Beyond Thresholds: A General Approach to Sensor Selection for Practical Deep Learning-based HAR

Geffen Cooper
The University of Texas at Austin
Email: geffen@utexas.edu

Radu Marculescu
The University of Texas at Austin
radum@utexas.edu

***Abstract*—In deep learning (DL) based human activity recognition (HAR), sensor selection seeks to balance prediction accuracy and sensor utilization (how often a sensor is used). With advances in on-device inference, sensors have become tightly integrated with DL, often restricting access to the underlying model used. Given only sensor predictions, how can we derive a selection policy which does efficient classification while maximizing accuracy? We propose a cascaded inference approach which, given the prediction of any one sensor, determines whether to query all other sensors. Typically, cascades use a sequence of classifiers which terminate once the confidence of a classifier exceeds a threshold. However, a threshold-based policy for sensor selection may be suboptimal; we define a more general class of policies which can surpass the threshold. We extend to settings where little or no *labeled* data is available for tuning the policy. Our analysis is validated on three HAR datasets by improving upon the F1-score of a threshold policy across several utilization budgets. Overall, our work enables practical analytics for HAR by relaxing the requirement of labeled data for sensor selection and reducing sensor utilization to directly extend a sensor system's lifetime.**

I. Introduction

Human activity recognition (HAR) has immense potential for improving daily life by enabling applications such as fall detection [1], rehabilitation monitoring [2], and health tracking [3]. A key driver of such applications are wearable inertial measurement units (IMUs) which can capture motion data of multiple body parts using accelerometers and gyroscopes. These affordable low-power sensors, along with advancements in deep learning (DL) for time series signals, have led to a surge in research within the field of IMU-based HAR [3].

Despite this progress, implementing practical analytics for IMU-based HAR systems using DL remains challenging due to the limited size of labeled datasets for HAR and data heterogeneity across users and environments which limit the generalizability of DL approaches. Thus, it is common to use *multiple* wearable IMUs across the body to obtain more information and build better performing prediction models.

DL-based works for *multi*-sensor HAR [4]–[6] tend to focus on architecture design [7]–[10] or sensor fusion [11]–[14] and assume that synchronous data from multiple sensors is available for analysis on a central device such as a phone. However, this disregards the resource requirements for streaming data from wearable devices which have limited battery life. Processing data from several sources also burdens the mobile device which already needs to communicate with multiple sensors. Hence, such approaches can limit the practicality and sustainability of DL for wearable HAR.

In light of this, many works [15]–[20] propose to activate a subset of sensors (or features) at a given time via *sensor selection* (or *feature selection*) to conserve energy while maintaining a certain level of accuracy. Some works [19] condition their selection policy on the input data while other works [17] only use sensor predictions as input to the selection policy. We work in the latter setting but further assume that each sensor uses a pretrained ML model which cannot be modified.

While full access to the raw sensor data and ML classifiers provides more information and flexibility when building the sensor selection policy, we may not have access to this information in practical settings. With the emergence of *machine learning sensors* [21], sensors and ML models have become tightly coupled, and access to the raw sensor data and ML model can be restricted. In our setting, visualized in Fig. 1a, only the model output is available to the sensor selection policy. While these constraints may limit how effective a selection policy can be, we believe this setting is underexplored yet important as off-the-shelf ML sensors become more common.

With such limited information, how can we derive a sensor selection policy to do classification more efficiently while maximizing accuracy? Given fixed prediction models, we focus on sensor selection through *cascaded inference* where the prediction confidence of any sensor determines whether to query and ensemble the predictions of all other sensors. As shown in Fig. 2, an input $\mathbf{x}$ is passed into a local model f_k of a sensor s_k, which outputs a prediction for the current activity along with a confidence score. This score is passed into a decision-making policy π which either uses the current prediction or queries additional sensors. Querying other sensors typically results in a better prediction, but increases the utilization of each sensor. Thus, we define the policy π to maximize the collective accuracy of the sensors, while keeping the utilization of all sensors under a prespecified budget.

Many works that leverage cascaded models [22]–[24] use a *threshold policy*, where a second model is queried only if the first model's confidence does not exceed a threshold. However, a threshold may not adequately describe the relationship between the first model's confidence and the second model's accuracy. For example, the second model may have similar accuracy to the first model when the first model has a confidence score in the interval $[0, 0.25]$ or $[0.75, 1.0]$. Thus, there is no

979-8-3503-7025-6/24/$31.00 ©2024 IEEE
DOI 10.1109/IoTDI61053.2024.00005

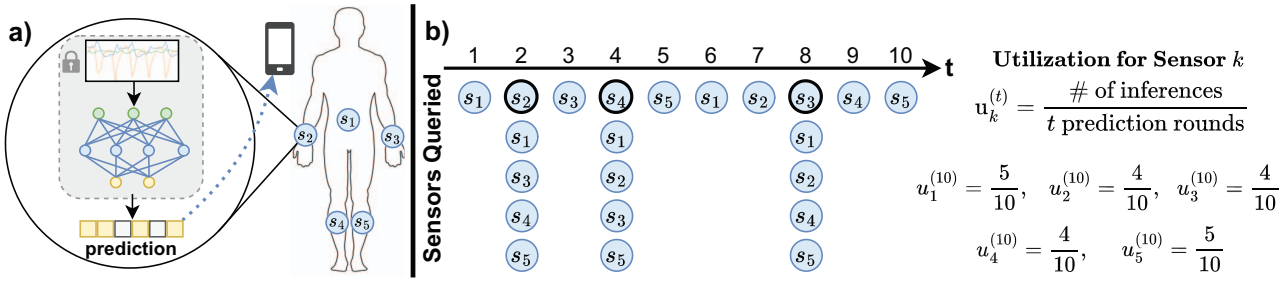

Fig. 1. **a)** *ML sensors* processes data locally and restrict access to the raw data and ML model. The prediction is sent to a mobile device where the selection policy is executed. **b)** An example of sensor utilization for $K = 5$ sensors over $t = 10$ steps. Sensors with dark circles at steps 2, 4, and 8 represent their policies triggering inference of the other sensors to ensemble their predictions. Note, this sequence is shown for an example calculation and is not realistic.

benefit to querying the second model when the first model has a confidence within those intervals. However, the second model may have much higher accuracy than the first model when the first model has a confidence score in the interval $[0.25, 0.75]$; here, querying the second model does provide an expected gain in accuracy. This raises the following questions: *Do such unstructured relationships between confidence and accuracy of successive models arise in practice? Can a policy which models this relationship be realistically implemented? What practical benefits does such a policy provide for sensor selection?* **Our contributions are as follows**:

1) We propose a general class of policies, for which the threshold is a special case, to tractably analyze the sensor selection task in the setting of cascaded inference.
2) We show under what conditions the threshold policy is optimal *within the defined class of policies* and when it can be improved upon. When a threshold is suboptimal, our policy directly extends the sensor system's lifetime.
3) We validate our analysis on three HAR datasets and extend our results to a practical setting where little to no labeled data is available for tuning the policy. *Without labels*, our policy is competitive with a threshold policy *tuned with labeled data* for multiple utilization budgets.

The rest of the paper is organized as follows. Section II describes how our work differs from the related works. In section III, we comprehensively describe our problem formulation and methodology for sensor selection. Section IV describes the implementation of our sensor selection policy and the full experimental procedure. In section V we analyze our results and discuss their relevance for practical HAR. Finally, section VI summarizes our work and discusses future directions.

II. Related Work and Novel Contribution

A. Sensor Selection

Sensor selection reduces the energy consumption of wearable HAR systems by activating a subset of the available sensors; the goal is to do so while maintaining a suitable accuracy. Our work relates to *dynamic* sensor and feature selection [15]–[19] which determine which sensors or features to use at prediction time. This is in contrast to *static* or *fixed* sensor selection which determines which sensors are the most

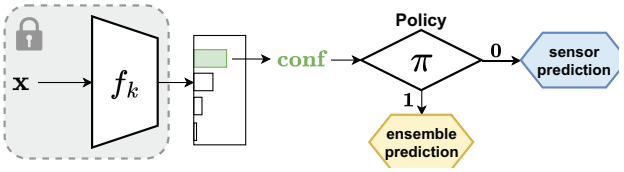

Fig. 2. Cascade perspective for sensor selection. For every input $\mathbf{x}$, the prediction confidence is passed to a policy π which determines whether to (0) use the prediction, or (1) query and ensemble all other sensor predictions. The policy cannot access the raw input or model parameters. A threshold policy has the form, $\pi(\text{conf}) = \text{conf} < \delta$ where δ is the threshold.

important at training time, and proceeds to use all selected sensors at runtime [25]. The authors of [15] use random forests to estimate the importance of pose-based features to create feature subsets. A model is trained on each subset and the best one is dynamically selected during prediction. The authors of [17] seek to leverage the predictability of future activities and weight the sensors based on their activity-sensor dependency. More recently, the work of [19] exploits activity sequences as well as instance-level information to minimize the number of sensors queried during inference using a Markov Decision Process. Our work approaches the sensor selection task from a different perspective; we examine the scenario of *one level cascades* where the prediction of one sensor determines if additional sensors should be queried at runtime. Beyond sensor selection, we provide insight into what factors influence the performance of a policy in the cascaded inference setting.

B. Cascaded Inference

Cascades enable DL models to adapt to different inputs using a sequence of classifiers; cascades and *selective inference* have been studied extensively [22], [23], [26]–[33]. Each classifier in a cascade relies on a *policy* to determine whether to use the current prediction, or to continue execution through the next model. We focus on the *threshold* policy which uses the current prediction if its *confidence* (max softmax score) exceeds a given threshold. While many works use a threshold policy for cascaded inference [23], [24], [34], only a few [28], [35] study their limitations. However, [28], [35] focus on centralized computer vision models where cascaded

models and their inputs are located on the same device. We work in the context of multi-sensor HAR where models and data are distributed across devices. Thus, our approach, which only requires access to model outputs, is better suited to this distributed HAR setting. Further, our approach improve upon the threshold policy by generalizing to multiple thresholds.

C. Model Calibration

Model calibration refers to the alignment of a model's output class probabilities (confidence scores) with the true likelihood of correctness [36]. A miscalibrated model outputs confidence scores that do not reflect the true probability of a prediction being correct. Thus, it seems that a miscalibrated model cannot be effectively used for a downstream task which relies on its confidence scores. The authors of [35] address this by integrating calibration into the loss function. They seek to train a cascade of models such that a classifier's confidence score not only reflects whether the current classifier will be correct, but also if the next classifier in the cascade will be correct. While optimizing cascaded models to output 'informative' confidence scores is appealing, it requires access to the ML models. Our setting considers pretrained models whose parameters are restricted; thus we extract any meaningful relationship between cascades solely from the confidence scores. Further, we show that a calibrated sensor model is not sufficient or necessary for the threshold policy to perform well.

III. Problem Formulation and Analysis

Now we comprehensively describe our methodology. For convenience, important notation is shown in Table I.

A. Sensor Utilization

Overview: We use sensor *utilization* as a proxy for the energy efficiency of a sensor system. Consider a set of K identical sensor nodes, $S = \{s_1, s_2, \ldots, s_K\}$, for wearable HAR[1]. Each node s_k periodically buffers data into a sliding window w_k of length l from an IMU, but otherwise remains in a low-power state. At each time step t, a subset $S_{active} \subseteq S$ of nodes will pass the current window $w_k^{(t)}$ into their neural network for inference and send the result to a central device.

Assume inference dominates energy consumption so that the lifetime of a sensor is defined by I, the number of inferences it can make before running out of battery. Further assume that at least one node must do inference at each step t. For simplicity, we define the system lifetime as the minimum lifetime among all sensors (if any sensor dies, the entire system dies).

Definition: The *utilization* of a sensor s_k after t steps is $u_k^{(t)} = \frac{a_k}{t}$ where a_k is the number of times sensor s_k was used for inference (see Fig. 1b). To maintain at least one active sensor at every step we must have $\sum_{k=1}^{K} u_k^{(t)} \geq 1$. Intuitively, utilization provides a knob to control a sensor's lifetime with higher utilization corresponding to a shorter lifetime.

We assume an application defines a *global* utilization *budget* u such that $u_k^{(t)} \leq u$ for all s_k where u determines the trade-off between the system lifetime and classification accuracy. A higher u corresponds to a shorter lifetime and higher accuracy. For a given u, we *ideally* have $u_1^{(t)} = u_2^{(t)} = \cdots = u_K^{(t)} = u$ because 'under-utilizing' a sensor (having $u_k^{(t)} < u$) does not extend the system lifetime since it is bounded by the most utilized sensor. Then, the global budget must satisfy $u \in [\frac{1}{K}, 1]$. If $u < \frac{1}{K}$ this violates $\sum_{k=1}^{K} u_k^{(t)} \geq 1$. Further, $u \leq 1$ since a sensor is active at most every round.

In practice, we cannot ensure $u_1^{(t)} = \cdots = u_K^{(t)} = u$ for each t. To enforce both $\sum_{k=1}^{K} u_k^{(t)} \geq 1$ and $u_k^{(t)} \leq u$, we query each sensor in a round-robin fashion at each t as shown in Fig. 1b to maintain the minimum utilization. Then, if the policy for a given sensor signals to call the ensemble, we check if this will cause any sensor's utilization $u_k^{(t)}$ to exceed the budget u. If it does we do not call the ensemble. Thus, we only consider budget violations as a result of calling the ensemble.

Implementation Considerations: In our setting we only control when a sensor does inference which fixes the cost of data processing from the perspective of the policy. For communication costs, BLE (Bluetooth Low Energy) based sensors would maintain a synchronized periodic connection with a central device based on a *connection interval* which we assume occurs more often than each time step t. Each time step, the central device queries the next sensor to do inference for window $w_k^{(t)}$, which returns a confidence score. If the policy, which executes on the central device, signals to call the ensemble, the central device will query all sensors $s_{k'}$, for $k' \neq k$, to do inference for the buffered window $w_{k'}^{(t)}$ on the next *connection event* which occurs before step $t+1$. Given that only the model prediction and policy result need to be communicated, we assume that these BLE transmissions are negligible compared to the cost of inference.

B. Confidence Based Sensor Selection Policy

Utilization Budget: Our goal is to maximize the prediction accuracy for a given utilization budget u. We consider the scenario where the confidence of any *one* sensor determines whether to do inference on *all* other sensors. First, we derive $\alpha_e \in [0, 1]$, the fraction of time the ensemble can be called while maintaining the budget u. By default, one sensor does inference each step so $u \geq \frac{1}{K}$. If we call the ensemble, the remaining $K-1$ sensors do inference. Thus, we express the budget as $u \geq (\alpha_e) \cdot \frac{K-1}{K} + \frac{1}{K}$ and $\alpha_e \leq (u - \frac{1}{K})/(\frac{K-1}{K})$. To interpret this expression, consider the two extremes. If $u = \frac{1}{K}$, then $\alpha_e = 0$ because there is no budget to use the ensemble. In contrast, if $u = 1$, then $\alpha_e = 1$ because there is effectively no budget constraint and we can call the ensemble every round.

Sensor Selection: Define each sensor's neural network as a function f_k which outputs softmax scores across C classes so that for input $\mathbf{x}$ we have $f_k(\mathbf{x}) \in [0, 1]^C$ and $\sum_c f_k(\mathbf{x})_c = 1$. Define a confidence-based policy for sensor s_k as a function $\pi_k : (0, 1] \to \{0, 1\}$; it takes as input the *confidence* (conf), $\max_c\{f_k(\mathbf{x})\}$, of sensor s_k and outputs a binary decision: `1` to call the ensemble or `0` to use $\text{argmax}_c\{f_k(\mathbf{x})\}$ as the prediction. As shown in Fig. 3, the policy π_k can be expressed

[1]Example of such sensors: https://www.movesense.com/movesense-active/

as a piecewise function of J disjoint intervals which specify which decision a confidence score corresponds to:

$$\pi_k(\text{conf}) = \begin{cases} 1, & \exists j \ s.t. \ \text{conf} \in (a_j, b_j], \ j = 1, \ldots, J \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

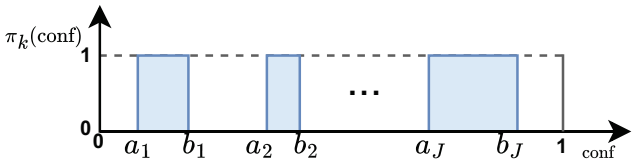

Fig. 3. Visualization of confidence-based policy based on the form in (1)

TABLE I
NOTATION

Notation	Description
$S = \{s_1, \ldots, s_K\}$	A set of K sensors, indexed by k
t	Time index when prediction occurs
$w_k^{(t)}$	Data window for k^{th} sensor at time step t
a_k	Total # of times the k^{th} sensor did inference
$u_k^{(t)} = \frac{a_k}{t}$	Utilization of k^{th} sensor at time step t
$u \in [\frac{1}{K}, 1]$	Utilization budget of a sensor system
α_e	Fraction of time the ensemble can be called
f_k	Classifier for the k^{th} sensor
π_k	Policy of the k^{th} sensor
M	# of bins used to discretize confidence distribution
$(a_m, b_m]$	Interval of bin m
$\mathbf{p}_k \in \mathbb{R}^M$	Probability that k^{th} sensor's prediction is correct given its confidence score (bin)
$\mathbf{p}_{e\|k} \in \mathbb{R}^M$	Probability that ensemble's prediction is correct given k^{th} sensor's confidence score (bin)
a_e	# of bins that can be allocated to the ensemble
$\mathbf{x}_k \in \mathbb{R}^M$	Optimized policy of the k^{th} sensor
$\mathbf{acc}_k \in \mathbb{R}^M$	Empircal estimate of $\mathbf{p}_k$
$\mathbf{acc}_{e\|k} \in \mathbb{R}^M$	Empircal estimate of $\mathbf{p}_{e\|k}$
I	Lifespan of sensor in terms of # of inferences

This form of the policy is intractable since we need to optimize over the number of intervals, and the width of each interval. Further, such a flexible class of policies can easily overfit by setting the intervals to memorize exact confidence values. Therefore, we add explicit structure to simplify the policy class. Consider f_k, the model for a single sensor. Over a range of inputs for a given user, this model will output a distribution of confidence values. First, we discretize the confidence distribution of f_k into M bins. The width of each bin is set such that the number of a predictions in each bin is equal. For bin $m \in \{1, \ldots, M\}$, we can estimate $\mathbf{p}_k = P(\hat{y}_k = y \,|\, \text{conf}_k \in (a_m, b_m])$, the probability that a prediction in bin m will be correct, using the accuracy of each bin. Here, $\hat{y}_k, y$ are the class prediction (for s_k) and label for a given sample, and $\text{conf}_k \in (a_m, b_m]$ means the confidence score from sensor s_k falls into bin m. This empirical estimate of $\mathbf{p}_k$ creates a *reliability diagram* [36]–[38] which describes a model's accuracy as a function of its confidence.

Reliability Diagram: Define the reliability diagram for the model f_k as $\mathbf{acc}_k \in \mathbb{R}^M$. We also define a conditional reliability diagram $\mathbf{acc}_{e|k} \in \mathbb{R}^M$ as the ensemble model's accuracy as a function of the confidence of sensor s_k. This is the empirical estimate of $\mathbf{p}_{e|k} = P(\hat{y}_e = y \,|\, \text{conf}_k \in (a_m, b_m])$, the probability that the *ensemble's* prediction will be correct given the *sensor* confidence. Intuitively, $\mathbf{acc}_k$ and $\mathbf{acc}_{e|k}$ tell us how the sensor model compares to the ensemble in terms of accuracy for each confidence bin. With this information, the form of the policy reduces to a binary vector $\mathbf{x}_k \in \{0, 1\}^M$ that says whether to allocate predictions from each confidence bin to the ensemble. Specifically, we map conf_k, the confidence of sensor s_k, to bin m; the policy returns the m^{th} entry of $\mathbf{x}_k$ which will be `1` if the ensemble should be called and `0` otherwise.

C. Optimizing the Policy

Using the formulation in section III-B, we can determine the policy which will maximize the probability of a correct prediction for a given utilization budget u. Suppose we know $\mathbf{p}_k = P(\hat{y}_k = y \,|\, \text{conf}_k \in (a_m, b_m])$ and $\mathbf{p}_{e|k} = P(\hat{y}_e = y \,|\, \text{conf}_k \in (a_m, b_m])$, the **true** probability that the sensor s_k and ensemble are correct given the confidence of sensor s_k.

Optimization Problem: We first derive the discrete *bin budget* a_e from the continuous budget u. First, map the budget u to α_e, the fraction of predictions which can be allocated to the ensemble. Given α_e, let $a_e = \lfloor \frac{\alpha_e}{1/M} \rfloor$ where a_e is the number of bins that can be allocated to the ensemble. For example, if $\alpha_e = 0.25$ and $M = 20$, then $a_e = 5$ (allocate at most 5 bins to the ensemble where each contains $\frac{1}{M}$ fraction of the predictions). M can be set to match the desired budget fidelity in terms of α_e; with $M = 10$ we can set $\alpha_e = 0.0, 0.1, 0.2, \ldots, 1.0$. Now, to maximize the probability of a correct prediction under the budget a_e, we solve the following optimization problem *for each sensor* s_k:

$$\begin{aligned} \underset{\mathbf{x}_k}{\text{maximize}} \quad & \frac{1}{M}\left(\mathbf{p}_k^\top(\mathbf{1} - \mathbf{x}_k) + \mathbf{p}_{e|k}^\top \mathbf{x}_k\right) \\ \text{subject to} \quad & \mathbf{x}_k^\top \mathbf{1} \leq a_e \end{aligned} \quad (2)$$

where $\mathbf{p}_k \in \mathbb{R}^M$, $\mathbf{p}_{e|k} \in \mathbb{R}^M$, $\mathbf{x}_k \in \{0, 1\}^M$, and M is the number of bins. $\mathbf{x}_k$ is the optimization variable; it is a binary vector that says whether to allocate predictions from each bin to the ensemble. The objective function is the probability of a prediction being correct; the factor of $\frac{1}{M}$ is the probability of landing in each confidence region (bin). This factor is omitted in the remaining analysis since it is a constant. The first term in the objective function is the contribution of the sensor and the second term is the contribution of the ensemble. The constraint ensures we do not exceed the utilization budget. If we expand the objective function we get:

$$\mathbf{p}_k^\top \mathbf{1} - \mathbf{p}_k^\top \mathbf{x}_k + \mathbf{p}_{e|k}^\top \mathbf{x}_k \quad (3)$$

$\mathbf{p}_k^\top \mathbf{1}$ is a constant so it can be taken out of the objective:

$$\begin{aligned} & -\mathbf{p}_k^\top \mathbf{x}_k + \mathbf{p}_{e|k}^\top \mathbf{x}_k \\ = & (\mathbf{p}_{e|k} - \mathbf{p}_k)^\top \mathbf{x}_k = \mathbf{p}_{\text{diff}}^\top \mathbf{x}_k \end{aligned} \quad (4)$$

The optimization problem has reduced to

$$\begin{aligned} \underset{\mathbf{x}_k}{\text{maximize}} \quad & \mathbf{p}_{\text{diff}}^\top \mathbf{x}_k \\ \text{subject to} \quad & \mathbf{x}_k^\top \mathbf{1} \leq a_e \end{aligned} \quad (5)$$

where $\mathbf{p}_{\text{diff}}$ represents how much more (or less) probable the ensemble is to be correct compared to sensor s_k for each bin.

Analysis: This objective will be maximized when $\mathbf{x}_k$ selects the a_e largest positive entries of $\mathbf{p}_{\text{diff}}$. If there are negative entries, this corresponds to the sensor having better accuracy than the ensemble for a certain bin. In these cases, we do not select the ensemble even if we have not used up the budget a_e. Overall, the optimal $\mathbf{x}_k$ (in terms of maximizing the objective) for sensor s_k is set by: **(1)** defining $\mathbf{p}_{\text{diff}} = \mathbf{p}_{e|k} - \mathbf{p}_k$, **(2)** then the $\mathbf{x}_k$ which has 1's in the a_e largest *positive* entries of $\mathbf{p}_{\text{diff}}$ is optimal. Intuitively, this allocates the bins with the largest positive accuracy margin (over sensor s_k) to the ensemble.

Discussion: Now recognize that a threshold is a special case where the a_e largest entries of $\mathbf{p}_{\text{diff}}$ are adjacent; specifically, there is a single separation point (i.e., a confidence threshold) where bins on one side are zeros and bins on the other side are ones. We can increase the number of bins M to match the fidelity of the threshold policy. For example, if $M = 100$ then we can represent all threshold policies which consider the $n \in \{0.0, 0.01, \ldots, 0.99, 1.0\}$ most confident predictions ($n = 0.75$ corresponds to a threshold which allocates the 25 percent of sensor predictions with the lowest confidence to the ensemble). Thus, our policy class is more general and should perform at least as well as the threshold policy. In practice, this is challenging since we can only empirically estimate $\mathbf{p}_{\text{diff}}$ and can't set M too high to avoid overfitting. Regardless, our empirical results in section V show consistent improvement over the threshold policy even when using $M = 10$ bins.

Implications: In theory, a threshold policy will match the optimal $\mathbf{x}_k$ in our policy class when $\mathbf{p}_{\text{diff}}$ has a monotonic structure because the a_e largest entries of $\mathbf{p}_{\text{diff}}$ will be adjacent. If the sensor model f_k is calibrated, $\mathbf{p}_k$ will be monotonically increasing. However, this does not imply that $\mathbf{p}_{\text{diff}}$ will be monotonic since it relies on $\mathbf{p}_{e|k}$ as well. In other words, a calibrated sensor model is not sufficient or necessary for a threshold policy to be successful. We observe that poorly calibrated sensor models with non-monotonically increasing reliability diagrams can still perform well with a threshold policy. Thus, the sub-optimality of the threshold is not due to miscalibration of the sensor model, but a misalignment between the confidence of the sensor and the accuracy of the ensemble. Our policy class overcomes this by generalizing to multiple thresholds represented by confidence regions. When the relationship between the sensor confidence and ensemble accuracy is unstructured (i.e. $\mathbf{p}_{\text{diff}}$ is not monotonic), we improve over the threshold policy by avoiding confidence regions for which the ensemble provides no accuracy gain. Note that monotonicity of $\mathbf{p}_{\text{diff}}$ is not necessary, but sufficient for a threshold to perform well. See Figs. 7, 8, 9 for examples.

D. Semi-supervised and Unsupervised Cases

A limitation of confidence based policies is that they require labeled data to find the best threshold, or in our case the best bin allocation. This is problematic if there are few labeled samples from the target user. Thus, we extend our methodology to cases where little to no labeled data is available.

Unsupervised Scenario: Suppose no labeled data for tuning the policy is available. In this setting, we cannot to measure the accuracy of the sensor model or the ensemble. Thus, assume: $\mathbf{p}_{e|k} \geq \mathbf{p}_k$, i.e. we assume the probability of the ensemble being correct is always higher than the sensor being correct, given the sensor confidence. Under this assumption, we use the *agreement* of the sensor model with the ensemble as a notion of accuracy because we don't know $\mathbf{p}_{e|k}$ either. Specifically, let $\mathbf{p}_{e|k} = \mathbf{1}$ which equally weights each bin of $\mathbf{p}_{e|k}$ and ensures $\mathbf{p}_{e|k} \geq \mathbf{p}_k$. Then replace $\mathbf{p}_k$ with $\mathbf{p}_{\text{agree}} = P(\hat{y}_k = \hat{y}_e \,|\, conf_k \in (a_m, b_m])$, the probability that the sensor agrees with the ensemble for a given bin. Substituting these into the simplified objective function from expression (4) we have:

$$(\mathbf{p}_{e|k} - \mathbf{p}_k)^\top \mathbf{x}_k = (\mathbf{1} - \mathbf{p}_{\text{agree}})^\top \mathbf{x}_k \tag{6}$$

which overall corresponds to

$$\begin{aligned} \underset{\mathbf{x}_k}{\text{maximize}} \quad & (\mathbf{1} - \mathbf{p}_{\text{agree}})^\top \mathbf{x}_k \\ \text{subject to} \quad & \mathbf{x}_k^\top \mathbf{1} \leq a_e \end{aligned} \tag{7}$$

The optimal $\mathbf{x}_k$ then corresponds to selecting the a_e bins which have the highest *disagreement* with the ensemble. It is important to note that the optimal $\mathbf{x}_k$ can be poor if the predictions for which the sensor agrees with the ensemble are not uniformly distributed over the correct and incorrect predictions of the ensemble. For example, the sensor can have many predictions which disagree with the ensemble when the sensor is actually right. Fig. 4 shows an example for this.

Semi-supervised Scenario: Next, suppose we have a small amount of *weak labels* for the ensemble predictions. A weak label is a form of binary feedback that indicates if a model was correct or incorrect. This feedback is more convenient and practical for a user of a HAR system to collect as they only need to respond to an activity prediction with a *yes* or *no* answer. With this feedback, we can estimate $\mathbf{p}_{e|k}$. Since the full label is not provided, we cannot use this information to also estimate $\mathbf{p}_k$. However, we can use the estimate of $\mathbf{p}_{e|k}$ to estimate $\mathbf{p}_k$ as $\mathbf{p}_k \approx \mathbf{p}_{e|k} \odot \mathbf{p}_{\text{agree}}$ where $\odot$ corresponds to the element-wise product. Intuitively, if the sensor model agrees with the ensemble model 50% of the time, then we expect it to achieve around 50% of the ensemble's accuracy. Substituting into the objective function in expression (4) we have:

$$\begin{aligned} & (\mathbf{p}_{e|k} - \mathbf{p}_k)^\top \mathbf{x}_k \\ = & (\mathbf{p}_{e|k} - \mathbf{p}_{e|k} \odot \mathbf{p}_{\text{agree}})^\top \mathbf{x}_k \\ = & \big(\mathbf{p}_{e|k} \odot (\mathbf{1} - \mathbf{p}_{\text{agree}})\big)^\top \mathbf{x}_k \end{aligned} \tag{8}$$

which overall corresponds to

$$\begin{aligned} \underset{\mathbf{x}_k}{\text{maximize}} \quad & \big(\mathbf{p}_{e|k} \odot (\mathbf{1} - \mathbf{p}_{\text{agree}})\big)^\top \mathbf{x}_k \\ \text{subject to} \quad & \mathbf{x}_k^\top \mathbf{1} \leq a_e \end{aligned} \tag{9}$$

The optimal $\mathbf{x}_k$ then corresponds to selecting the a_e bins which have the highest *disagreement* with the ensemble, but now *scaled* by $\mathbf{p}_{e|k}$. Thus the objective function is essentially

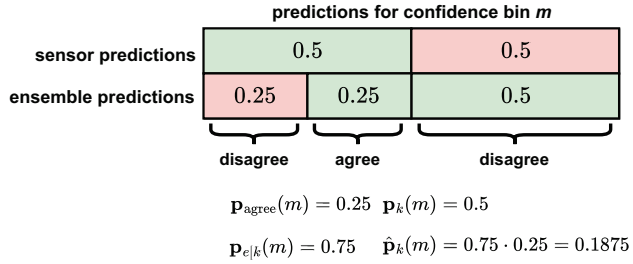

Fig. 4. A possible failure case of estimating $\mathbf{p}_k$ using $\mathbf{p}_{\text{agree}}$. The diagram shows how correct (green) and incorrect (red) predictions of the sensor and ensemble *align* for a given confidence bin m. For predictions in this bin, the probability the sensor is correct is 0.5, the probability the ensemble is correct is 0.75, and the probability that the sensor agrees with the ensemble is 0.25. Using $\mathbf{p}_k(m) \approx \mathbf{p}_{e|k}(m) \cdot \mathbf{p}_{\text{agree}}(m)$ drastically underestimates the sensor performance since the sensor is correct for a large number of predictions where it disagrees with the ensemble. This may result in a suboptimal policy.

the same as the unsupervised case, except we use the *weak label* information to update the prior assumption $\mathbf{p}_{e|k} = \mathbf{1}$ which equally weighted each bin of $\mathbf{p}_{e|k}$. However, as with the unsupervised case, the assumption $\mathbf{p}_k \approx \mathbf{p}_{e|k} \odot \mathbf{p}_{\text{agree}}$ can fail if the distribution of predictions for which the sensor agrees with the ensemble is not uniformly distributed over the correct and incorrect predictions of the ensemble (see Fig. 4).

IV. Experimental Procedure

A. Datasets and Preprocessing

We evaluate our methods on the **DSADS** [39], **RWHAR** [40], and **Opportunity** [41] datasets. These contain accelerometer data for multiple users completing many activities. The details are described below with a summary in table II.

- **DSADS** (Daily Sports and Activities Dataset): 19 daily and sports activities performed by 8 subjects for 5 minutes. Five IMUs are used on the torso, arms, and legs. We use accelerometer data from all body parts and activities.
- **RWHAR** (Real World HAR): Uses acceleration, GPS, gyroscope, light, magnetic field, and sound level data of users completing 8 activities. We use accelerometer data from all body parts and activities. We exclude users 2 and 6 as they have missing data for some body parts.
- **Opportunity**: Has readings of motion sensors from users doing daily activities in a natural sequence. Users completed multiple runs (we use run 5 for validation and others for training). We use accelerometer data from the right and left upper arm, right and left shoe, right and left lower arm, and back. Classes are imbalanced.

We preprocess each dataset as follows. For each user in a dataset, we partition the data from each activity using a sliding window of two seconds with an overlap of 50%. Since each dataset has a different sampling rate, these windows vary in length. Before partitioning, we split the data from each activity into training and validation segments. We use the first 80% for training and the last 20% for validation. Each window has three channels for the XYZ accelerometer signals. We rescale the data to be in the range $[-1, 1]$ and then standardize by subtracting the mean and dividing by the standard deviation. We apply the statistics (min, max, mean, standard deviation) from the $U - 1$ training users to the held out test user.

TABLE II
Summary of Datasets

Name	activities	users	sampling	sensors	Channels
DSADS	19	8	25 Hz	5	acc. XYZ
RWHAR	8	13	50 Hz	7	acc. XYZ
Opportunity	5	4	30 Hz	7	acc. XYZ

B. Training Details

We train a 1D CNN (see section IV-D) independently for each body part. For all models, we use the cross entropy loss with a label smoothing value of 0.1 which replaces the one-hot labels with softened probabilities. Without label smoothing the majority of the confidence scores of each model tend to be very close to one. Label smoothing helps spread out the confidence distribution. We train the models on DSADS and Opportunity for 50 epochs using a batch size of 128. For RWHAR, we train for 25 epochs using a batch size of 32. For all models, we use the SGD optimizer with a learning rate of 0.01, momentum of 0.9, and weight decay of 0.0001. We also use a cosine annealing learning rate scheduler. We train a model for each body part across three random seeds.

C. Evaluation Procedure

For evaluation, we use leave-one-user-out cross validation which means we train a model on the data from $U-1$ users and test on the held out user. We do this for every user, resulting in $K \cdot U \cdot 3$ models for each dataset where K and U are the number of body parts and users in each dataset respectively and 3 is the number of random seeds. Each sample $\mathbf{x} \in \mathbb{R}^{3 \times l}$ is a window of l samples across the XYZ channels.

We evaluate three policies: random, threshold, and the optimal estimate. To define the policies, we set aside 10% of the data from the held out test user. The random policy simply selects the ensemble with probability α_e. The threshold selects a confidence value and allocates all the predictions less than it to the ensemble. We search for the threshold which gives the best accuracy on the 10% of the data from the held out test user (under the constraint of α_e). For the optimal estimate, we use $\mathbf{acc}_k$ and $\mathbf{acc}_{e|k}$ to estimate $\mathbf{p}_{\text{diff}}$ for each sensor and then select the top a_e positive bins as explained in section III-C. Algorithm 1 explains this step-by-step. Recall that in our setting, we do not have direct access to the model f_k and the raw data $\mathbf{x}$ but we show these algorithm 1 for clarity.

For the unsupervised scenario, we do not use any labeled data and estimate $\mathbf{p}_{\text{agree}}$ from the 10% of data set aside for defining the policies. For the semi-supervised scenario, we only set aside 1% or 5% of the dataset to estimate $\mathbf{p}_{e|k}$. Recall that this only requires weak feedback, not the full label.

We evaluate on a uniform sweep of 11 utilization budgets from $\frac{1}{K} \rightarrow 1$. This corresponds to α_e values of

Algorithm 1 Estimate Optimal Policy for Sensor s_k

Require: f_k, the sensor model; f_e, the ensemble model; M, the # of bins; $D = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$, ∼500 labeled IID samples from the test user; α_e, the ensemble budget

$\text{Preds}_k \leftarrow \text{argmax}_c\{f_k(D)\}$ ▷ get sensor predictions
$\text{Confs}_k \leftarrow \max_c\{f_k(D)\}$ ▷ get sensor confidence scores
$\text{Preds}_e \leftarrow \text{argmax}_c\{f_e(D)\}$ ▷ get ensemble predictions
$\text{Confs}_e \leftarrow \max_c\{f_e(D)\}$ ▷ get ensemble conf. scores
sort(Preds_k, Confs_k) ▷ sort sensor preds. by ↑ confidence
sort(Preds_e, Confs_k) ▷ sort ens. preds. by ↑ sensor conf.
Partition sensor and ens. preds. into M bins of size $\frac{N}{M}$
for bin m in M **do**
    $\mathbf{acc}_k[m] \leftarrow$ accuracy of $\text{Preds}_k[\frac{mN}{M} : \frac{(m+1)N}{M}]$
    $\mathbf{acc}_{e|k}[m] \leftarrow$ accuracy of $\text{Preds}_e[\frac{mN}{M} : \frac{(m+1)N}{M}]$
end for
$\mathbf{acc}_{\text{diff}} \leftarrow \mathbf{acc}_{e|k} - \mathbf{acc}_k$ ▷ get accuracy margin
$a_e \leftarrow \lfloor \frac{\alpha_e}{1/M} \rfloor$ ▷ get discrete bin budget
idxs ← sort($\mathbf{acc}_{\text{diff}}$)[: a_e] ▷ indices of values in ↓ order
$\mathbf{x}_k \leftarrow \mathbf{0}$ ▷ initialize policy to M zeros
$\mathbf{x}_k[\mathbf{acc}_{\text{diff}}[\text{idxs}] > 0] \leftarrow 1$ ▷ set a_e most positive entries
return $\mathbf{x}_k$

$\{0.0, 0.1, 0.2, \ldots, 1.0\}$. We use $M = 10$ bins for the reliability diagrams since $\frac{1}{M} = 0.1$ so we can exactly capture each α_e.

The evaluation procedure is as follows. We truncate the remaining 90% of the test user data to be a multiple of K. We then query the sensors in a round-robin fashion. At each prediction round, we use the current's sensor prediction and policy to determine whether to call the ensemble. If the policy signals to call the ensemble, we first check if such an action will not cause any sensor to exceed the specified utilization budget. If it does, we use the current sensor's prediction. See Algorithm 2 for the control flow. Note that the ensemble model simply averages the softmax scores from each sensor but we show it as a model f_e in Algorithm 2 to be clear and concise.

Since these policies are subject to randomness in the order of the data samples, we run our evaluation across three random seeds and get the average and standard deviation of the results. For the evaluation metric, we use the macro-averaged F1-score since there is some class imbalance in the RWHAR and Opportunity datasets. This metric is most commonly used for HAR evaluation. When tuning the policies, we use accuracy rather than the F1-score since we may only have a few samples per class in each bin which can result in a large variance of F1-scores. This can make the policies much more sensitive to the random selection of the samples used for tuning.

D. Model Architectures

We use simple 1D convolutional neural network architectures given that these models would be running on resource constrained devices. The architectures, parameters, and MACs (multiply accumulate) used for each dataset are summarized in table III. In the table, *channels* is abbreviated to 'Ch.', 'S/P' means Stride/Padding and '-' means that the column is not applicable for the corresponding layer.

Algorithm 2 Policy Evaluation

Require: $\{\mathbf{x}_k\}_{k=1}^{K}$, the policies for all K sensors; $\{f_k\}_{k=1}^{K}$ the models for all sensors; f_e, the ensemble model; $D = \{\mathbf{x}_i, y_i\}_{i=n}^{N}$, the remaining labeled samples from the test user (assume $N\%K = 0$); u, the utilization budget

$\mathbf{u} \leftarrow \mathbf{0}$ ▷ init sensor utilizations
for sample $(\mathbf{x}_i, y_i)$ in D **do**
    $k = i\%K$ ▷ round-robin
    $\text{Pred}_k, \text{Conf}_k = \text{argmax}_c\{f_k(\mathbf{x}_i)\}, \max_c\{f_k(\mathbf{x}_i)\}$
    $m \leftarrow$ bin index for Conf_k
    $d \leftarrow \mathbf{x}_k[m]$ ▷ policy decision
    if $(d == 1)$ and $(\exists k$ s.t. $(\mathbf{u}[k] + 1)/i > u)$ **then**
        $d \leftarrow 0$ ▷ budget exceeded
    end if
    if $d == 0$ **then** ▷ choose sensor prediction
        output_prediction ← Pred_k
        $\mathbf{u}[k] \leftarrow \mathbf{u}[k] + 1$
    else ▷ choose ensemble prediction
        output_prediction ← $\text{argmax}_c\{f_e(\mathbf{x}_i)\}$
        $\mathbf{u} \leftarrow \mathbf{u} + \mathbf{1}$
    end if
end for

V. Results and Discussion

In this section, the F1-score for all policies are plotted over a sweep of utilization budgets to visualize the trade-off of each policy. Given the variation induced by different users as well as the random seeds, we plot the mean and standard deviation for each scenario. Specifically, Fig. 5 shows the mean and standard deviation of F1-scores across all held out test users, while Fig. 6 shows the mean and standard deviation of F1-scores across each random seed. In figures 5 and 6 *optmial_estimate_U* (dashed red) is the unsupervised policy, while *optimal_estimate_1* (dashed purple) and *optimal_estimate_5* (dashed brown) are the semi-supervised policies. The 1 and 5 correspond to the percent of data used for weak feedback in the semi-supervised scenario.

Note that the F1-score and utilization budget are competing objectives. Thus, in the plots, curves which are higher (larger F1-score) for a given utilization budget are better. To emphasize the differences between the optimal estimate and the threshold policy, we show three explicit examples of how the reliability diagram structure influences the resulting policies in figures 7, 8, 9. Note that each of these examples show the policies for *one* sensor from the RWHAR dataset. The tangible implications of these results are discussed in section V-B.

A. Supervised Scenario Results

The supervised scenario serves as the upper bound for policy performance since we use labeled data to tune the policies. In figures 5 and 6 the supervised policies are the *threshold* (solid orange) and *optimal_estimate* (solid green) lines. For all random seeds and datasets in Fig. 5, the curve for the optimal estimate matches or improves upon the threshold. For the per-user results across datasets in Fig. 6, we see that the threshold

TABLE III
MODEL ARCHITECTURES

	DSADS Architecture (3×50 Input)				**RWHAR** Architecture (3×100 Input)				**Opportunity** Architecture (3×60 Input)			
Layer	In Ch.	Out Ch.	Kernel	S/P	In Ch.	Out Ch.	Kernel	S/P	In Ch.	Out Ch.	Kernel	S/P
1D Conv/ReLU	3	16	5	1/1	3	16	5	1/None	3	16	5	1/2
1D Conv/ReLU	16	16	3	3/None	16	16	3	3/None	16	16	3	3/None
1D Conv/ReLU	16	16	3	1/None	16	16	3	1/None	16	16	3	1/None
1D Conv/ReLU	16	16	3	1/None	16	16	3	3/None	16	16	3	3/None
1D Conv/ReLU	16	32	12	1/None	16	32	10	1/None	16	24	16	1/None
Linear	32	19	-	-/-	32	8	-	-/-	24	5	-	-/-
Model Statistics	**# Params**: 9.41 K		**# MACs**: 52.02 K		**# Params**: 8.02 K		**# MACs**: 84.44 K		**# Params**: 8.99 K		**# MACs**: 63.99 K	

and optimal estimate policies perform similarly for some users while for others there is a large gap. As discussed in the *implications* portion of section III-C, our policy improves upon the threshold when $\mathbf{p}_{\text{diff}}$ is unstructured which depends on the model and data; otherwise the performance will be similar.

B. Extending the Sensor System Lifetime

To interpret the improvement of our policy over the threshold in terms of energy savings, recall the following from section III-A: *assume that inference dominates energy consumption so that the lifetime of a sensor is defined by I, the number of inferences it can make before running out of battery*. Under this setting, the ratio of two utilization budgets is proportional to the increase in the system lifetime. Suppose a sensor can last $I = 1000$ inferences. If a sensor has a budget of $u = 0.5$, then it can last $\frac{I}{u} = \frac{1000}{0.5} = 2000$ time steps, where the sensor system makes a prediction every time step.

Now consider two utilization budgets u_1 and u_2. Under budget u_1, the sensor system will last $\frac{I}{u_1} / \frac{I}{u_2} = \frac{u_2}{u_1}$ times as long as the system under budget u_2. For example, if $u_1 = 0.8$ and $u_2 = 0.4$, the system will last $\frac{0.4}{0.8} = \frac{1}{2}$ as long under budget u_1 compared to u_2. As a tangible result, consider the middle plot from Fig. 5 (Seed 456, RWHAR Dataset). Suppose we fix the F1-score at 0.85 based on the desired performance for some application. We see that our policy (green) crosses this point at a utilization of $u_1 = 0.6$ while the threshold policy (orange) crosses this point at a utilization of $u_2 = 0.75$. Then under this F1-score constraint, the sensor system would last $\frac{0.75}{0.6} = 1.25$ times as long using our policy.

C. Unsupervised and Semi-supervised Scenario Results

In general, the unsupervised scenario serves as the lower bound (beyond the random policy) since no labeled data is used, while the semi-supervised scenarios represent intermediate cases between the lower and upper bounds. In Fig. 5 we see that the unsupervised and semi-supervised policies are competitive with the supervised threshold, matching the F1-score for many utilization budgets. We note that past a certain budget, the unsupervised and semi-supervised policies plateau towards the peak of the random policy while the supervised threshold policy plateaus toward the peak of the optimal estimate. This discrepancy is a result of the poor estimates of $\mathbf{p}_k$ discussed in Fig. 4. For higher utilization budgets, the unsupervised and semi-supervised policies are underestimating $\mathbf{p}_k$ and thus they prioritize the ensemble. At the extreme of $u = 1.0$, the the unsupervised and semi-supervised policies match the peak of the random policy which corresponds to always choosing the ensemble. However, as seen by the supervised policies, a higher F1-score can be achieved by choosing the sensor over the ensemble for certain bins. Without a mechanism to estimate $\mathbf{p}_k$, we cannot determine when the sensor outperforms the ensemble which is a limitation of the semi-supervised and unsupervised cases.

Despite this, we see strong results for lower and mid-utilization budgets which may serve as more practical budgets since a given application will want to minimize utilization to extend the system lifetime. **In addition, since the semi-supervised and unsupervised policies do not require full labels, they can be updated over time**. Thus, if a user changes the way they carry out activities (e.g. by moving to a new environment), the supervised policies may perform poorly as they were tuned using labels from a different data distribution. In such a case where a distribution shift could be detected, the supervised policy may revert to a random policy which clearly performs worse than the unsupervised policy.

D. Reliability Diagram Analysis

As discussed in section III-C, the structure of the reliability diagrams influence the optimality of the threshold policy. We show three examples, one in which the threshold is optimal (Fig. 8) and two in which the threshold is suboptimal (figures 7 and 9). For these examples, $\alpha_e = 0.5$ and $a_e = 5$. In these figures, the green numbers correspond to the a_e bins with largest positive accuracy margin for the ensemble, while the orange arrow corresponds to the threshold.

For the reliability diagrams, blue bars represent the accuracy of a bin, while red bars represent the gap to the confidence of each bin. Bins with red bars above blue bars represent overconfident bins whose confidence is higher than their accuracy while bins with red bars below blue bars represent underconfident bins whose confidence is lower than their accuracy. The bins in the reliability diagrams have variable width since they each have an equal number of predictions. Thus, the earlier bins tend to be wider since the majority of predictions tend to be concentrated at high confidence values.

The results clearly show that it is not sufficient or necessary for the sensor model to be calibrated for a threshold based policy to perform well; rather the relationship between the

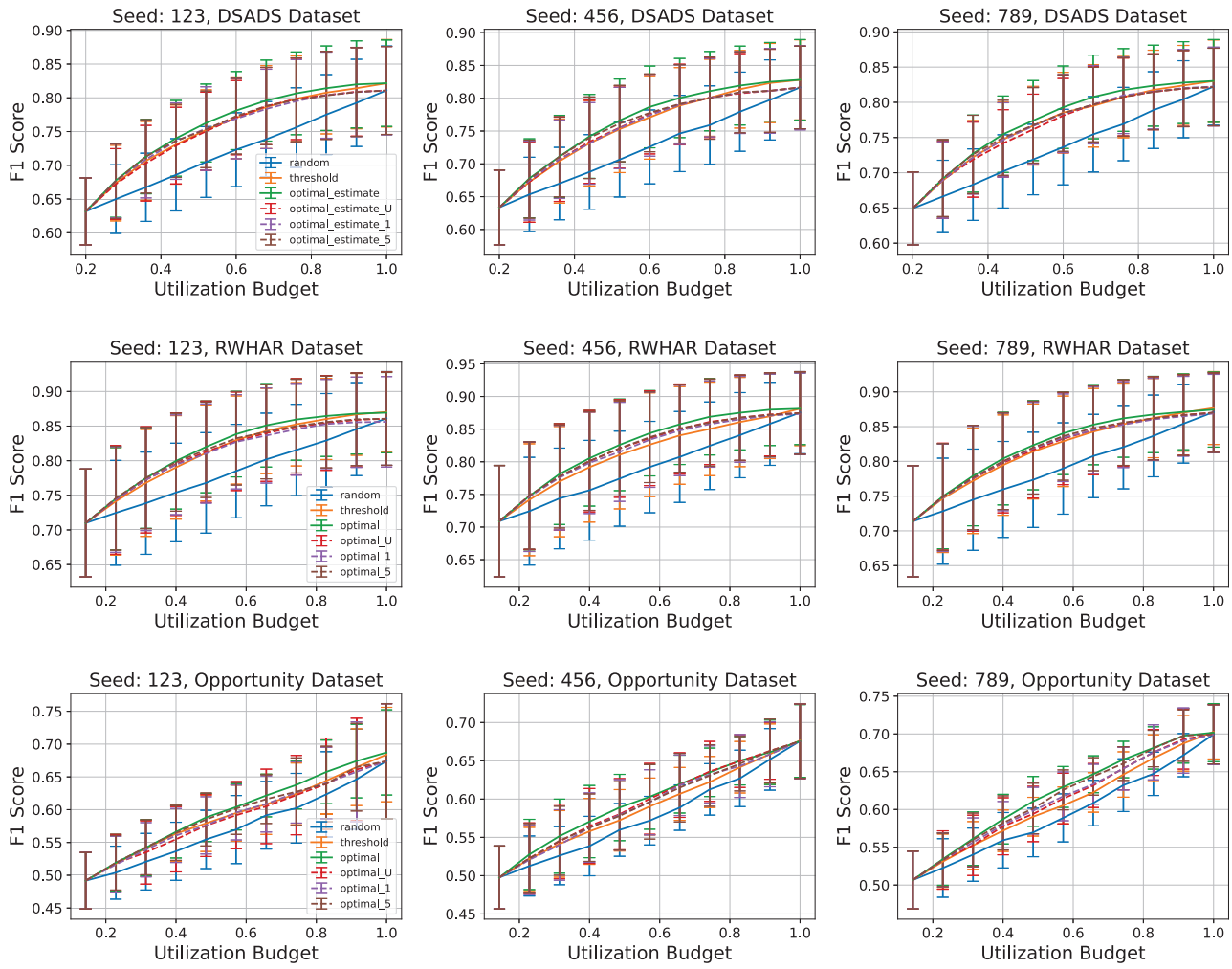

Fig. 5. Comparison of six policies on the DSADS, RWHAR, and Opportunity datasets. Each row of plots corresponds to a different dataset and each column of plots corresponds to a different random seed. We plot the F1-score for a sweep of utilization budgets from $\frac{1}{K} \rightarrow 1$. Higher F1-score for each utilization budget is better. Each curve is the average over the held out users and the error bars show the standard deviation. The standard deviation across users varies more than across random seeds reflecting how heterogeneity in the data significantly affects the policy. The main takeaway from these plots is described in section V-B which describes the tangible benefit of our policy requiring less utilization for a given accuracy in terms of extending the sensor lifetime.

confidence and accuracy of successive models needs to be structured. Further, we see that highly unstructured reliability diagrams appear in practice, reflecting the need for a more general policy to account for this to avoid allocating predictions to the ensemble when there is no expected gain in accuracy.

VI. Conclusion and Future Work

In this paper, we carried out an analysis of sensor selection for IMU-based HAR by optimizing the system accuracy under the constraint of a sensor utilization budget in the cascaded inference setting. Our results directly show the need and benefit of a general policy beyond a single threshold. We explicitly visualized the unstructured relationship that can appear between a sensor's confidence and the ensemble accuracy; a threshold is not capable of modeling this, leading to wasted computation for certain confidence regions. Our policy can handle this lack of structure and directly extends the sensor system lifetime when a threshold is suboptimal. Finally, we show that these benefits can be achieved in the practical setting where little to no labeled data is available for tuning the policy. In future work, we plan to generalize to settings in which any subset of sensors can be used in the ensemble, rather than all of them. In this case, sensors can be queried sequentially in a multi-level cascade allowing for further efficiency gains.

Acknowledgement

This research was supported in part by NSF Grant CCF-2107085 and a Graduate Fellowship provided by the Cockrell School of Engineering at The University of Texas at Austin.

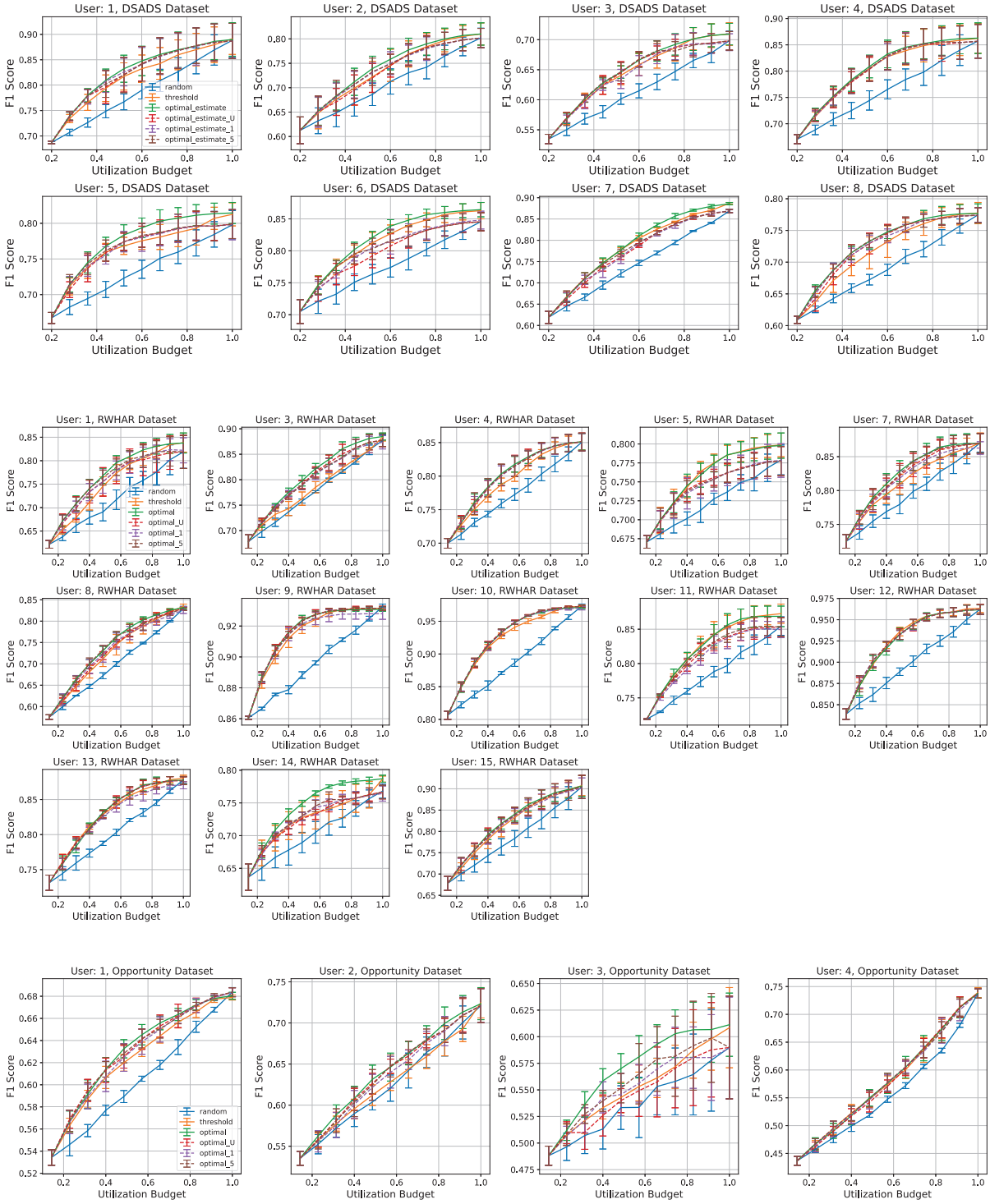

Fig. 6. Comparison of six policies on the DSADS, RWHAR, and Opportunity datasets. Each set of plots corresponds to per-user results for a given dataset. We plot the F1-score for a sweep of utilization budgets from $\frac{1}{K} \rightarrow 1$. Higher F1-score for each utilization budget is better. Each curve is the average over the three random seeds and the error bars show the standard deviation. Results across random seeds vary less than across users as shown by the lower standard deviation. Note that some users benefit greatly from the optimal estimate while for others the threshold performs similarly. Since we can only estimate the optimal policy, there may be small regions where the threshold performs slightly better in the per-user results. We find that this discrepancy is reduced when more data and bins are used to estimate the optimal and threshold policies.

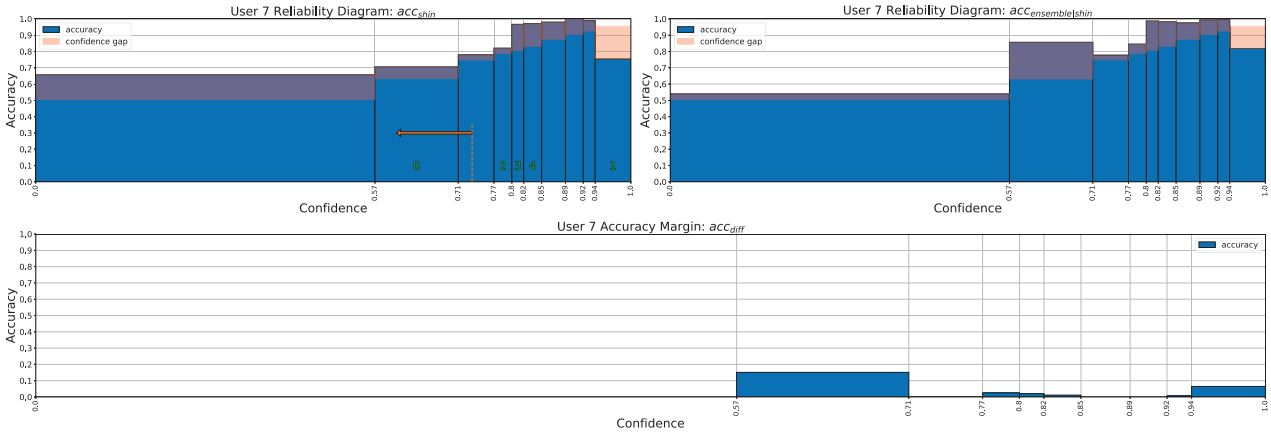

Fig. 7. The reliability diagrams for the shin sensor and ensemble for user 7 in the RWHAR dataset are shown at the top. The bottom shows the difference in the accuracy of each bin which represents an empirical estimate of $\mathbf{p}_{\text{diff}}$. We see that $\mathbf{acc}_{\text{diff}}$ is not monotonic and many bins are actually negative (shown with no height here) since the sensor has higher accuracy than the ensemble in certain sensor confidence regions. The threshold (orange) policy is not able to model this and incorrectly allocates the first bin to the ensemble despite the sensor performing significantly better.

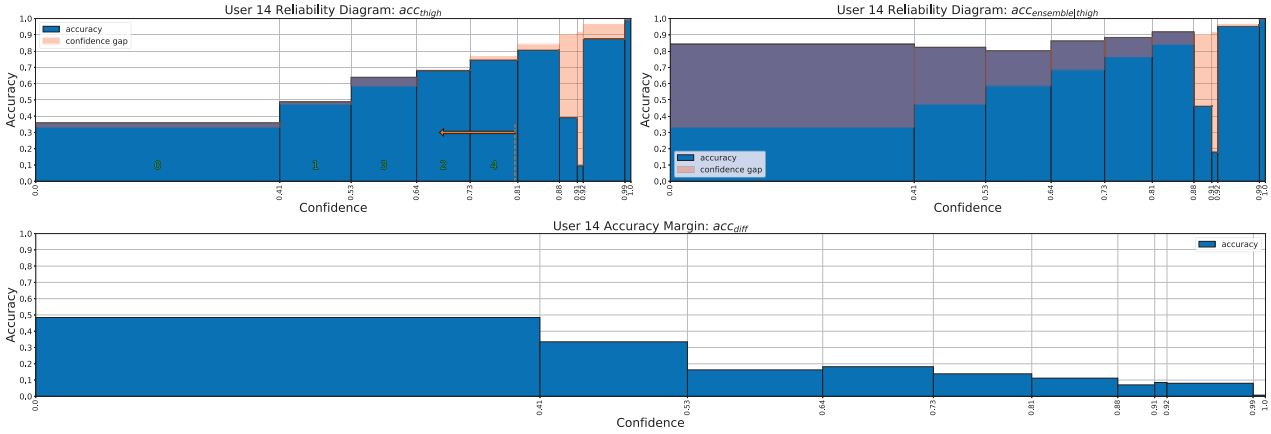

Fig. 8. The reliability diagrams for the thigh sensor and ensemble for user 14 in the RWHAR dataset are shown at the top. The threshold and optimal estimate policies are approximately the same despite the miscalibration of the sensor model.

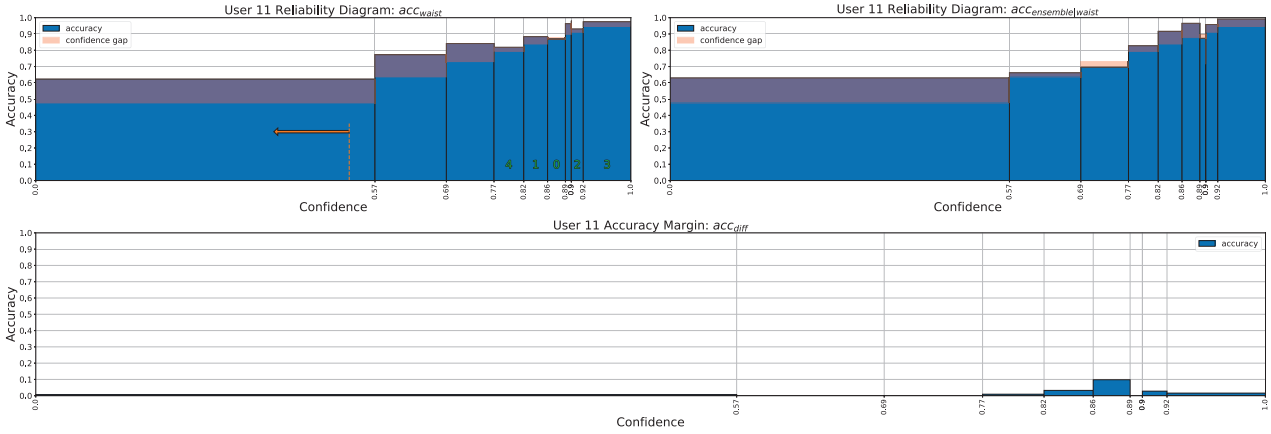

Fig. 9. The reliability diagrams for the waist sensor and ensemble for user 14 in the RWHAR dataset are shown at the top. This shows a scenario where the threshold and optimal estimate policies are very different even though the sensor model's reliability diagram is approximately monotonic. This emphasizes the need to exploit the relationship between the two models and that simply calibrating the sensor model is not sufficient.

REFERENCES

[1] S. Usmani, A. Saboor, M. Haris, M. A. Khan, and H. Park, "Latest research trends in fall detection and prevention using machine learning: A systematic review," *Sensors*, vol. 21, no. 15, p. 5134, 2021.

[2] L. Schrader, A. Vargas Toro, S. Konietzny, S. Rüping, B. Schäpers, M. Steinböck, C. Krewer, F. Müller, J. Güttler, and T. Bock, "Advanced sensing and human activity recognition in early intervention and rehabilitation of elderly people," *Journal of Population Ageing*, vol. 13, pp. 139–165, 2020.

[3] S. Zhang, Y. Li, S. Zhang, F. Shahabi, S. Xia, Y. Deng, and N. Alshurafa, "Deep learning in human activity recognition with wearable sensors: A review on advances," *Sensors*, vol. 22, no. 4, p. 1476, 2022.

[4] K. Chen, D. Zhang, L. Yao, B. Guo, Z. Yu, and Y. Liu, "Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities," *ACM Computing Surveys (CSUR)*, vol. 54, no. 4, pp. 1–40, 2021.

[5] R. Gravina, P. Alinia, H. Ghasemzadeh, and G. Fortino, "Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges," *Information Fusion*, vol. 35, pp. 68–80, 2017.

[6] C. Hou, "A study on imu-based human activity recognition using deep learning and traditional machine learning," in *2020 5th International Conference on Computer and Communication Systems (ICCCS)*. IEEE, 2020, pp. 225–234.

[7] Y. Guan and T. Plötz, "Ensembles of deep lstm learners for activity recognition using wearables," *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 2, pp. 1–28, 2017.

[8] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[9] C. A. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," *Expert systems with applications*, vol. 59, pp. 235–244, 2016.

[10] J. Yang, M. N. Nguyen, P. P. San, X. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition." in *IJCAI*, vol. 15. Buenos Aires, Argentina, 2015, pp. 3995–4001.

[11] J.-H. Choi and J.-S. Lee, "Confidence-based deep multimodal fusion for activity recognition," in *Proc. of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, 2018, pp. 1548–1556.

[12] H. Guo, L. Chen, L. Peng, and G. Chen, "Wearable sensor based multimodal human activity recognition exploiting the diversity of classifier ensemble," in *Proc. of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016, pp. 1112–1123.

[13] S. Münzner, P. Schmidt, A. Reiss, M. Hanselmann, R. Stiefelhagen, and R. Dürichen, "Cnn-based sensor fusion techniques for multimodal human activity recognition," in *Proc. of the 2017 ACM International Symposium on Wearable Computers*, 2017, pp. 158–165.

[14] V. Radu, C. Tong, S. Bhattacharya, N. D. Lane, C. Mascolo, M. K. Marina, and F. Kawsar, "Multimodal deep learning for activity and context recognition," *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 4, pp. 1–27, 2018.

[15] V. Bloom, V. Argyriou, and D. Makris, "Dynamic feature selection for online action recognition," in *Human Behavior Understanding: 4th International Workshop, HBU 2013, Barcelona, Spain, October 22, 2013. Proc. 4*. Springer, 2013, pp. 64–76.

[16] G. Dulac-Arnold, L. Denoyer, P. Preux, and P. Gallinari, "Datum-wise classification: a sequential approach to sparsity," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proc., Part I 11*. Springer, 2011, pp. 375–390.

[17] D. Gordon, J. Czerny, T. Miyaki, and M. Beigl, "Energy-efficient activity recognition using prediction," in *2012 16th International Symposium on Wearable Computers*. IEEE, 2012, pp. 29–36.

[18] S. Karayev, M. J. Fritz, and T. Darrell, "Dynamic feature selection for classification on a budget," in *International Conference on Machine Learning (ICML): Workshop on Prediction with Sequential Models*, vol. 95, 2013.

[19] X. Yang, Y. Chen, H. Yu, Y. Zhang, W. Lu, and R. Sun, "Instance-wise dynamic sensor selection for human activity recognition," in *Proc. of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 1104–1111.

[20] P. Zappi, C. Lombriser, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Tröster, "Activity recognition from on-body sensors: accuracy-power trade-off by dynamic sensor selection," in *Wireless Sensor Networks: 5th European Conference, EWSN 2008, Bologna, Italy, January 30-February 1, 2008. Proc.* Springer, 2008, pp. 17–33.

[21] P. Warden, M. Stewart, B. Plancher, C. Banbury, S. Prakash, E. Chen, Z. Asgar, S. Katti, and V. J. Reddi, "Machine learning sensors," *arXiv preprint arXiv:2206.03266*, 2022.

[22] Y. Geifman and R. El-Yaniv, "Selectivenet: A deep neural network with an integrated reject option," in *International conference on machine learning*. PMLR, 2019, pp. 2151–2159.

[23] X. Wang, Y. Luo, D. Crankshaw, A. Tumanov, F. Yu, and J. E. Gonzalez, "Idk cascades: Fast deep learning by learning not to overthink," *arXiv preprint arXiv:1706.00885*, 2017.

[24] X. Wang, D. Kondratyuk, E. Christiansen, K. M. Kitani, Y. Alon, and E. Eban, "Wisdom of committees: An overlooked approach to faster and more accurate models," *arXiv preprint arXiv:2012.01988*, 2020.

[25] Ö. F. Ertuğrul and Y. Kaya, "Determining the optimal number of body-worn sensors for human activity recognition," *Soft Computing*, vol. 21, no. 17, pp. 5053–5060, 2017.

[26] A. Gangrade, A. Kag, and V. Saligrama, "Selective classification via one-sided prediction," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2179–2187.

[27] N. García-Pedrajas, D. Ortiz-Boyer, R. del Castillo-Gomariz, and C. Hervás-Martínez, "Cascade ensembles," in *Computational Intelligence and Bioinspired Systems: 8th International Work-Conference on Artificial Neural Networks, IWANN 2005, Vilanova i la Geltrú, Barcelona, Spain, June 8-10, 2005. Proc. 8*. Springer, 2005, pp. 598–603.

[28] W. Jitkrittum, N. Gupta, A. K. Menon, H. Narasimhan, A. S. Rawat, and S. Kumar, "When does confidence-based cascade deferral suffice?" *arXiv preprint arXiv:2307.02764*, 2023.

[29] A. Kag, I. Fedorov, A. Gangrade, P. Whatmough, and V. Saligrama, "Efficient edge inference by selective query," in *The Eleventh International Conference on Learning Representations*, 2022.

[30] H. Mozannar and D. Sontag, "Consistent estimators for learning to defer to an expert," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7076–7087.

[31] H. Narasimhan, W. Jitkrittum, A. K. Menon, A. Rawat, and S. Kumar, "Post-hoc estimators for learning to defer to an expert," *Advances in Neural Information Processing Systems*, vol. 35, pp. 29 292–29 304, 2022.

[32] T. Narayan, H. Jiang, S. Zhao, and S. Kumar, "Predicting on the edge: Identifying where a larger model does better," *arXiv preprint arXiv:2202.07652*, 2022.

[33] K. Trapeznikov and V. Saligrama, "Supervised sequential classification under budget constraints," in *Artificial Intelligence and Statistics*. PMLR, 2013, pp. 581–589.

[34] Y. Han, G. Huang, S. Song, L. Yang, H. Wang, and Y. Wang, "Dynamic neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7436–7456, 2021.

[35] S. Enomoro and T. Eda, "Learning to cascade: Confidence calibration for improving the accuracy and computational cost of cascade inference systems," in *Proc. of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, 2021, pp. 7331–7339.

[36] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1321–1330.

[37] M. H. DeGroot and S. E. Fienberg, "The comparison and evaluation of forecasters," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 32, no. 1-2, pp. 12–22, 1983.

[38] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proc. of the 22nd International Conference on Machine Learning*, 2005, pp. 625–632.

[39] B. Barshan and K. Altun, "Daily and Sports Activities," UCI Machine Learning Repository, 2013, DOI: https://doi.org/10.24432/C5C59F.

[40] T. Sztyler and H. Stuckenschmidt, "On-body localization of wearable devices: An investigation of position-aware activity recognition," in *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2016, pp. 1–9.

[41] D. Roggen, A. Calatroni, L.-V. Nguyen-Dinh, R. Chavarriaga, and H. Sagha, "OPPORTUNITY Activity Recognition," UCI Machine Learning Repository, 2012, DOI: https://doi.org/10.24432/C5M027.