Egocentric Activity Recognition on a Budget

Rafael Possas * Sheila Pinto Caceres *
Fabio Ramos
School of Information Technologies
University of Sydney
{rafael.possas, fabio.ramos}@sydney.edu.au, spin3586@uni.sydney.edu.au

Abstract

Recent advances in embedded technology have enabled more pervasive machine learning. One of the common applications in this field is Egocentric Activity Recognition (EAR), where users wearing a device such as a smartphone or smartglasses are able to receive feedback from the embedded device. Recent research on activity recognition has mainly focused on improving accuracy by using resource intensive techniques such as multi-stream deep networks. Although this approach has provided state-of-the-art results, in most cases it neglects the natural resource constraints (e.g. battery) of wearable devices. We develop a Reinforcement Learning model-free method to learn energy-aware policies that maximize the use of low-energy cost predictors while keeping competitive accuracy levels. Our results show that a policy trained on an egocentric dataset is able use the synergy between motion and vision sensors to effectively tradeoff energy expenditure and accuracy on smartglasses operating in realistic, real-world conditions.

1. Introduction

The use of wearable technologies has increased the demand for applications that can efficiently process large amounts of raw data from motion sensors and videos. Recognizing an activity is possibly the first step in understanding the context in which a person is embedded. Therefore, creating robust methods to recognize activities under the constraints of smart devices becomes one of the main challenges in the awakening of this technological trend.

Extensive research has been done in EAR. Traditional vision methods commonly encode prior knowledge of the egocentric paradigm by using handcrafted features to build mid-level representations based on the detection/segmentation of objects [6, 15, 17, 19, 26], hands [15, 16, 17, 19, 23, 37], gaze [15, 16], among others. However, the use of these specific representations prevent the generalization to a more realistic set of activities. For example, hand detection has been widely used in kitchen-related activities but would be ineffective in the recognition of the walking activity. Ideally, learning algorithms should be less dependent on prior knowledge and instead be able to learn adequate features from data automatically [3]. Deep learning methods have shown that they can achieve this task quite well in several domains. Recent research on Activity Recognition has used very deep neural networks from external [1, 4, 9, 24, 30] and egocentric [32] perspectives achieving encouraging results. These models, however, demand high computing resources and energy which are commonly not available in wearable devices hindering their use in most of real-life applications.

The egocentric domain also entails new challenges. Cameras often produce shaken and blurred shots due to the natural movements of the wearer. Unintelligible images can be produced by real life situations such as dark and rainy environments. Therefore, alternative sources of information such as motion sensors can be used to increase the prediction performance at a low power consumption cost. In fact, the use of sensors such as accelerometers [2, 6] have played an important role in EAR. Traditionally, these devices were attached to several parts of the body [2, 12, 43], to external objects [14, 40] and to the ambient [12, 43] which often limited their use to controlled environments which can be quite different to a real-life setting.

Few approaches [13, 41, 44] have tackled activity recognition while considering the energy constraints on devices such as smartphones. However, they consider a small group of simple activities. State-of-the-art performance on more complex activities comes from recent work [32, 33] that uses data from both camera and motion sensors to perform activity recognition from an egocentric perspective. Their methods, nevertheless, are extremely energy inefficient as they rely on both resource intensive multi-stream Deep Neural Networks and on expensive feature extraction techniques such as the one from stabilized optical flow.

*Both authors contributed equally to this work.

In this paper, we propose an energy-aware framework for EAR whose goal is to minimize energy consumption while maintaining reasonable predictive performance. Specifically, we make the following contributions:

- We propose a Reinforcement Learning (RL) Policy Gradient framework that balances energy consumption and accuracy through a customizable hyper-parameter.
- We provide a novel egocentric dataset called DataEgo with continuous activities collected in real conditions with high variations in illumination, different environments, and mulitple subjects.
- We achieve higher accuracy over other benchmark, Multimodal egocentric dataset [32] while using less energy compared to previous work.

Results from rigorous experiments over the new dataset DataEgo, and the Multimodal dataset [32] show that our trained policy is able to optimize the use of energy effectively while maintaining a competitive predictive accuracy.

2. Related Work

Activity recognition from visual information has attracted great attention in the last years. It has been traditionally tackled using third-person view cameras and then extended to egocentric cameras. In the external perspective context, the successful application of deep learning approaches for image classification [11] has resulted in their extension to the context of activity recognition over video [1, 4, 9, 24, 30].

At first, a natural step has been to extend Convolutional Neural Networks (CNNs) 2D filters that explore spatial information over images to 3D filters to add temporal information over videos as it was done in [7]. In fact, Karpathy et al. [9], explored spatio-temporal schemes using both 2D and 3D convolutions, finding that 3D convolutions over videos were giving a very short gain in performance in comparison with 2D convolutions over images of single frames of videos. This fact suggests that (CNNs) architectures are not able to properly learn motion features, due to the non-existence of sufficiently large video datasets. Simonyan et al. [30] proposed the use of stacks of dense optical flows to encode temporal information achieving higher success.

In order to preserve longer temporal information, some architectures based on Recurrent Neural Networks (RNNs) have been proposed [1, 4, 24]. Donahue et al. [4] have used Long Short-Term Memory (LSTM) networks over features obtained from a CNN applied to single frames of videos. Ng et al. [24] proposed a similar idea over both single frames and optical flow. Finally, Ballas et al. [1] suggested the use of convolutions inside the recurrent units of GRU to better capture temporal features of the sequence of images.

In the context of EAR, traditional vision methods have encoded prior knowledge by using handcrafted features and mid-level representations involving the detection of objects [6, 15, 17, 19, 26], hands [15, 16, 17, 19, 23, 37], gaze [15, 16], motion [16, 17, 25, 28, 42], among others. Object-based techniques presumes that an activity can be inferred by the group of objects that appear in a video. Thus, object-based techniques rely on the object recognition domain and, therefore, inherit its challenges. Another commonly used strategy has involved the use of optical flows to express motion [32, 42]. Motion-based techniques focus on the fact that different kind of activities create different body motions presenting remarkable robustness to deal with some of the vision challenges. However, they perform poorly when dealing with activities that lacks movement patterns such as sitting, watching TV, and reading.

Most of the aforementioned representations prevents the learning algorithm from generalizing to a more realistic set of activities. Ideally, a framework must learn meaningful features automatically. Recent advances in third-person activity recognition have used deep learning to address this requirement. Simonyan et al. [30] proposed a Two-Streams CNN approach over spatial (image frames) and temporal (optical flows) streams and has been taken as a baseline of other works since it outperformed hand-crafted features methods [38]. Song et al. [32] extended this approach to the EAR domain obtaining encouraging results.

Wearable cameras can produce visual information that is affected by real conditions of the wearer leading to unintelligible images. Thus, the use of other sources of information is needed. Wearable devices have typically been attached to some parts of the body [2, 12, 43], to external objects [14, 40] and to the environment [12, 43] and their use was often limited to controlled experiments, differing in high degree from a real life application.

Reinforcement Learning (RL) is another technique that has achieved successful results lately. Successes are wide ranging, from playing Atari games [21] to teaching a robot to play soccer [27] or detecting objects in vision tasks [18]. In the context of budget-restricted prediction, Karayev et al. [8] applied RL to determine the order of computation of handcrafted features. Although budget awareness is desirable, handcrafted features increments the complexity and could prevent generalization to new domains as described before. To the best of our knowledge, there is no current application of RL on the EAR problem.

Energy awareness has been greatly overlooked in all recent multi-stream activity recognition work. This factor can not be ignored anymore if we desire to use these methods in a realistic setting which has major constraints such as battery consumption. Despite all previous work, there have been no/few attempts to balance computational resources and other constraints with the overall accuracy on activity

recognition tasks. Therefore we propose a Reinforcement Learning framework that makes use of policy learning in order to balance two different activity predictors using data from motion and vision sensors.

3. Method

The goal of the method introduced in this section is to perform decision making in the EAR context using a RL approach that balances energy consumption and accuracy on resource constrained devices.

3.1. Overview

We consider tasks in which a wearable device receives data from motion sensors and uses a policy $\pi_\theta(a_t|s_t)$ to take actions $a \in (\alpha, \beta)$, where action α means motion predictor is used while action β means the vision predictor is used to recognize the activity. At each time-step t we give a reward r_t that reflects the accuracy and energy consumption of the selected action. The long-term return $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$ is the total accumulated reward from time step t with discount factor $\gamma \in (0, 1]$. We also define a training dataset $D = \{x_t, y_t\}_{t=1}^{N}$ with $x_t = \{I_t, acc_t, gyr_t\}$ where I_t is a sequence of images from the device's camera, and acc_t and gyr_t is a sequence of accelerometer/gyroscope x,y,z values. The class labels for the 20 activities are denoted by $y_t \in \{y^1, ..., y^{20}\}$.

Our framework is shown in Figure 1. We learn a policy for energy optimization through a LSTM [5], mapping accelerometer and gyroscope data to actions. The network outputs a probability distribution over actions $a \in (\alpha, \beta)$ that attempts to select actions with higher average rewards. The model's actions define whether the final prediction should come from a motion predictor, defined by $\rho^m(x_t)$, or vision predictor, defined by $\rho^v(x_t)$. These are represented by a separate LSTM and LRCN Network [4] respectively.

3.2. LSTM (Motion) Predictor

For predictions of sensor data we use a LSTM network. This is a specific Recurrent Neural Network (RNN) architecture that has been widely used to solve problems where data has an intrinsic temporal structure. Its main idea is to regulate the flow of information through neurons' specific gates. They control how much information should be remembered or whether the network should forget or keep a memory. LSTMs were designed to solve the problem of long-term dependencies on vanilla RNNs, where data with long temporal dependencies caused the gradient of the network to vanish or explode. Our motion predictor architecture is composed of one layer with 64 neurons unrolled through time and is optimized using RMSProp with learning rate of 0.001. The main advantage of using motion data for prediction is that it has a very low energy profile as mobile devices require minimum energy to capture their values. Its predictive strength concentrates on activities with high body movement patterns such as running, walking and cycling. On the other hand, the network often performs poorly for activities where there is no such pattern and/or limited movement. For this reason, we also use a vision predictor that helps to increase accuracy in these scenarios.

3.3. LRCN (Vision) Predictor

CNNs have dominated recent image recognition tasks. However, these models are not very effective on tasks involving sequences of images. Our vision predictor uses a mix of CNNs and RNNs called Long-term Recurrent Convolutional Networks (LRCN) [4]. The model's first layer uses an Inception V3 [36] pre-trained on Imagenet [11] followed by a LSTM with 512 neurons and a softmax on the last layer. The CNN acts as a feature extractor while the LSTM captures the temporal structure of the data. All training happens end-to-end where we first freeze the inception layers and train only the LSTM for 10 epochs. Then, we unfreeze the 3 last blocks of the CNN and train again for 20 more epochs. The model receives a sequence of images and outputs a vector of activity probabilities. Even though the accuracy of vision methods have shown higher overall accuracy on previous works [31, 33, 32], it still is very energy inefficient model. For instance, we evaluated the camera's consumption on a Vuzix M300 smartglasses, and it takes on average three times more energy than only motion sensor measurements. Our aim is to optimize the overall accuracy by balancing our predictors usage appropriately.

3.4. Reinforcement Learning

Until recently, RL methods were constrained to discrete state spaces and actions. However, the use of deep networks as function approximators have extended their use to continuous inputs and outputs. While in supervised learning the main goal is to only map inputs to outputs, in RL the choice of what to approximate is what makes current methods to differ from each other. One could optimize policies, value functions, dynamic models or some combination thereof. In fact, there are mainly two perpendicular choices to be made: what kind of objective to optimize (e.g. policy, value function or dynamics) and what kind of function approximators to use. In these lines, our framework can be fully defined by the tuple $< S, O, A, r, \lambda, \gamma >$, with:

- S: Set of states $\{s_1, ..., s_n\}$ where $s_t := (o_t, y_t)$ where,
  o_t: Sensor observation at time-step t,
  y_t: True activity label at time-step t.
- O: Set of observations $\{o_1, ..., o_n\}$ where $o_t := (acc_t, gyr_t)$ and,
  acc_t: 3D accelerometer values at time-step t,
  gyr_t: 3D gyroscope values at time-step t.

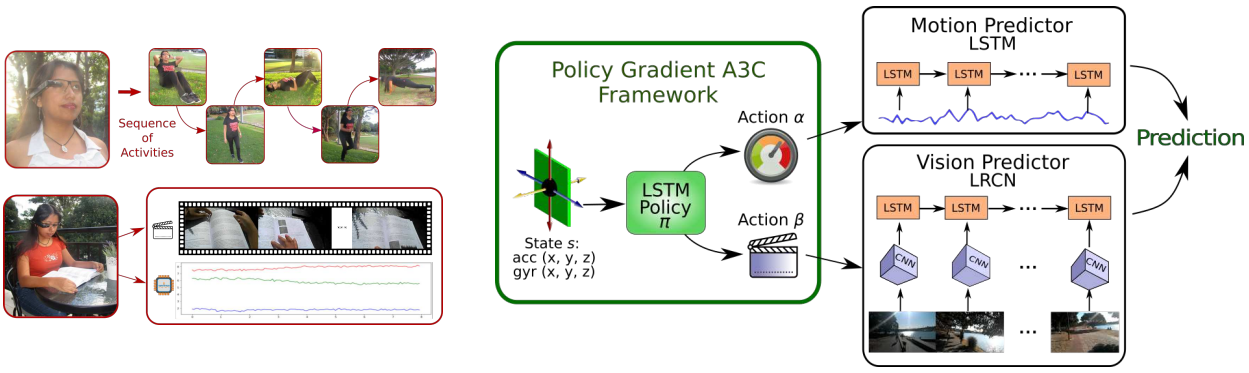

Figure 1: **(Left)**: Our smartglasses capture data from different sources; motion sensors and camera. **(Middle)**: Policy Function approximator represented by a LSTM Neural Network. **(Right)**: Activity predictors pre-trained on an activity recognition dataset.

- A: Set of actions $\{a_1, ..., a_n\}$ where $a_t \in [\alpha, \beta]$ where,
  α: Motion (LSTM) is used for prediction,
  β: Vision (LRCN) is used for prediction.
- r: $S \times A \rightarrow \{1+\lambda, -1-\lambda, 1, -1, \lambda\}$ is a reward function with five possible outputs, corresponding to executing an action a_t in state s_t.
- $\lambda \in [0, 1]$: Parameter that limits the rewards given to high energy consumption actions.
- $\gamma \in (0, 1]$: is the discount factor on future rewards.

Further, we denote the LSTM policy network by $\pi_\theta(a_t|s_t)$ which is parametrized by θ. Through the RL framework, we select actions that give the highest average reward for every hidden representation of the input state s_t. The network architecture is composed of a layer with 256 neurons followed by a softmax with one neuron for each action. The policy $\pi_\theta(a_t|s_t)$ chooses between two different actions: predict y_t (the activity label) from the motion predictor $\rho^m(x_t)$ or predict y_t from the vision predictor $\rho^v(x_t)$. Since our state is partially represented by the accelerometer and gyroscope readings, these are required to be turned on in the device at all times.

Reward Function: The design of the reward function is one of the most important parts of any RL framework. Its outputs should be able to increase or decrease the likelihood of selecting an action in order to provide a better behavior for the entire system. Positive rewards increases the probability of an action through gradient ascent, while negative rewards will decrease it through gradient descent. In our case, we have a lower accuracy but more energy efficient motion predictor and a more energy demanding/accurate vision predictor. The goal is to only use vision methods when low-energy motion predictions are incapable of providing the right outcome. In order to determine the reward, we use a training dataset where we have access to the correct labels for all given inputs. We evaluate the results on both motion and vision predictors calculating the advantage of choosing one predictor over the other and updating our policy accordingly. For instance, we give only λ reward for choosing the LRCN model for prediction using vision when we observe that the LSTM is also able to provide a correct outcome using motion sensors. The function gives constant rewards $r_t \in [-1, 1]$ for choosing the motion predictor while giving λ controlled rewards to vision predictor actions. The formal definition of the reward $r_t(s, a)$ is as follows,

$$\begin{cases} 1 & \pi_\theta(a_t|s_t) = \alpha,\ \rho^m(x_t) = y_t \\ 1+\lambda & \pi_\theta(a_t|s_t) = \beta,\ \rho^m(x_t) \neq y_t,\ \rho^v(x_t) = y_t \\ \lambda & \pi_\theta(a_t|s_t) = \beta,\ \rho^m(x_t) = y_t,\ \rho^v(x_t) = y_t \\ -1-\lambda & \pi_\theta(a_t|s_t) = \beta,\ \rho^m(x_t) = y_t,\ \rho^v(x_t) \neq y_t \\ -1 & \text{otherwise.} \end{cases}$$

Episodes and Steps: Each episode contains 15 seconds of data split in equal time-steps t of 1 second each. Videos on the Multimodal dataset have 1 single activity per 15 seconds of footage while on DataEgo, a video has 5 minutes of duration conformed by 4-6 activities. Actions are taken every second using the readings from both accelerometer and gyroscope. A separate buffer of image is kept during training in order to evaluate the reward function. On real life settings the camera would need to be turned on in order to provide data for the LRCN model.

3.5. Policy Learning

We solve the RL problem through the use of a model free framework. The goal is to optimize a parametrized stochastic policy $\pi_\theta(a_t|s_t)$ and a value function $V_{\theta_v}(s_t)$ using gradient methods [35]. While policy methods learn a policy directly, value iteration methods such as Q-Learning focuses on updating state-value functions using the Bellman equation [34]. Our method uses an Actor-Critic [10] framework that combines the benefits of both approaches. The actor

takes actions based on a policy $\pi_\theta(a_t|s_t)$ and an estimate of the value function (critic) $V_{\theta_v}(s_t)$. The value function $V_{\theta_v}(s_t)$ determines how good a certain state is while following a policy π_θ. While actor and critic parameters θ and θ_v are shown as being separate, we share some of the parameters in practice to improve stability. We use our LSTM policy network with a softmax output for the policy $\pi_\theta(a_t|s_t)$ and one linear output for the value function $V_{\theta_v}(s_t)$.

Algorithm 1 Asynchronous advantage actor-critic (A3C)

//global shared parameter vectors = θ and θ_v
//thread-specific parameter vectors = θ' and θ'_v
Inputs: π_θ, D
Outputs: π^*_θ
$t \leftarrow 1$
repeat
  $d\theta, d\theta_v, \leftarrow 0$
  $\theta' \leftarrow \theta$
  $\theta'_v \leftarrow \theta_v$
  $t_{start} \leftarrow t$
  repeat *//run episode and save batches until t_{max}*
    $a_t \leftarrow \pi_{\theta'}(a_t|s_t)$
    Caculate $r_t(s_t, a_t)$
    $s_t \leftarrow s_{t+1}$
    $t \leftarrow t + 1$
    $T \leftarrow T + 1$
  until terminal s_t or $t - t_{start} == t_{max}$
  if $s_t == terminal$, $R = 0$ **else** $R = V_{\theta_v}(s_t)$
  for $i \in \{t-1, ..., t_{start}\}$ **do**
    $R \leftarrow r_i + \gamma R$
    $dg \leftarrow \nabla_{\theta'} \log \pi_{\theta'}(a_t|s_t) A(s_t, a_t, \theta'_v)$
    $dh \leftarrow \nabla_{\theta'} H(\pi_{\theta'}(a_t|s_t))$
    $d\theta \leftarrow d\theta' + dg + dh$
    $d\theta_v \leftarrow d\theta'_v + \partial(R - V_{\theta'_v}(s_t))^2/\partial\theta'_v$
  end for
  $\theta \leftarrow \theta + d\theta'$ //asynchronous update
  $\theta_v \leftarrow \theta_v + d\theta'_v$ //asynchronous update
until $T > T_{max}$

Policy learning is performed through infinitesimal updates in both θ and θ_v. The sign and magnitude of our reward determines if we are making an action more or less probable as it performs gradient ascent and descent respectively. For the policy update, we calculate the gradient over an expectation using a *score function* estimator as shown in the work of Sutton et al. [35]. The value function is updated using a squared loss between the discounted reward and the estimate of the value under parameters θ_v. Optimization is a two-step process where we first train our predictors $\rho^m(x_t)$ and $\rho^v(x_t)$ on the training dataset and then we use their predictions to optimize both the policy θ parameters and value function θ_v parameters.

The main benefit of the actor-critic method is to use an advantage function instead of discounted rewards in the update rule. As rewards have high variance during learning, the use of an estimated value speeds up the process while reducing variance on updates. The advantage function $A(s_t, a_t, \theta_v)$ is an estimate of the advantage and is given by $A(s_t, a_t, \theta_v) = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V_{\theta_v}(s_{t+k}) - V_{\theta_v}(s_t)$.

We also added entropy regularization to our policy updates as proposed by Williams and Peng [39]. The idea is to have a term in the objective function that discourages premature convergence to suboptimal deterministic policies. The final update rule for the algorithm takes the form $\theta \leftarrow \theta + \nabla_\theta(\log\ \pi_\theta(a_t|s_t))\ A(s_t, a_t, \theta_v) + \ \eta\ \nabla_\theta H(\pi_\theta(a_t|s_t))$, where $H(\pi_\theta(a_t|s_t))$ is the entropy for our policy and η is the parameter that controls its relative importance.

3.6. Asynchronous optimization

The availability of multi-core processors justifies the development of RL techniques with asynchronous updates. Recent work [22, 29] have shown that updates in on-line methods are strongly correlated mainly because the data observed from RL agents is non-stationary [20]. Techniques like experience replay [22] focuses on storing batches of experience and then performing gradient updates with random samples from each batch. Here, we follow a different approach to solve the same problem. We use asynchronous gradient optimization of our controllers that executes multiple workers in parallel on multiple instances of the environment. This process decorrelates the data into a more stationary process. In fact, this simple idea enables a much larger spectrum of RL algorithms to be executed effectively. The asynchronous variant of actor-critic methods is the state-of-the-art method on several RL complex domains as it was shown by Mnih et al (2016). Algorithm 1 shows the implementation for each actor-learner, which we call Asynchronous advantage actor-critic (A3C). They run independently on each CPU core while the central model receives updates from all workers asynchronously.

4. Experiments

4.1. Datasets

The large majority of previous work on egocentric Activity Recognition have used either raw data acquired from sensors or video data from cameras (not both). One of the few datasets available was proposed by Song et al. [33]. We refer to this dataset as Multimodal. Its main limitation is that videos are split by activity instead of a more natural setting where there is a flow between different activities.

We present a novel egocentric activity dataset DataEgo that contains a very natural set of activities developed in a wide range of scenarios. There are 20 activities performed in different conditions and by different subjects. Each recording has 5 minutes of footage and contains a se-

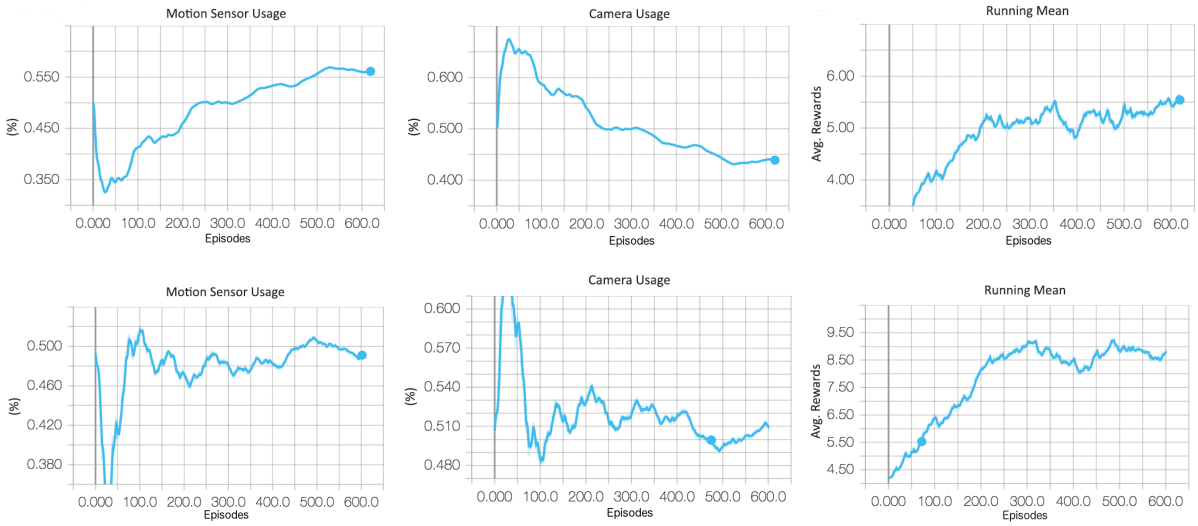

Figure 2: Convergence of A3C shows small variance on motion/vision usage and average rewards after 600 episodes for both Multimodal (**top**) and DataEgo (**bottom**) datasets.

quence of 4-6 different activities. Images from the camera are synchronized with readings from the accelerometer and gyroscope captured at 15 fps and 15 Hz respectively. In total, our dataset contains approximately 4 hours of continuous activity while the multimodal dataset has only 50 minutes of separate activities. We make DataEgo publicly available in the following link [1].

4.2. Predictors Benchmark

The results for our individual predictors are shown on Table 1. It can be seen that if we consider methods individually without any type of sensor fusion or extra features such as the ones from optical flow, our methods have the highest overall accuracy. Our LRCN network has achieved an accuracy of 78.70% which sits very close to the more resource intensive methods from previous work [32, 33]. Our LSTM also outperforms previous work on motion sensors [32] by almost 10%. We believe that this result is due to the stateful approach we used during training. The idea is to save the hidden states in between batches so as to better capture the temporal structure within the data.

4.3. Convergence of A3C

Training results of the RL framework are shown on Figure 2. Convergence was achieved with approximately 600 episodes for each of the actor-learners. The running mean of rewards presents an exponential increase initially while stabilizing with fixed small variance at the end. The results illustrates that our algorithm finds a stable policy for λ = 0.2 while equally balancing usage of low/high energy consumption predictors.

[1]Dataset link: http://sheilacaceres.com/dataego/

Method	**Dataset**	**Accuracy** (%)
LRCN (vision)	Multimodal	78.70%
LSTM (motion)	Multimodal	61.24%
LRCN (vision)	DataEgo	71%
LSTM (motion)	DataEgo	58%
CNN FBF (vision) [32]	Multimodal	70%
Multi Max (both) [32]	Multimodal	80.5%
Fisher Vector (both) [33]	Multimodal	83.7%
LSTM (motion) [32]	Multimodal	49%

Table 1: Comparison of motion and vision predictors with previous work shows higher accuracy when comparing to single stream methods.

4.4. Motion vs Vision Tradeoff

Figure 3 compares the effect of λ on the overall per class results for the Multimodal dataset. Activities such as organizing files, riding elevators and others have greatly improved their accuracy by using the vision predictor. This shows that our policy is in fact learning actions that exploits the different strengths of our predictors.

Validation for the aforementioned results was perfomed through an analysis of how actions were being chosen amongst different activities. We sampled the softmax outputs on the multimodal test dataset while using a learned policy with λ = 0.2. As can be seen on Figure 4, activities such as organizing files and riding elevators/escalators presented higher probabilities on using the vision predictor, while running, doing sit-ups and walking up/downstairs were dominated by the motion predictor. This fact is con-

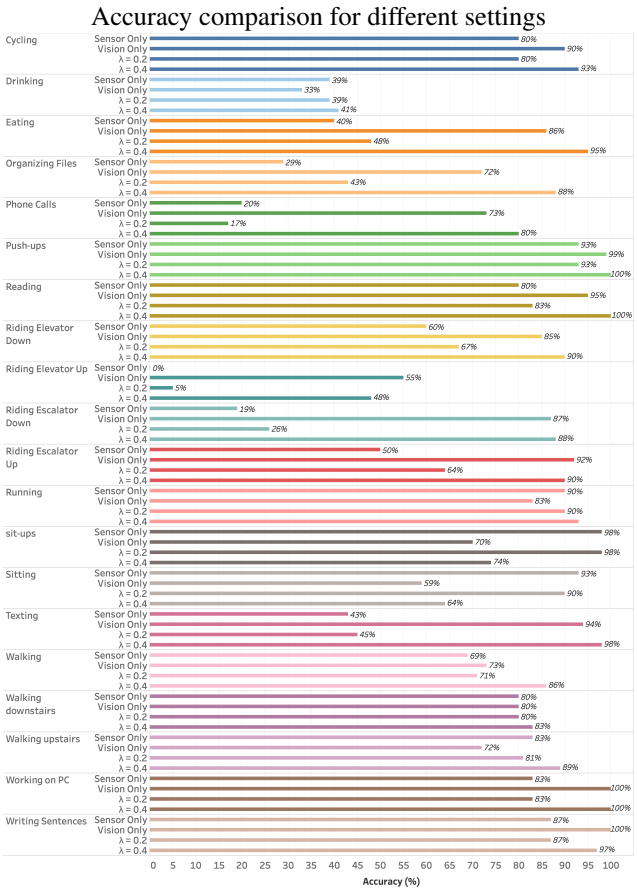

Figure 3: Individual per-class accuracy on Multimodal dataset shows that activities with low body movement seems to benefit the most from vision predictor.

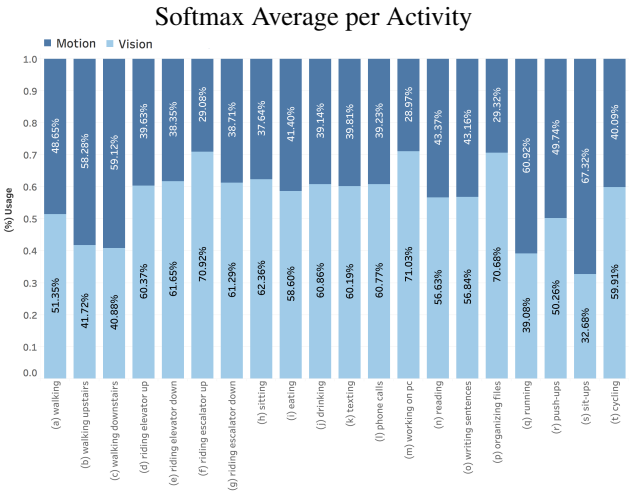

Figure 4: The softmax average per activity shows synergy between predictors on a $\lambda = 0.2$ policy.

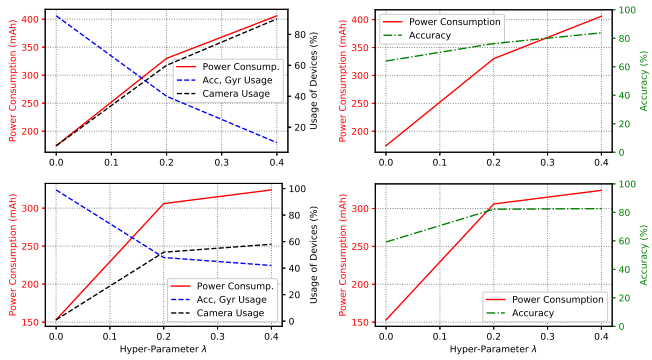

Figure 5: Summary of Accuracy, Energy Consumption and Sensor/Vision usage for both Multimodal (**top**) and DataEgo (**bottom**) datasets.

sistent with a real life setting as activities with higher movement patterns are prone to perform better with motion sensor data.

Figure 5 shows the energy consumption trend as we increase the use of vision methods. A vision only approach would have an energy cost of 450 mAh, while motion based only costs 150 mAh. As it was mentioned before, motion methods are three times more energy efficient than vision methods. For both datasets, $\lambda = 0.2$ seems to be the best tradeoff between accuracy/energy consumption.

Maximum energy efficiency was achieved allowing minimum use of the vision predictor by making $\lambda = 0$. With only 8% of vision usage the model achieved 64.02% accuracy on the Multimodal dataset as shown on Figure 5. This represents an increase of almost 4% if we compare with the motion only predictor. The benefit is that we tradeoff minimum vision usage for a huge leap in accuracy. This is only possible due to the optimal decision making behavior learned by our policy.

The parameter λ allows our method to learn policies with both low energy profile and with high predictive capacity. This can be achieved through proper parameter tunning. For instance, predictive performance is improved as we increase the value of λ while energy consumption is reduced as we pick smaller values.

The results on large values of λ have shown promising results. With 95% of vision usage and only 5% of motion, the overall accuracy on the Multimodal dataset was 84.84%. This is the highest accuracy achieved in this dataset to date. The policy outperforms the vision only model by almost 5%. Not only we outperform previous work [32] but we also provide a more energy efficient predictor as 5% of all actions comes from a low energy source. It is worth to note that the benchmarked methods require the camera to be on at all times as they rely on its data to make continuous predictions.

Precision and Recall on the DataEgo dataset with different λ values showing Motion / Vision Usage

	$\lambda = 0$ (99% / 1%)		$\lambda = 0.2$ (48% / 52%)		$\lambda = 0.4$ (42 % / 58%)		Max Pooling [32]	
Activity	Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall
Walking	**0.79**	0.84	0.73	**0.88**	0.73	**0.88**	0.68	0.75
Walking Up/downstairs	**0.74**	**0.35**	0.39	0.27	0.67	0.29	0.64	**0.35**
Chopping Food	0.026	0.017	**1.0**	**0.85**	1.0	0.83	0.88	0.80
Riding Elevators	0.18	0.0434	0.42	0.13	0.54	0.17	**0.58**	**0.20**
Brushing Teeth	0.05	0.073	0.14	**1.0**	0.14	**1.0**	**0.20**	0.99
Riding Escalators	0.08	0.03	0.62	0.41	0.71	0.52	**0.78**	**0.60**
Talking with people	0.46	0.70	0.46	0.70	**0.5**	**0.71**	**0.5**	0.70
Watching TV	0.29	0.33	**0.29**	**0.33**	0.28	**0.33**	0.22	0.28
Eating and Drinking	0.49	0.61	0.82	0.97	0.87	**0.98**	**0.88**	0.94
Cooking on Stove	0.19	0.16	**0.94**	**0.94**	0.90	**0.94**	0.90	0.91
Browsing Mobile Phone	0.16	0.28	**0.96**	0.91	0.94	0.92	0.95	**0.97**
Washing dishes	0.21	0.78	**0.81**	0.94	0.76	**1.0**	0.80	1.0
Working on PC	0.30	0.5	0.82	0.58	**0.86**	0.69	0.84	**0.72**
Reading	0.26	0.10	**0.95**	0.85	**0.95**	**0.87**	0.90	0.78
Writing	0.28	0.26	**0.96**	**0.93**	0.90	0.89	0.83	0.83
Lying Down	**0.98**	**0.96**	0.91	**0.96**	0.91	**0.96**	0.76	0.50
Running	0.96	0.92	0.96	0.90	0.94	0.84	**0.97**	**1.0**
Doing push ups	0.69	0.79	0.97	0.89	0.97	0.88	**0.99**	**0.96**
Doing sit ups	0.89	0.82	0.88	0.82	0.88	0.81	**1.0**	**1.0**
Cycling	0.69	0.81	**0.78**	**0.87**	0.64	0.82	0.75	0.83

Table 2: Results on the DataEgo dataset shows that our method has better performance when compared to previous work.

The impact of λ was quite different between datasets as can be observed on Figure 5. This is due to a lower predictive capacity of the LRCN on the DataEgo dataset. As the difference of accuracy between predictors decreases, our method requires smaller values of λ in order to balance their usage. Therefore, when choosing a value of λ, the predictors accuracy should be taken into consideration as it is directly related to the magnitude of λ. Moreover, deterministic policies were in fact learned when λ values were too high or too low. For instance, λ = 0 provided 99% of motion based predictions usage and only 1% of vision based ones while using the Multimodal dataset.

Table 2 compares the max pooling method from previous work [32] on the DataEgo dataset. The dataset presents a more complicated problem as now the model needs to also learn activity transitions. Benchmarked methods performance is worse than ours with λ = 0.4. This shows that our framework is more suitable on situations where a more realistic setting is used. Our policy seems to suffer less from activity transition noise as it provides better overall accuracy.

5. Conclusion

We presented a novel RL framework for the Egocentric Activity Recognition problem. Our method was able to achieve state-of-the-art performance on current multistream datasets while saving energy by trading off vision-based activity recognition with low power motion based sensor. This approach attempts to be more realistic for practical implementation on wearable devices as these devices have limited computational and energy resources and therefore are not able to not only run expensive deep learning models but also keep the device's camera on for extensive periods of time.

We believe that in order to create more pervasive machine learning applications, resource optimization should be an important consideration to make. As wearables usually suffer from low energy availability, optimizing its use seems to be a good approach to enable a better use of state of the art deep learning techniques in these devices.

References

[1] N. Ballas, L. Yao, C. Pal, and A. Courville. Delving Deeper into Convolutional Networks for Learning Video Representations. pages 1–11, 2016. 1, 2

[2] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive Computing*, 2004. 1, 2

[3] Y. Bengio. Deep learning of representations: looking forward. In *Statistical Language and Speech Processing*, volume 7978 of *Lecture Notes in Computer Science*, pages 1–37. Springer, 2013. 1

[4] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015. 1, 2, 3

[5] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[6] P. J. Hsieh, Y. L. Lin, Y. H. Chen, and W. Hsu. Egocentric activity recognition by leveraging multiple mid-level representations. In *2016 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2016. 1, 2

[7] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, Jan 2013. 2

[8] S. Karayev, M. Fritz, and T. Darrell. Anytime recognition of objects and scenes. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, 2014. 2

[9] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-Scale Video Classification with Convolutional Neural Networks. *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1725–1732, 2014. 1, 2

[10] V. R. Konda and J. N. Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000. 4

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. 2, 3

[12] N. D. Lane, P. Georgiev, and L. Qendro. Deepear: Robust smartphone audio sensing in unconstrained acoustic environments using deep learning. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '15, pages 283–294, New York, NY, USA, 2015. ACM. 1, 2

[13] J. Lee and J. Kim. Energy-efficient real-time human activity recognition on smart mobile devices. 2016:1–12, 07 2016. 1

[14] X. Li, Y. Zhang, I. Marsic, A. Sarcevic, and R. S. Burd. Deep learning for rfid-based activity recognition. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems*, SenSys '16, pages 164–175, New York, NY, USA, 2016. ACM. 1, 2

[15] Y. Li, A. Fathi, and J. M. Rehg. Learning to predict gaze in egocentric video. In *2013 IEEE International Conference on Computer Vision*, pages 3216–3223, Dec 2013. 1, 2

[16] Y. Li, Z. Ye, and J. M. Rehg. Delving into egocentric actions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 287–295, June 2015. 1, 2

[17] M. Ma, H. Fan, and K. M. Kitani. Going deeper into first-person activity recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2

[18] S. Mathe, A. Pirinen, and C. Sminchisescu. Reinforcement learning for visual object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2894–2902, 2016. 2

[19] K. Matsuo, K. Yamada, S. Ueno, and S. Naito. An attention-based activity recognition for egocentric video. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 565–570, June 2014. 1, 2

[20] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016. 5

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 2

[22] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 5

[23] P. Morerio, L. Marcenaro, and C. S. Regazzoni. Hand detection in first person vision. In *Proceedings of the 16th International Conference on Information Fusion*, pages 1502–1507, July 2013. 1, 2

[24] J. Y. H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June-2015:4694–4702, 2015. 1, 2

[25] K. Ogaki, K. M. Kitani, Y. Sugano, and Y. Sato. Coupling eye-motion and ego-motion features for first-person activity recognition. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–7, June 2012. 2

[26] H. Pirsiavash and D. Ramanan. Detecting activities of daily living in first-person camera views. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012. 1, 2

[27] M. Riedmiller, T. Gabel, R. Hafner, and S. Lange. Reinforcement learning for robot soccer. *Autonomous Robots*, 27(1):55–73, 2009. 2

[28] M. S. Ryoo, B. Rothrock, and L. Matthies. Pooled motion features for first-person videos. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 896–904, June 2015. 2

[29] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *Proceedings of the 32nd*

International Conference on Machine Learning (ICML-15), pages 1889–1897, 2015. 5

[30] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'14, pages 568–576, Cambridge, MA, USA, 2014. MIT Press. 1, 2

[31] S. Song, V. Chandrasekhar, N.-M. Cheung, S. Narayan, L. Li, and J.-H. Lim. Activity recognition in egocentric life-logging videos. In C. V. Jawahar and S. Shan, editors, *Computer Vision - ACCV 2014 Workshops*, volume 9010 of *Lecture Notes in Computer Science*, pages 445–458. Springer International Publishing, 2015. 3

[32] S. Song, V. Chandrasekhar, B. Mandal, L. Li, J. H. Lim, G. S. Babu, P. P. San, and N. M. Cheung. Multimodal multi-stream deep learning for egocentric activity recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016. 1, 2, 3, 6, 7, 8

[33] S. Song, N. M. Cheung, V. Chandrasekhar, B. Mandal, and J. Liri. Egocentric activity recognition with multimodal fisher vector. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, volume 2016-May, pages 2717–2721, 2016. 1, 3, 5, 6

[34] R. S. Sutton and A. G. Barto. *Reinforcement Learning : An Introduction*. MIT Press, 1998. 4

[35] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000. 4, 5

[36] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,*, 2016. 3

[37] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *Int. J. Comput. Vis.*, 103(1):60–79, mai 2013. 1, 2

[38] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *Computer Vision – ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII*, volume 9912, pages 20–36, Cham, 2016. Springer. 2

[39] R. J. Williams and J. Peng. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991. 5

[40] J. Wu, A. Osuntogun, T. Choudhury, M. Philipose, and J. M. Rehg. A scalable approach to activity recognition based on object use. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Oct. 2007. 1, 2

[41] Z. Yan, V. Subbaraju, D. Chakraborty, A. Misra, and K. Aberer. Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach. In *2012 16th International Symposium on Wearable Computers*, pages 17–24, June 2012. 1

[42] K. Zhan, S. Faux, and F. Ramos. Multi-scale conditional random fields for first-person activity recognition. In *Pervasive Computing and Communications (PerCom), 2014 IEEE International Conference on*, pages 51–59, March 2014. 2

[43] X. Zhang, J. Wang, Q. Gao, X. Ma, and H. Wang. Device-free wireless localization and activity recognition with deep learning. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, pages 1–5, March 2016. 1, 2

[44] L. Zheng, D. Wu, X. Ruan, S. Weng, A. Peng, B. Tang, H. Lu, H. Shi, and H. Zheng. A novel energy-efficient approach for human activity recognition. 17:2064, 09 2017. 1