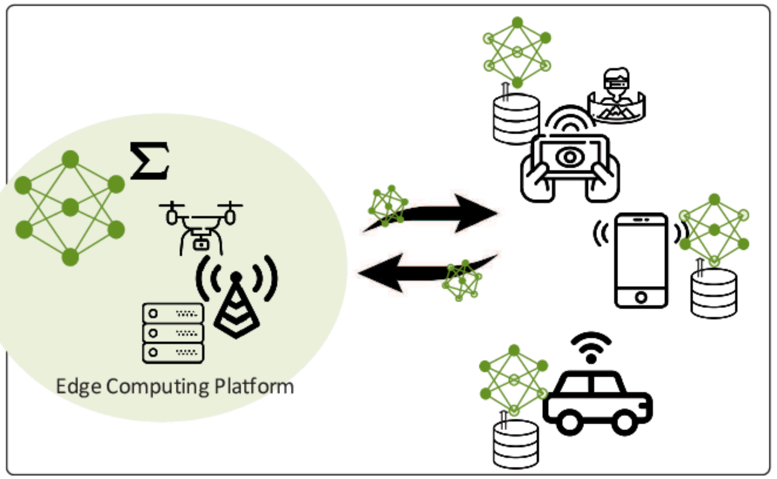


Picture taken from
https://blog.ml.cmu.edu/2019/11/12/federated-learning-challenges-methods-and-future-directions/

IoT ON THE EDGE



WIRELESS/TELECOM

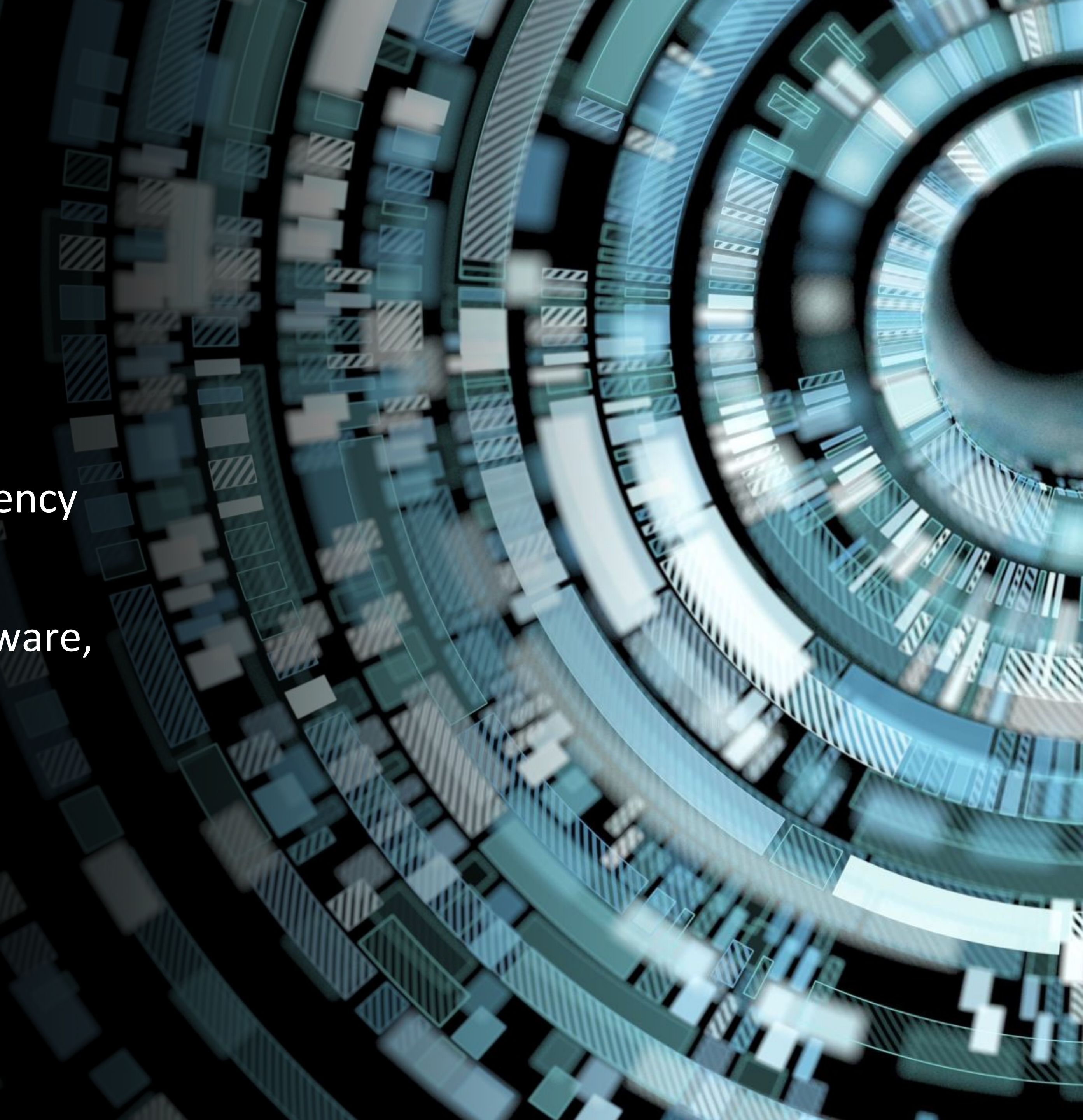


Picture taken from
https://arxiv.org/pdf/1908.06847.pdf

FEDERATED LEARNING APPLICATIONS

PROMINENT CHALLENGES IN RESOURCE-CONSTRAINED FEDERATED LEARNING

- Communication Overheads – Reducing Latency
- Privacy Concerns – Sensitive Data Transfer
- Systems Heterogeneities – Hardware, Software, Network, Power (Resource Constraints)
- Statistical Heterogeneities
  - Non-IIDness
  - Model Heterogeneities



PROMINENT CHALLENGES IN RESOURCE-CONSTRAINED FEDERATED LEARNING

- Communication Overheads – Reducing Latency
- Privacy Concerns – Sensitive Data Transfer
- Systems Heterogeneities – Hardware, Software, Network, Power (Resource Constraints)
- Statistical Heterogeneities
  - Non-IIDness
  - Model Heterogeneities
  - Label Heterogeneities

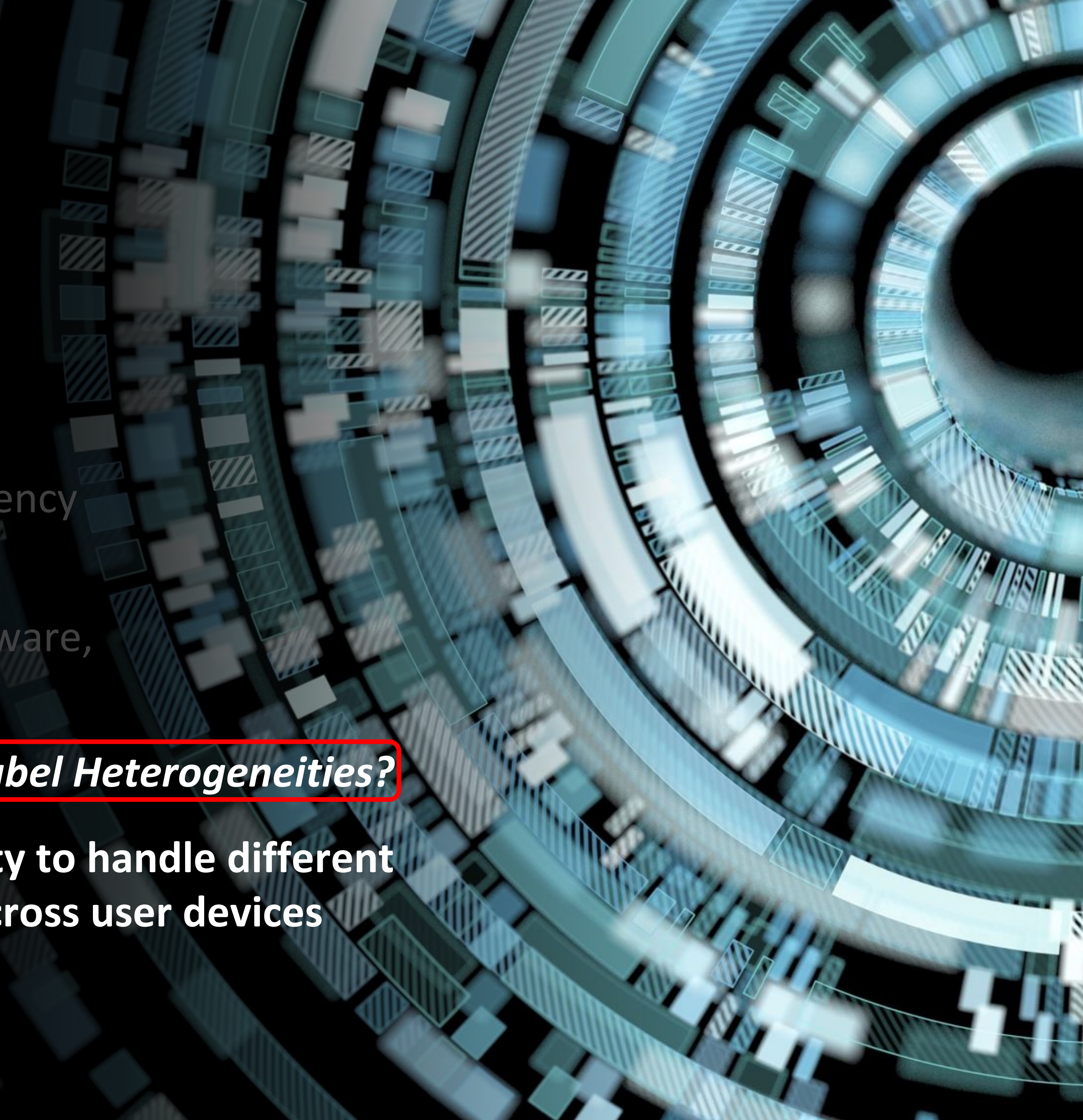
What are *Label Heterogeneities?*

PROMINENT CHALLENGES IN RESOURCE-CONSTRAINED FEDERATED LEARNING

- Communication Overheads – Reducing Latency
- Privacy Concerns – Sensitive Data Transfer
- Systems Heterogeneities – Hardware, Software, Network, Power (Resource Constraints)
- Statistical Heterogeneities
  - Non-IIDness
  - Model Heterogeneities
  - Label Heterogeneities

What are *Label Heterogeneities?*

The flexibility to handle different labels across user devices

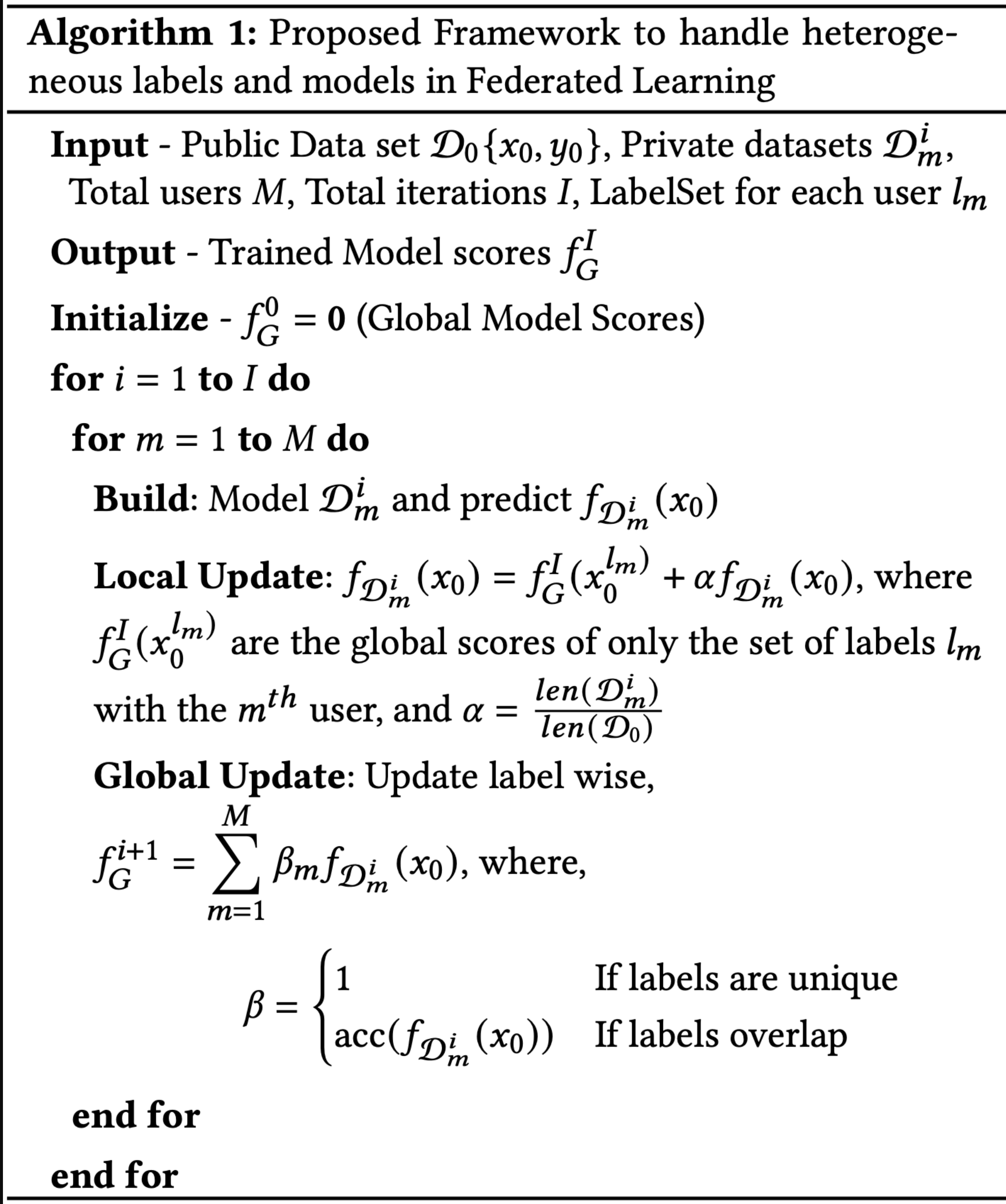


GOALS OF OUR PROPOSED SYSTEM

- A framework to allow ***flexible heterogeneous selection of labels***, thereby leveraging information pertaining to specific classes (with and without label overlap).
- Flexibility in ***preferred local model architectures*** in a federated learning setting, for effective transfer learning between global and local models.
- Empirical demonstration of the framework's ability to handle different data distributions (***statistical heterogeneities and non-IID***) across various user devices.
- Demonstrating the ***feasibility of on-device personalized federated learning***, and resource-friendly; independent of users (*User Adaptability*).

PROPOSED FRAMEWORK

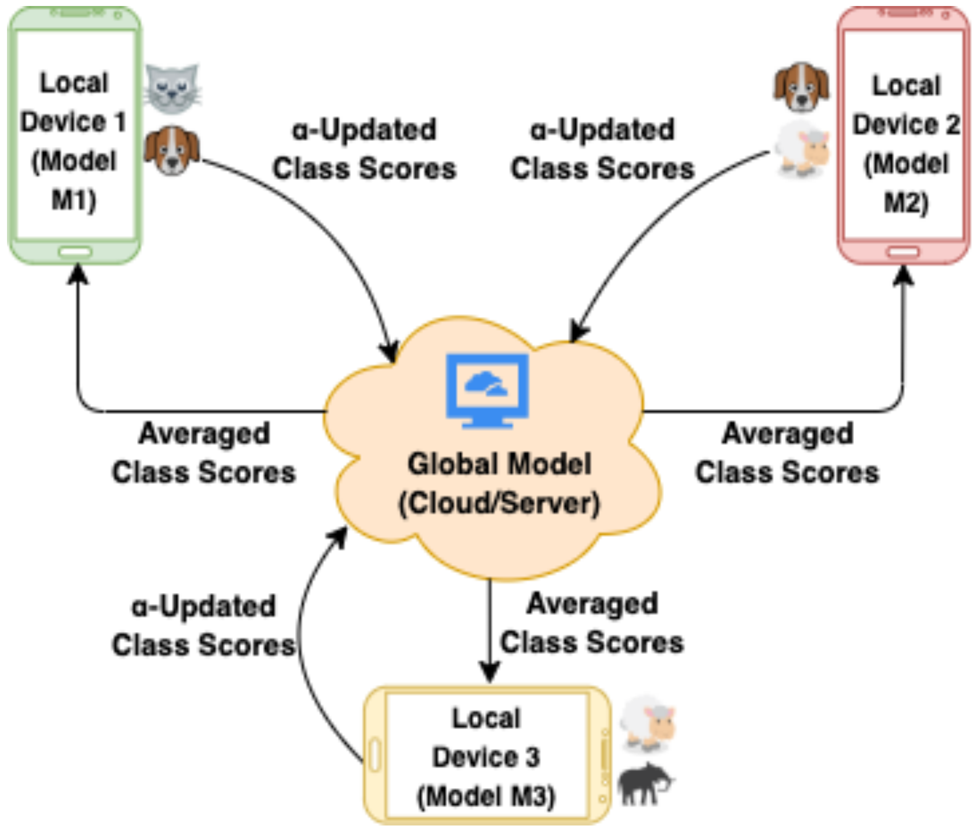
- *Model scores*, instead of model weights are sent to the cloud during every federated learning iteration.
- ***Build:*** We build the model on the incoming data pertaining to each local user at specific iteration.
- ***Local Update:*** Weighted average of scores across different iterations on same user.
  - We propose a weighted α-update, where α is the ratio between the size of current private dataset and the size of public dataset.
  - Governs the contributions of the new and old models.
- ***Global Update:*** Weighted average of scores across all users in same iteration.
  - β parameter governs the weightage given to overlapping labels across users.



Algorithm 1: Proposed Framework to handle heterogeneous labels and models in Federated Learning

```
Input - Public Data set $\mathcal{D}_0\{x_0, y_0\}$, Private datasets $\mathcal{D}_m^i$,
 Total users $M$, Total iterations $I$, LabelSet for each user $l_m$
Output - Trained Model scores $f_G^I$
Initialize - $f_G^0 = \mathbf{0}$ (Global Model Scores)
for $i = 1$ to $I$ do
  for $m = 1$ to $M$ do
    Build: Model $\mathcal{D}_m^i$ and predict $f_{\mathcal{D}_m^i}(x_0)$
    Local Update: $f_{\mathcal{D}_m^i}(x_0) = f_G^I(x_0^{l_m}) + \alpha f_{\mathcal{D}_m^i}(x_0)$, where
    $f_G^I(x_0^{l_m})$ are the global scores of only the set of labels $l_m$
    with the $m^{th}$ user, and $\alpha = \frac{len(\mathcal{D}_m^i)}{len(\mathcal{D}_0)}$
    Global Update: Update label wise,
    $f_G^{i+1} = \sum_{m=1}^{M} \beta_m f_{\mathcal{D}_m^i}(x_0)$, where,
          $\beta = \begin{cases} 1 & \text{If labels are unique} \\ \text{acc}(f_{\mathcal{D}_m^i}(x_0)) & \text{If labels overlap} \end{cases}$
  end for
end for
```

PROPOSED SYSTEM/ ARCHITECTURE



EXPERIMENTAL SETUP

- Animals-10 Dataset.
- 4 labels {Cat, Dog, Sheep, Elephant} simulated for 15 iterations across 3 users.
- D_0 is the public dataset (also test dataset), with 500 images per label – 2000 labels in total.
- Average the model scores on public dataset D_0 from the built models in each iteration.
- Image data across different iterations are split with disparities in both labels and distributions of data (*non-IID*).

	User 1	User 2	User 3	Global User
Model Arch.	2-Layer CNN {16, 32} Softmax Activation	3-Layer CNN {16, 16, 32} ReLU Activation	2-Layer CNN {16, 32} ReLU Activation	—
Labels	{Cat, Dog}	{Dog, Sheep}	{Sheep, Elephant}	{Cat, Dog, Sheep, Elephant}
Images per Iter	{500, 500}	{500, 500}	{500, 500}	{500, 500, 500, 500}

Alessio, C. Animals-10 Dataset, 2019.
https://www.kaggle.com/alessiocorrado99/animals10

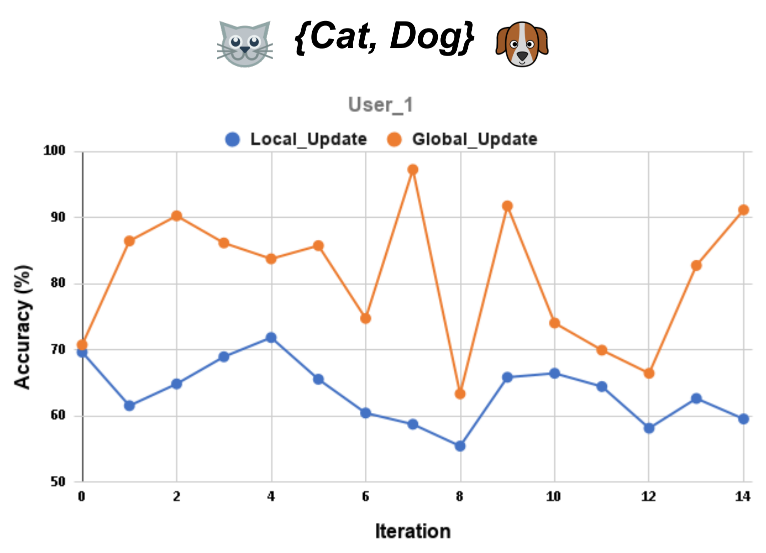
HETEROGENEITY IN MODEL ARCHITECTURES ACROSS ITERATIONS

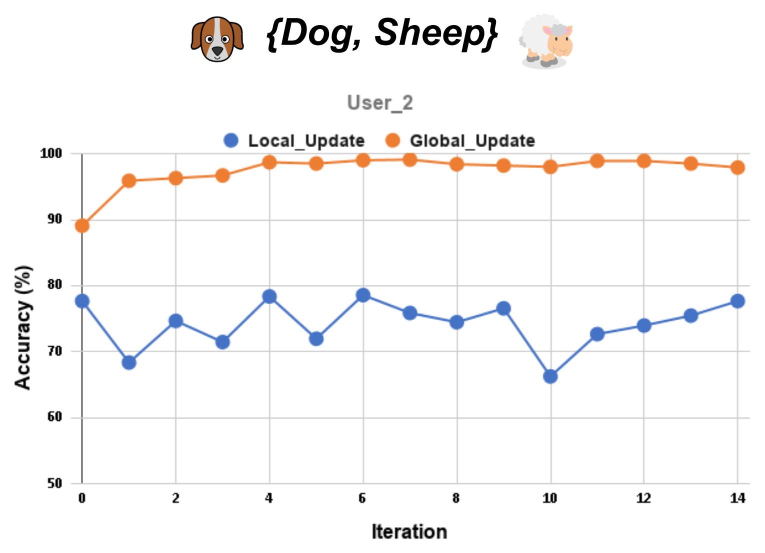
Iterations	New Model Arch.
User 1 Iteration 10	3-Layer ANN (16, 16, 32) ReLU Activation
User 1 Iteration 14	1-Layer CNN (16) Softmax Activation
User 2 Iteration 6	3-Layer CNN (16, 16, 32) Softmax Activation
User 3 Iteration 5	4-Layer CNN (8, 16, 16, 32) Softmax Activation

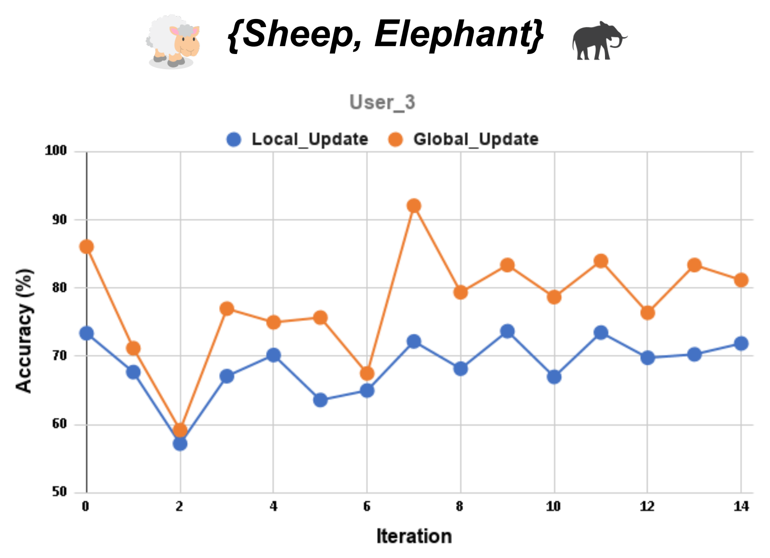
AVERAGE INCREASE IN ACCURACIES ACROSS USERS

- Accuracies of all global updates in each user are deterministically higher than their respective accuracies of local updates.
- Information gain in User 2, maximum overlapped labels; more robust in global updates.
- Overall increase of **~16.7%** across all three users.

	Local Update	Global Update	Accurac y Increase
User 1	63.66	81.02	17.36
User 2	74.3	97.47	**23.17**
User 3	68.72	78.02	9.3
Average	**68.89**	**85.5**	**16.7**



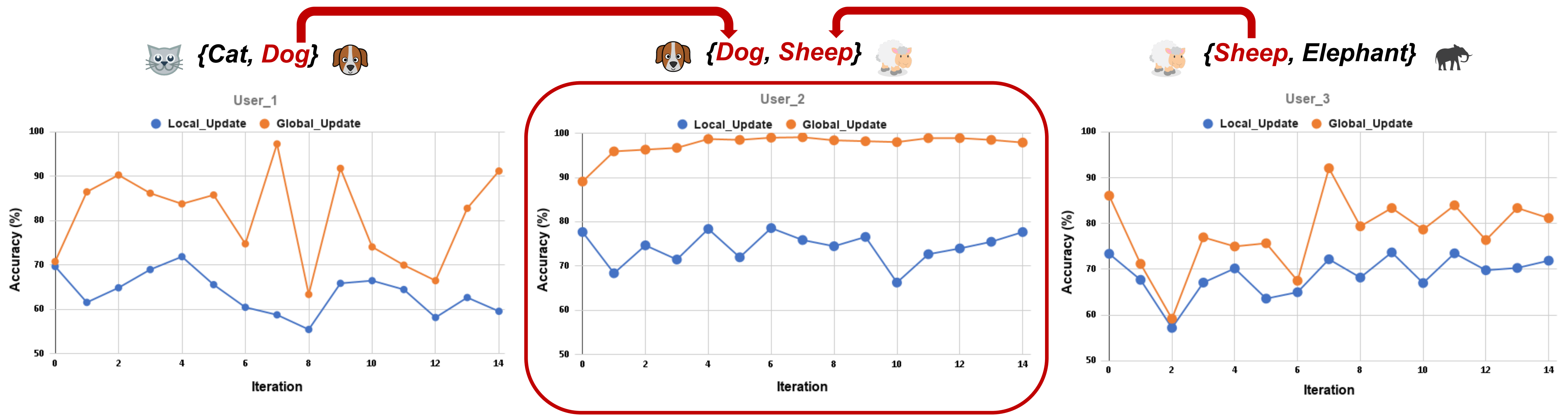




LOCAL MODEL ACCURACY vs ITERATIONS

Local Update signifies the accuracy of each local updated model (after ith iteration) tested on Public Dataset D_0.

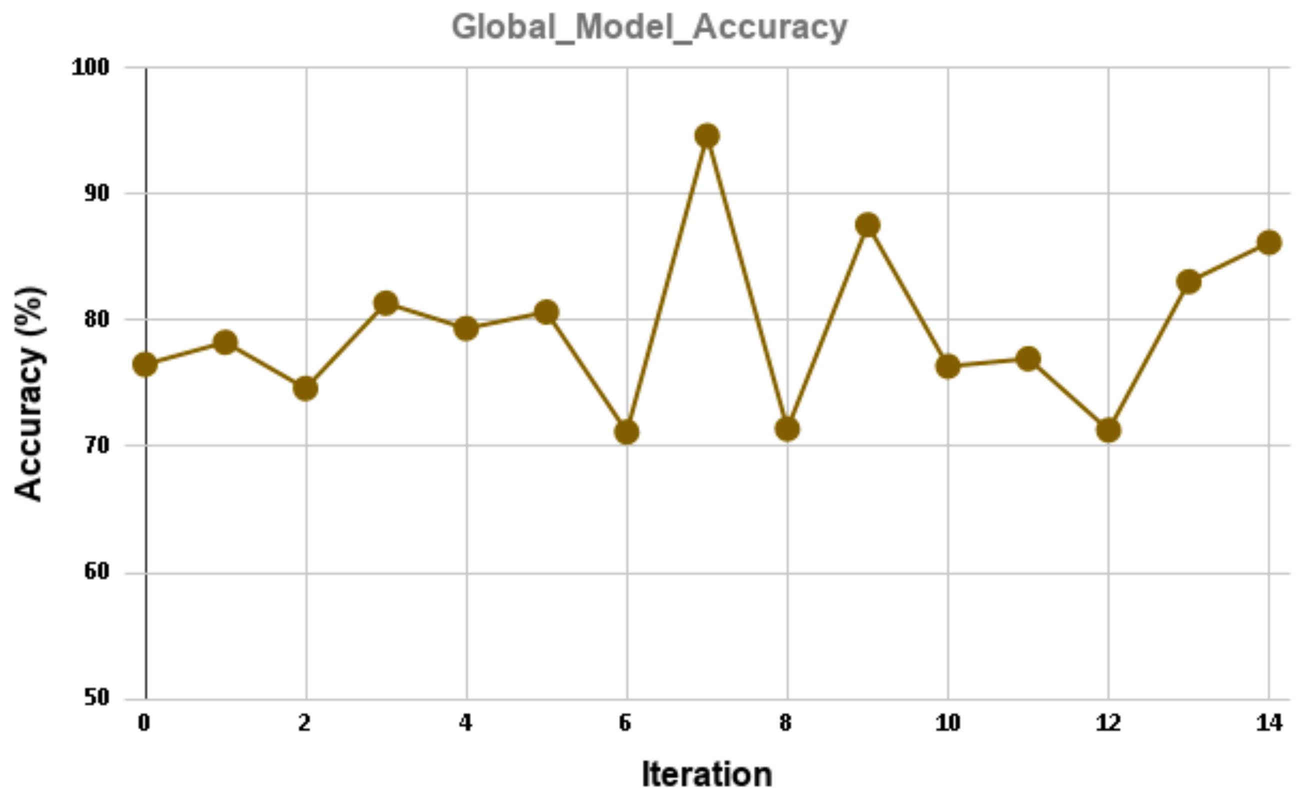
Global Update signifies the accuracy of the corresponding global updated model (after ith iteration) tested on Public Dataset D_0.



LOCAL MODEL ACCURACY vs ITERATIONS

Local Update signifies the accuracy of each local updated model (after i^th iteration) tested on Public Dataset D_0.

Global Update signifies the accuracy of the corresponding global updated model (after i^th iteration) tested on Public Dataset D_0.



FINAL GLOBAL AVERAGE ACCURACIES vs ITERATIONS

ON-DEVICE PERFORMANCE

- On-device performance of our proposed framework is experimented on a Raspberry Pi 2.
- Similar (HW/SW) specifications with that of predominant contemporary IoT/edge/mobile devices.
- Clearly feasible.

Process	Computational Time
Training time per epoch in a FL iteration	1.8 sec
Inference time	15 ms

CONCLUSION

- A unified method with to handle both heterogeneous labels and model architectures in Federated Learning setting.
- Both global and local updates require computation of global model accuracy and are weighted based on it (α and β updates).
- Overlapping labels are found to make our framework robust, and also helps in effective accuracy increase.
- Exhibit on-device feasibility of federated learning and inference.

TAKEAWAYS

- Ubiquitous Computing is the future, and we are all a part of it, whether we know it or not.
- Enabling intelligence into mobile and edge devices is exciting, personalization at its best.
- Lots of interesting challenges in marrying UbiComp and ML/DL.
- There is a huge intersection of theory and applied ML in this space. Theoretically guaranteed and resource-aware/resource-constrained algorithms are needed for application on such devices.
- Be resource-aware! Carbon footprint is extremely low, attitude change towards looking at ML problems from a resource-aware practical standpoint helps.

Contact:

Gautham Krishna Gudur
gauthamkrishna.gudur@gmail.com

Let’s Connect!

THANK YOU!

QUESTIONS?