Towards Intelligent Management of Internet of Modern Drones

Supadchaya Puangpontip(✉) and Rattikorn Hewett

Texas Tech University, Lubbock, TX 79409, USA
{supadchaya.puangpontip,rattikorn.hewett}@ttu.edu

Abstract. Internet of drones (IoD) provides coordinated access, between drones and users, over the Internet to controlled airspace. With advanced drone, mobile and Artificial Intelligence (AI) technologies, today's drones are equipped with sophisticated onboard AI software that enhances drone services and our way of life (e.g., package delivery, traffic surveillance). As IoD grows, there is a need to effectively manage large-scaled drones with multiple regulation and resource constraints, particularly energy usage. This paper presents preliminary work on generic architecture and operations to lay foundations for intelligent drone management systems. By also introducing a method to pre-determine estimated energy consumption of deep neural net image analysis deployed in drones, the paper illustrates this work on managing the search rescue drone autonomy to decide on its actions based on energy consumption. The proposed approach can be extended to manage a network of drones and additional resource constraints including response time, safety or environmental compliance and financial budget.

Keywords: IoD · Energy consumption · Deep learning · CNN · Drone

1 Introduction

Internet of Things (IoT) is a key enabler to services that improve quality of our life. Evolving from IoT, Internet of drones (IoD) [1, 2] provides an infrastructure for coordinated access, between drones and users, over the Internet to controlled airspace. With advanced mobile and AI (Artificial Intelligence) technologies, modern drones are more affordable and equipped with sophisticated onboard smart software to enhance drone capabilities (e.g., image recognition with deep neural net learning (DNN), navigation). The applications of IoD are ample from traffic surveillance to data collection, to package delivery to disaster mitigation and rescue [1, 3]. There is an increasing need to effectively manage large-scaled drones with multiple resource (e.g., time, airspace, cost) and regulation (e.g., environmental, safety) constraints, particularly energy usage. Without good planning and energy management, the drone can exhaust its battery before it gets to the destination or finish the mission.

Recent research in IoD deals with energy-efficient solutions e.g., optimizing transmission power or flight time [4–6], harvesting energy or managing charging stations [3] to increase the drone's operating time. Little work has been done on energy management

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
F. Neri et al. (Eds.): CCCE 2022, CCIS 1630, pp. 3–13, 2022.
https://doi.org/10.1007/978-3-031-17422-3_1

of the drones or energy modeling of the software, such as DNN, deployed in drones [3, 4, 7]. This paper presents preliminary work on generic architecture and operations for intelligent drone management systems. The "intelligence" is contributed by its adaptiveness to dynamically changing conditions and constraints. The paper also proposes a method to estimate energy consumption of DNN image analysis deployed in drones and illustrates its use in managing the rescue drones to recommend appropriate actions based on the remained energy. Unlike previous energy modeling [7], ours gives a method for deriving energy measurements from the DNN configurations.

The rest of the paper is organized as follows. Sections 2 and 3 are our main contributions of the proposed framework and drone energy models, respectively. The paper illustrates and experiments on rescue drones in Sect. 4 and concludes in Sect. 5.

2 The Proposed Framework for Intelligent Drone Management

Consider a system of modern drones of various sizes, functions (e.g., capture image or video, fly) and capabilities. For example, some are equipped with intelligent software on board (e.g., to detect certain objects from an image captured by the drone, navigate, or decide whether to fly back to the base or continue the excursion). Some may be able to fly at high speed but for a short duration, whereas some may have a long battery life but can only fly at medium to low speed. Our objective is to develop a framework for building an intelligent system (e.g., adaptive to changing and uncertain situations) to assist management of drones on a particular mission (e.g., drug delivery in rural area, surveillance, or rescue search). In this paper, we focus on managing the drone autonomy on its actions based on limited resource (i.e., energy). To convey the idea clearly, we pick a case scenario of using drones to help find victims from a disaster (e.g., flood or wildfires). Although we have not done this, we conjecture that the proposed approach can be extended to manage a team of drones and additional resource constraints (e.g., time, safety or environmental compliance and budget).

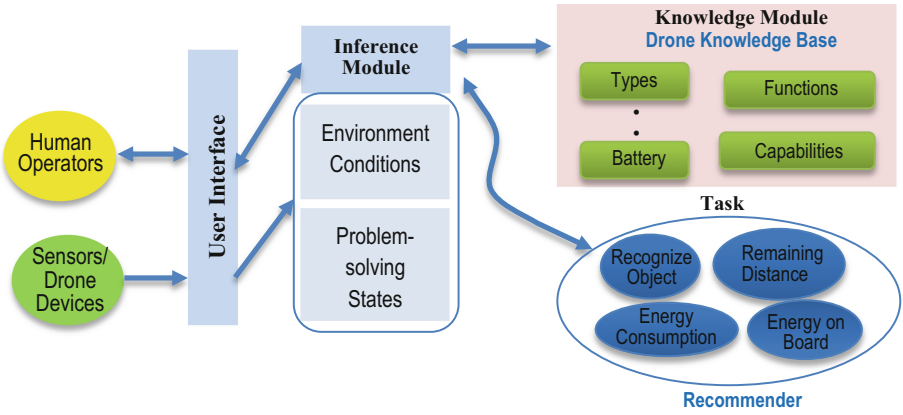

Fig. 1. An overall architecture of drone management framework.

We will refer to the drone management system employing the proposed architecture as RESCUER. Figure 1 shows an overview of RESCUER's proposed architecture that consists of three basic components*: Inference Module*, *Knowledge Module* and *Task Module* (e.g., *Recommender*). The architecture is data/event driven where the data are transmitted from external environments including sensors (e.g., temperature, weather/wind condition), human operators/users, and drone devices (e.g., image or video), as shown to the left of Fig. 1. The data interact with RESCUER via its User Interface to a reasoning module during problem solving. Each component is described below.

Inference Module: Includes an inference engine that provides basic reasoning mechanisms (e.g., forward or back-ward inferences) and a working module to maintain problem-solving states of reasoning tasks. In each reasoning cycle, recent each event/data triggers applicable reasoning operations from various tasks and select the most appropriate one for execution, which in turn would produce new events initiating a new reasoning cycle. Control mechanisms to select rules to act can use priorities, or meta-control rules [8].

Knowledge Module: Represents information about drones including type specifications and models, functions and requirements (e.g., transmission), capabilities, and battery life. These are used by the task module to assist drones to take appropriate actions. Note that some of the knowledge can be generic (e.g., physics of drone flying) and some are specific to task applications.

Task Module: Represents several computing tasks required or influence the recommendation to the drone. For example, by "*recognize object*", the drone analyzes the image captured (using deep learning technology to recognize a target that it is searching). If the result is positive, then it will send the image to RESCUER to send a team to rescue the victim. This helps the drone to save transmission energy from transmitting all images captured. The "energy consumption" quantifies energy usage so far, while the "energy on board" calculates remaining energy that the drone has. Together with the "remaining distance" to travel, the recommender can recommend the drone to 1) fly back (if not enough energy, or environmental condition is not safe), or 2) continue the search (if enough energy and more area to search, 3) transmit the image that detects a target victim, and 4) alarm and increase speed when in danger.

Note that the proposed framework is general in that although the framework is applied to drone rescuers, it is also applicable to other IoT application domains.

3 Energy Estimation of Drone's Activities

This section describes a method to estimate energy consumption of different activities of the drone. Drone actions (e.g., analyze image, transmit image, go back, continue search, alarm) are driven by its awareness of remaining energy and the mission. The total consumption of the drone is the summation of the consumptions of all activities. Table 1 summarizes all but one of basic drone functions have estimated energy usage based on published work [4, 9].

Table 1. Energy consumption of some basic functions of drones.

Drone functions	Energy consumption	Variable descriptions
Hovering [4]	$((mg)^3/2\pi r_p^2 n_p \rho)^{1/2} t$	m: mass of the drone, g: gravitational acceleration, r_p: radius of the propellers, n_p: number of the propellers, and ρ: air density
Transition [4] (Flying)	$\sum_{i=1}^{n-1} (p_{full}/v_{full}) d_{i,i+1}$	p_{full}: hardware power of the drone at full speed, v_{full}: full speed, n: #locations, $d_{i,i+1}$ distance from location i to $i+1$
Transmission [10]	$(\eta p + p_s)(s/(b \log(1 + (pG/N_0 b))$	η: coeff. of transmission power, p: transmission power, p_s power of the drone when no transmission, s: data size to transmit, b: bandwidth, G: channel power gain between the drone and the gateway, N_0: power spectral density

3.1 Hovering Energy

The hovering energy (E_{hov}) is the energy the drone consumes while remaining stationary in the air [11]. It depends on hovering power p_{hov} and transmission time t, shown in (1).

$$E_{hov} = p_{hov} t \tag{1}$$

$$p_{hov} = ((mg)^3/2\pi r_p^2 n_p \rho)^{1/2} \tag{2}$$

The hovering power can be obtained by using (2) where m is the mass of the drone, g is the gravitational acceleration, r_p is the radius of the propellers, n_p is number of the propellers, and ρ is the air density [4, 11].

3.2 Transition Energy

The transition (or flying; E_{fly}) energy is the energy the drone consumes during moving from one location to another. This can be obtained from hardware power of the drone p_{full} when moving at the full speed v_{full} and distance $d_{i,i+1}$ between the location i and location $i+1$ [4]. Given n locations, the transition energy can be found using (3).

$$E_{fly} = \sum_{i=1}^{n-1} (p_{full}/v_{full}) d_{i,i+1} \tag{3}$$

3.3 Transmission Energy

Transmission energy (E_{trans}) of the drone depends on transmission power p and transmission time t [10]. The latter can be obtained by dividing the total number of bits to be transmitted s by the transmission rate r. This gives,

$$E_{trans} = pt = p(s/r) \tag{4}$$

The transmission rate r can be obtained from (5) where b is bandwidth, p is transmission power, h is channel power gain between the drone and the IoD gateway and N_0 is power spectral density of the Gaussian noise [6, 9, 10, 12].

$$r = b \log(1 + (pG/N_0 b)) \tag{5}$$

3.4 Software Energy

While estimating software energy can be easily done via electrical power measurement, when designing a system with limited resource, the ability to pre-estimate energy consumption without having to run them can be very useful.

Energy consumption of software is from computation (E_{comp}) and data movement (E_{data}). E_{comp} is αn_{MACs}, for a constant α and n_{MACs} number of MAC (*multiply-and-accumulate*) operations [7, 13]. For example, to compute $p = \sum_{i=0}^{n} w_i x_i$, each iteration i takes one MAC operation. With n iterations, it takes n MACs. On the other hand, one MAC requires three data reads (for w_i, x_i and p_{i-1}) and one write (i.e., new partial result p_i). Suppose data moves between two memory levels: *cache* and *DRAM* with a cache hit rate h. Data are first looked up in the cache and if they are not found (cache miss), they will be fetched from DRAM and store in cache. As a result, we can obtain data movement energy to be: $E_{data} = \Sigma_{v \in V} (\beta_{cache} a_v + \beta_{DRAM}(1-h)a_v)p$ where β_m is a *hardware energy cost per data access* in m memory and V is a set of data (e.g., input, output, weight), a_v is the

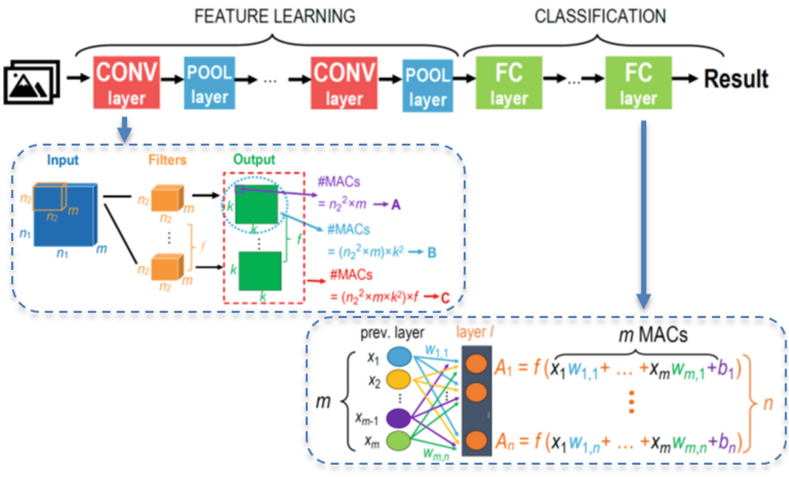

Fig. 2. Computation and associated data in CONV layer (left) and FC layer (right).

number of data accesses for data of type v, and p, precision of data in bits (e.g., 8, 16). Thus, total energy $E_{software}$ is shown in (6). More details are in [14].

$$E_{software} = \alpha\ n_{MACs} + \Sigma_{v \in V} (\beta_{cache} a_v + \beta_{DRAM} (1 - h) a_v) p \tag{6}$$

We propose a method to estimate energy consumption for image analysis that applies a popular trained DNN (Deep Neural Net) model, specifically the Convolutional Neural Net (CNN) architecture, to detect a human (victim) object in the captured image. The proposed energy model is different from our previous model [14]. However, both are based on the same logic. As shown at the top of Fig. 2, the flow of computation in CNN passes through multiple *convolution* (CONV) layers to extract features, *Pooling* (POOL) to reduce the dimensions and *fully connected* (FC) layers for classification. The basic computation of CONV and FC layers are shown at the bottom left and right of Fig. 2, respectively. Due to space limitation, we will omit POOL layer and the derivation of the number of MACs and data accesses for modeling energy in the CONV layer whose details can be found in [4]. Below are energy models with dimensions of input, filter and output which are specified in Fig. 2 (e.g., n_1, n_2, k, f, etc. in CONV).

CONV Layer Energy. First, we find n_{MACs}. As shown in Fig. 2 (bottom left), each CONV layer takes data input of size $n_1 \times n_1 \times m$ and convolves them with each of the f filters of size $n_2 \times n_2 \times m$ to give a final output of size $k \times k \times f$. The convolution process starts with a pointwise multiplication of an input plane with the filter plane of the same dimension (i.e., $n_2 \times n_2 \times m$). This step takes ${n_2}^2$ MACs. Then the results of each plane are summed for each of m channels. Thus, a total number of MACs required for convolving just one cell in this step is ${n_2}^2 m$ (as denoted by A in Fig. 2). Since there are k^2 cells for each filter and f filters, and thus it takes a total of $({n_2}^2 m)(k^2 f)$ MACs (as denoted by B and C as a final count in Fig. 2). However, since each convoluted cell is fed to an activation function whose computation requires $n^{\mathrm{a}}_{\mathrm{MACs}}$ MACs. With this additional computation, the CONV layer requires a total of $({n_2}^2 m + n^{\mathrm{a}}_{\mathrm{MACs}})(k^2 f)$ MACs. Hence, we obtain:

$$n_{MACs} = (n_2^2 m + n^{a}_{\mathrm{MACs}})(k^2 f) \tag{7}$$

Next estimate data movement energy. As shown in Fig. 2 (bottom left), each input data is accessed at least once for each filter requiring ${n_1}^2 m$ MACs. As the filter slides over the input, some of the input data are re-accessed. The number of input data re-accessed is $\sum_{i=2}^{t} c_i i$ where c_i is the number of data that are reused i times and there is at most t reuses. Thus, the number of "input" data accesses is $({n_1}^2 m + \sum_{i=2}^{t} c_i i) \cdot f$ for f filters (**). Similarly, each weight and each bias is accessed once for each output value. Since for one cell and one filter, there are ${n_2}^2 m$ weights and 1 bias, thus, a total "weight" and "bias" accesses become ${n_2}^2 m(fk^2)$ (**) and (fk^2) (**), respectively, for k^2 cells and f filters. Finally, for each of "output" value of ${n_2}^2 m$ being accessed $2{n_2}^2 m$ times (for read and write). Since there are $k^2 f$ output values, thus number of "output" data accesses is $(2{n_2}^2 m)(k^2 f)$ (**). Summing all (**)'s, a total number of data access is derived in (8).

$$a_{CONV} = (n_1^2 m + \sum\nolimits_{i=2}^{t} c_i i + 3 n_2^2 m k^2 + k^2) f \tag{8}$$

By substituting (7) and (8) in (6), a total energy consumption of a CONV layer can be obtained as shown below:

$$E_{CONV} = \alpha(n_2^2 m + n_{\text{MACs}}^a)k^2 f + p[\beta_{cache} a_{CONV} + (1-h)\beta b_{DRAM} a_{CONV}] \quad (9)$$

FC Layer Energy. Each FC layer takes m neurons from previous layer. As shown Fig. 2, for each of n neurons in the FC layer, we compute m weighted sums (argument of an activation function f). Thus, the total number of MACs in the FC layer is shown in (10) where $n_{\text{MACs}}^{\text{a}}$ is the number of MACs used in the activation function (e.g., Sigmoid takes one MAC (for a division) with extra computation γ for exponential function).

$$n_{MACs} = n(m + n_{\text{MACs}}^a) \quad (10)$$

For data movement energy, we consider m input neurons, mn weights, n biases and n output neurons in FC layer (or output layer). As shown in computation of A_i's in Fig. 2, each input x_i is read n times while each weight and each bias are each read once. Thus, a total number of data accesses for input, weight and bias would be mn, mn, and n, respectively. Each output A_i includes read/write accesses of m products, yielding 2 m data accesses. Since there are n output neurons, a total of number of data accesses for output would be 2 mn. As a result, Total number of data accesses is obtained below.

$$a_{FC} = 4\,\text{mn} + n \quad (11)$$

By substituting (10) and (11) in (6), we obtain total energy of the FC layer as shown in (12).

$$E_{FC} = \alpha n(m + n_{\text{MACs}}^{\text{a}}) + p(\beta_{cache}(4\,\text{mn} + n) + \beta_{DRAM}(1-h)(4\,\text{mn} + n)) \quad (12)$$

4 Search and Rescue Drones: Experiments and Results

Consider a scenario of the IoD on a rescue mission, we set up simulations based on the RESCUER management system developed from the framework architecture described in Sect. 2. The IoD has 10 drones, each of which takes 10 images. Comparing two drones, one is a typical drone where all images captured will be transmitted whereas the other is a RESCUER (or smart) drone that analyzes the image and transmits on if victims are detected.

Table 2 shows that the typical drone consumes more transmission energy and has higher total energy consumption. Consequently, it covers less distance and less number of locations. The smart drone, on the other hand, performs better. The ability to select what images to send allows the drone to save energy and spends them on searching. In this particular instance, the transmission energy is reduced by about 70% and the drone can visit 65% more locations. Figure 3 shows average energy breakdowns of 10 smart drones that detect victims (thus, image transmission) randomly with probability of 0.3. On the average, energy consumption is highest by software (or DNN) and lowest by flying. Image analysis task takes more energy on the average than transmission.

Table 2. Typical vs. RESCUER drones.

	Typical	RESCUER
Total Energy (J)	1922.94	1129.97
Transmission Energy (J)	1173.10	349.4
Software (DNN) Energy (J)	0	484.36
Hovering Energy (J)	651.48	194.04
Flying Energy (J)	98.36	102.17

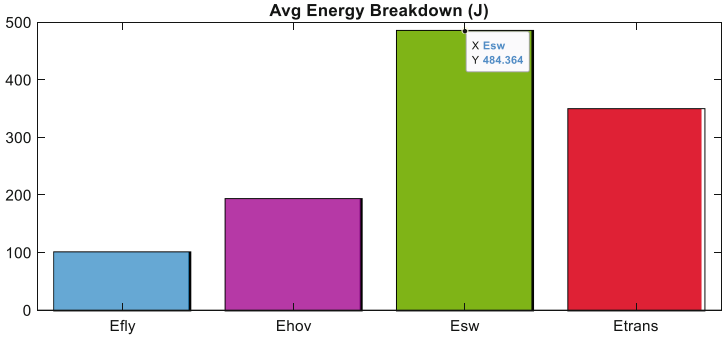

Fig. 3. Average energy breakdown by activities.

In addition, we experiment how varying hop distances will impact energy consumption. As shown in Fig. 4, for 100 m, the drones consume highest energy that gradually decreases for longer hop length. Since the drone is assumed to take pictures, analyze, and possibly transmit the pictures at every location, shorter hop distance means it has to perform these tasks more frequently and marks more locations. As a result, the total energy spent on these tasks is more than on traveling. Reversely, for 500 m, the drone covers more distance and consumes less energy. They however visit, on average, about

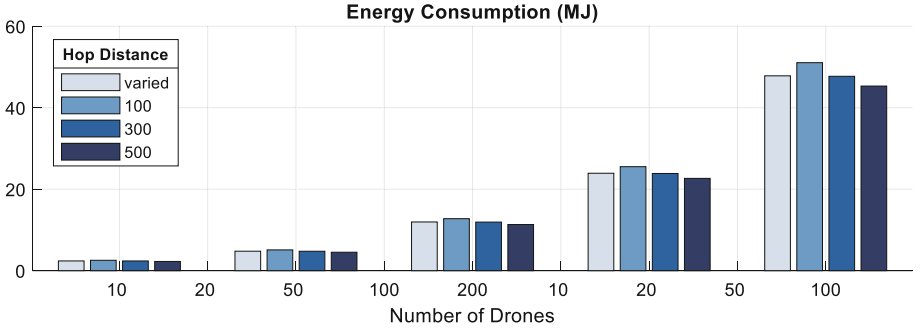

Fig. 4. Effects of varying hop distances.

280 less locations than that of 100 m setting. The randomly varied hop distance consumes about the same energy as those of the 300 m hop distance. This may be because average of the varied distance is close to 300 m.

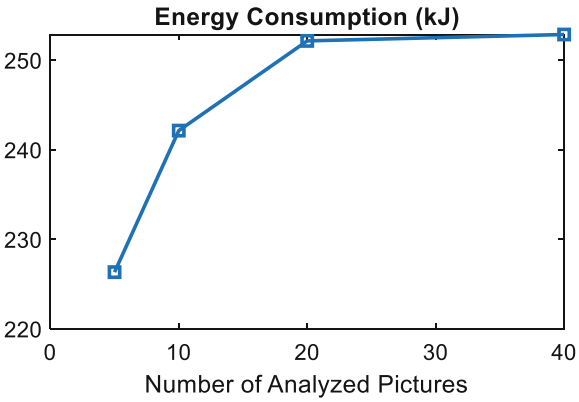

Fig. 5. Image analysis & energy

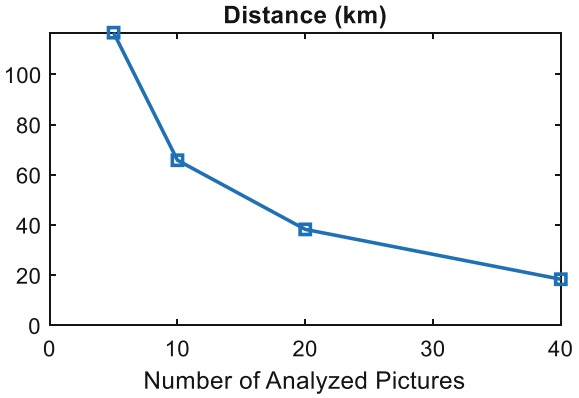

Fig. 6. Image analysis & distance

Figures 5–8 investigates relationships among relevant factors that may impact the mission including the number of images analyzed, distance, bandwidth, transmission rate and energy consumption. Number of analyzed pictures directly affect software energy and transmission energy, since it is the number of times the drone runs a DNN for victim detection and decides whether to transmit the image or not. On the other hand, bandwidth is a network resource which is not always plentiful. It can directly influence the drone's transmission and hovering energy.

As shown in Fig. 5, suppose the drone analyzes 5 images at every location, the total energy is about 230 kJ compared to analyzing 20 images with 250 kJ of energy. The more the images are analyzed, the more energy is consumed as expected. One may argue

at the difference of 20 kJ may not be much, however, the distance covered can reduce significantly.

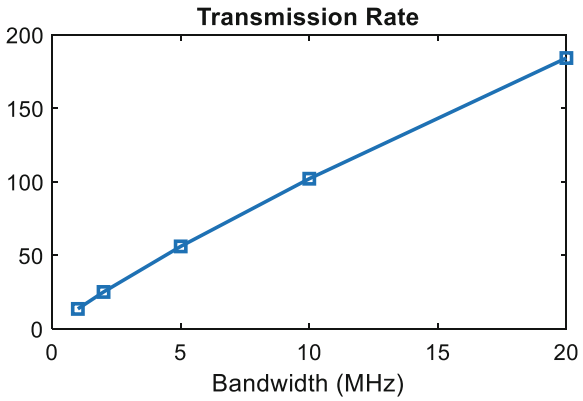

Fig. 7. Bandwidth & transmission.

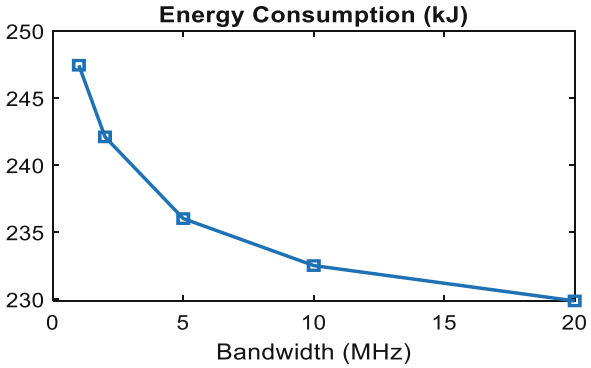

Fig. 8. Bandwidth & energy.

As shown in Fig. 6, the drone can travel about 113, 66, 35 and 18 km, with 5, 10, 20 and 40 analyzed images, respectively. The number of image processing impacts the distance traveled because energy is spent more on software (and transmission), less are spent on the travel. Figure 7 shows that high bandwidth gives high transmission rate. This in turn reduces time to transmit, thus, transmission energy, hovering energy, and the total energy consumption as shown in Fig. 8.

5 Conclusion

This paper presents a generic framework for building intelligent drone management systems. We also present a method for estimating energy consumption of drone functions, particularly the DNN deployed for object recognition in IoD and uses it in our

illustration on RESCUER, an intelligent drone management. By focusing on the drone autonomy, RESCUER recommends appropriate actions to drones based on the remaining energy. Although we use simple scenarios and the experiments are preliminary, the results conform with the variable effects as anticipated. Our future research includes extending the framework to manage a network of drones, incorporating security and privacy aspects, and accounting for additional resource constraints e.g., response time, safety or environmental compliance and financial budget.

References

1. Abualigah, L., Diabat, A., Sumari, P., Gandomi, A.H.: Applications, deployments, and integration of Internet of Drones (IoD): a review. IEEE Sens. J. **21**, 25532–25546 (2021). https://doi.org/10.1109/JSEN.2021.3114266
2. Boccadoro, P., Striccoli, D., Grieco, L.A.: An extensive survey on the Internet of Drones. Ad Hoc Netw. **122**, 102600 (2021). https://doi.org/10.1016/j.adhoc.2021.102600
3. Alsamhi, S.H., Ma, O., Ansari, M.S., Almalki, F.A.: Survey on collaborative smart drones and internet of things for improving smartness of smart cities. IEEE Access **7**, 128125–128152 (2019). https://doi.org/10.1109/ACCESS.2019.2934998
4. Yao, J., Ansari, N.: QoS-aware power control in internet of drones for data collection service. IEEE Trans. Veh. Technol. **68**, 6649–6656 (2019). https://doi.org/10.1109/TVT.2019.2915270
5. Sarkar, S., Khare, S., Totaro, M.W., Kumar, A.: A novel energy aware secure internet of drones design: ESIoD. In: IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 1–6 (2021). https://doi.org/10.1109/INFOCOMWKSHPS51825.2021.9484461
6. Wang, L., Hu, B., Chen, S.: Energy efficient placement of a drone base station for minimum required transmit power. IEEE Wirel. Commun. Lett. **9**, 2010–2014 (2020). https://doi.org/10.1109/LWC.2018.2808957
7. Yang, T.J., Chen, Y.H., Emer, J., Sze, V.: A method to estimate the energy consumption of deep neural networks. In: Conference Record of 51st Asilomar Conference on Signals, Systems and Computers, ACSSC 2017, 2017 October, pp. 1916–1920 (2018). https://doi.org/10.1109/ACSSC.2017.8335698
8. Russell, S., Norvig, P.: AI a modern approach. Learning **2**, 4 (2005)
9. Mozaffari, M., Saad, W., Bennis, M., Debbah, M.: Drone small cells in the clouds: design, deployment and performance analysis. In: 2015 IEEE Global Communications Conference (GLOBECOM), pp. 1–6 (2015). https://doi.org/10.1109/GLOCOM.2015.7417609
10. Auer, G., et al.: How much energy is needed to run a wireless network? IEEE Wirel. Commun. **18**, 40–49 (2011). https://doi.org/10.1109/MWC.2011.6056691
11. Gundlach, J., Gundlach, J.: Designing unmanned aircraft systems: a comprehensive approach. American Institute of Aeronautics and Astronautics Reston, VA (2012)
12. Duangsuwan, S., Maw, M.M.: Comparison of path loss prediction models for UAV and IoT air-to-ground communication system in rural precision farming environment. J. Commun. **16**, 60–66 (2021)
13. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
14. Puangpontip, S., Hewett, R.: Energy usage of deep learning in smart cities. In: Proceedings - 2020 International Conference on Computational Science and Computational Intelligence, CSCI 2020, pp. 1143–1148 (2020). https://doi.org/10.1109/CSCI51800.2020.00214