

# A<sup>2</sup>-UAV: Application-Aware Content and Network Optimization of Edge-Assisted UAV Systems

Andrea Coletta<sup>†</sup>, Flavio Giorgi<sup>†</sup>, Gaia Maselli<sup>†</sup>, Matteo Prata<sup>†</sup>,  
Domenicomichele Silvestri<sup>†</sup>, Jonathan Ashdown<sup>‡</sup>, and Francesco Restuccia<sup>\*</sup>

<sup>†</sup>Department of Computer Science, Sapienza University of Rome, Italy

<sup>‡</sup>Air Force Research Laboratory, United States

<sup>\*</sup>Institute for the Wireless Internet of Things, Northeastern University, United States

Corresponding Author: Andrea Coletta, e-mail: coletta@di.uniroma1.it

**Abstract**—To perform advanced surveillance, Unmanned Aerial Vehicles (UAVs) require the execution of edge-assisted computer vision (CV) tasks. In multi-hop UAV networks, the successful transmission of these tasks to the edge is severely challenged due to severe bandwidth constraints. For this reason, we propose a novel A<sup>2</sup>-UAV framework to optimize the number of correctly executed tasks at the edge. In stark contrast with existing art, we take an *application-aware* approach and formulate a novel Application-Aware Task Planning Problem (A<sup>2</sup>-TPP) that takes into account (i) the relationship between deep neural network (DNN) accuracy and image compression for the classes of interest based on the available dataset, (ii) the target positions, (iii) the current energy/position of the UAVs to optimize routing, data pre-processing and target assignment for each UAV. We demonstrate A<sup>2</sup>-TPP is NP-Hard and propose a polynomial-time algorithm to solve it efficiently. We extensively evaluate A<sup>2</sup>-UAV through real-world experiments with a testbed composed by four DJI Mavic Air 2 UAVs. We consider state-of-the-art image classification tasks with four different DNN models (i.e., DenseNet, ResNet152, ResNet50 and MobileNet-V2) and object detection tasks using YoloV4 trained on the ImageNet dataset. Results show that A<sup>2</sup>-UAV attains on average around 38% more accomplished tasks than the state of the art, with 400% more accomplished tasks when the number of targets increase significantly. To allow full reproducibility, we pledge to share datasets and code with the research community.

## I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs), or drones, have obtained significant attention thanks to their potential use in post-disaster scenarios, where human intervention is difficult or inefficient due to the vastness and/or harshness of the area. The key advantage of UAVs is the combined presence of advanced sensor equipment, wireless multi-hop networking and mobility in the same device, thus enabling critical applications such as automatic target (e.g., object, person) detection and tracking.

To perform their functions, modern UAVs necessarily depend on the execution of computation-heavy deep learning (DL) tasks to analyze in real time the images of the target area. These tasks usually rely on very deep neural networks (DNNs) such as ResNet [1] and DenseNet [2], which are computationally prohibitive for UAVs [3]. To extend UAVs battery lifetime and keep task execution time within acceptable levels, offloading the stream of tasks to neighboring edge

servers (e.g., the depot) is a feasible option [4–9]. Unfortunately, UAVs networks typically experience limited bandwidth and frequent packet loss [10]. Prior work on UAV edge task offloading — discussed in details in Section II — assumes a single-hop communication between the UAVs and the edge [11] [12], or focuses only on networking aspects on a multi-hop communication [13]. All fall short in considering the specific task requirements, which ultimately limits the number of correctly executed tasks. Existing work also rarely uses a testbed to measure performance experimentally.



Fig. 1: Examples of application-aware compression. A and C (B and D) represent the images before (after) processing.

In stark contrast, we propose A<sup>2</sup>-UAV, an *application-aware* (A<sup>2</sup>) optimization framework which jointly optimizes UAV network deployment and task accuracy. Our key intuition is that a different image compression will result in a different accuracy for the DNN model. Specifically, compressed images have low impact on the network throughput but decrease the DNN accuracy. Conversely, uncompressed images are likely to be classified more correctly but cause higher network overhead. To better highlight this intuition, Figure 1 shows an example of an original and compressed ( $l = 20$ ) images of *Wildlife* and *Tools* from the ImageNet dataset [14]. Figure 1B shows how an hatchet at compression level  $l = 5$  is less recognizable with respect to its original version (Figure 1A), due to the low contrast with the background and the small object dimension. On the other hand, Figure 1D shows how a jaguar at level  $l = 20$  is easily distinguishable, thanks to its fur pattern and the higher contrast with the background. We consider lossy compression algorithms for images (e.g., JPEG algorithm [15]), as lossless algorithms do not usually meet the requirements of real-time applications.

*Trading-off network load and latency with task accuracy while aiming at maximizing the target coverage is a daunting challenge.* A<sup>2</sup>-UAV considers all these aspects and balances

between images compression and network coverage according to current network conditions and application requirements. In Section III-C we show that different applications have starkly different compression-accuracy relationships. Specifically,  $A^2$ -UAV maximizes the number of accomplished tasks at the edge, producing a connected coverage formation for the UAVs of the squad and a compression level assignment for the UAVs that satisfy the application requirements. The connected coverage formation is designed such that it jointly maximizes the number of covered targets, minimizes network delay due to inefficient routes or channel contention, and minimizes task miss-classification due to high input compression. We show through simulation and *real-world experiments* with a testbed that GREEDY-Application-Aware Task Planning Problem ( $A^2$ -TPP) attains on the average 38% more accomplished tasks than the state-of-the-art networking-based approach, with a sharp increase (400% more accomplished tasks) when the number of tasks to offload drastically increases.

**This paper makes the following novel contributions:**

- We design  $A^2$ -UAV — a novel application-aware framework that optimizes the number of accomplished tasks at the edge by finding the best network deployment and compression level assignment for the considered application. We first design a *Application Aware Task Analyzer* ( $A^2$ -TA) to learn the requirements of the tasks, and to map the possible data compression of the UAVs to the expected task accuracy at the edge. Then, we define the  $A^2$ -TPP to assign the UAVs to the tasks. We prove the NP-hardness of the problem and formulate a polynomial-time GREEDY- $A^2$ -TPP algorithm to solve it efficiently;
- We study the performance of  $A^2$ -UAV with extensive simulations. We analyze six different critical applications for UAVs, including Search-and-Rescue, Maritime and Wildlife monitoring. We show how GREEDY- $A^2$ -TPP accomplishes around 38% more tasks with respect to existing network-based approaches, thanks to its application-aware optimization; Whereas, the NP-hard version (OPT- $A^2$ -TPP) attains on average 50% performance increase on restricted problem instances.
- We implement  $A^2$ -UAV into a testbed and perform *real-field experiments*. We consider four UAVs and a Jetson Nano board, mounting a Raspberry PI for computation and communication. We execute an image analysis application in which UAVs periodically acquire images from on-board sensors, with different delay requirements. We implement four state-of-the-art image classification models (i.e., DenseNet [2], ResNet152, ResNet50 [1] and MobileNet-V2 [16]), and one object detection model (YoloV4 [17]) which are executed at the edge server on the Jetson Nano board, and we let the UAVs offload tasks through WiFi connection. Experimental results confirm the outstanding performance of  $A^2$ -UAV.

## II. RELATED WORK

Only very recently has the literature considered task offloading in the context of edge-assisted DL-based applications [10, 12]. Chuprov et al. [10] show how the performance of

the end-line ML systems is affected by the quality of data and network. They consider packet loss and limited bandwidth in a image classification task, and they recommend to stop the system when packet loss reaches 2-5%. In Section V-C we show that our system enables the classification task even with 15% of packet loss. Chen et al. [13] consider a hierarchical offloading of computation tasks. Conversely from us, they focus on the communication and routing of tasks toward a more computational powerful device, without focusing on the specific task requirements. Yang et al. [11] propose a hierarchical DL task execution framework, in which only a few lower layers of a Convolutional Neural Network (CNN) are on the UAVs, while the edge server contains the higher layers of the model, which need more resources. However, a single-hop high-performance 4G network is considered, while we focus on the more challenging scenario of multi-hop connectivity toward the edge. Recently, Callegaro *et al.* [12] proposed SeReMAS, a framework where the application-, network- and telemetry-based features are used to select and assign UAVs tasks to the most reliable edge servers. However, a single-hop system is considered, and data compression is not explored.

As one of the output of  $A^2$ -UAV is a connected coverage formation, we mention also some prior art on UAVs deployment optimization [13, 18-22], which however does not consider task offloading. Natalizio et al. [23] have considered the problem of minimizing networking resources while maximizing the user experience (i.e., perceived quality) when filming sport events. Moreover, [4, 5, 24] optimize network deployment under continuous or periodic connectivity constraints, but they do not consider critical indicators such as task accuracy with delay constraints, which are critical to the UAVs mission. Recently, Nguyen proposed a Steiner-Tree-Based Algorithm (STBA) for target coverage and network connectivity [25], where Fermat points and the node-weighted Steiner tree algorithm are used to find a tree such that most of the targets are covered, and the UAVs are minimized. In Section V, we consider variants of [25] as performance benchmarks for  $A^2$ -UAV.

## III. THE $A^2$ -UAV FRAMEWORK

In this section, we give an overview of  $A^2$ -UAV (Section III-B) and describe the two key components of  $A^2$ -UAV:  $A^2$ -TA (Section III-C) and  $A^2$ -TPP (Section III-D).

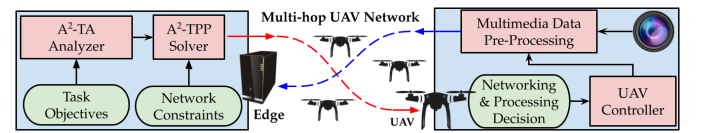


Fig. 2: High-level overview of  $A^2$ -UAV.

Figure 2 shows a high-level overview of  $A^2$ -UAV. The Application-Aware Task Analyzer ( $A^2$ -TA) at the *edge server* learns the relationship between the image compression and the accuracy on a set of classes of interest, and passes its output to the Application-Aware Task Planning Problem ( $A^2$ -TPP) solver, which jointly optimizes UAVs positions, routing policy,

and data compression strategy to maximize the number of correctly executed tasks per unit of time. The optimal network deployment and image compression levels are sent to the *UAVs network*, which moves to the targets, monitors them and streams tasks to the edge through the multi-hop connection.

#### A. System Model and Assumptions

We assume one or more UAVs are deployed over an Area of Interest (AoI), which contains several *targets*, e.g., the location of a vehicle, person, or any entity of interest. Each UAVs is equipped at least with (i) multimedia sensors (e.g., camera and microphone); (ii) a single radio for communication; and (iii) a computational unit. The edge server is equipped with low-latency hardware for DL computation. We do not rely on any communication infrastructures (e.g., 5G) and assume edge offloading is realized through multi-hop communication. A UAV monitors a target by sampling data through its sensors and generates a *task* to be executed at the edge. A task could be “car, bicycle, or bus detection on a video camera frame every 10 frames”. The task is then sent to the edge server through a multi-hop connection, where a state-of-the-art DL model is run to perform the task. We assume each task has mission-driven constraints in terms of (i) minimum classification accuracy given a specified DL model; (ii) maximum latency, defined as the time between the task generation and its successful execution. Thus, a task is successfully executed if (i) promptly offloaded to the edge; and (ii) correctly analyzed by the model within a deadline.

Notation	Description
$\mathcal{U}$	A set of available UAVs of the fleet
$\mathcal{T}$	A set of targets to cover
$\sigma$	The edge server
$r_{\text{com}}$	Drone's communication radius
$r_{\text{sens}}$	Drone's sensing radius
$l_u$	Distance traveled by a drone
$\delta_u$	Drone's overall energy consumption
$\hat{e}_{ij}^s$	Amount of data transmitted through the link between UAV $i$ and UAV $j$
$\hat{e}_{ij}^a$	Expected task accuracy at the edge, for each task
$\rho_{i,j}$	Estimated channel data rate
$\vec{p}_u$	Position vector assigned by the solver to drone
$\beta_u$	Energy spent for each distance unit traveled at constant speed
$\alpha_u$	Energy spent in a steady position
$\hat{\omega}_{ij}^u$	Drone $u$ monitors the target $i$ with a compression $j$ or not
$\phi_u$	UAV initial energy
$Q(s, l)$	A tuple with expected accuracy and data size of the frame of the application scenario $s$ , with compression level $l$
$\Psi$	The final connected coverage formation returned by the greedy algorithm
$\tau_{\text{best}}$	Best coverage found during an iteration
$\tau_{\text{par}}$	Partial coverage to enhance or join with $\Psi$

TABLE I: Table of Symbols.

#### B. Overview of $A^2$ -UAV

The ultimate goal of  $A^2$ -UAV is to maximize the number of correctly executed tasks. To approach this challenging issue, and conversely from existing work,  $A^2$ -UAV takes into account how the task success is affected by the image compression. To this end,  $A^2$ -UAV jointly optimizes the deployment of UAVs and the task offloading to maximize the number of executed

tasks. Each UAV is made up of two key modules. First, the *UAV Controller* implements networking and data processing decisions (next position, targets to cover, sampling process, and offloading routes) received from the  $A^2$ -TPP. Second, the *Multimedia Data Pre-Processing* module samples data and creates tasks by pre-processing collected multimedia data according to the  $A^2$ -TPP solution.

#### C. Application-Aware Task Analyzer ( $A^2$ -TA)

The  $A^2$ -TA module determines the relationship between different image compression levels and task accuracy so that UAVs can reduce the amount of transmitted data and avoid network congestion, while satisfying application requirements.

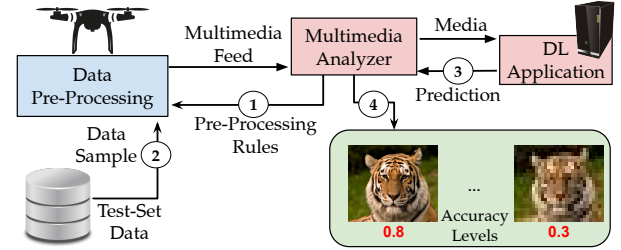


Fig. 3: Main Operations of the  $A^2$ -TA module.

Figure 3 shows the workflow of the  $A^2$ -TA, which is executed before the network deployment, by using the datasets of the specific DL application. The  $A^2$ -TA iterates over the sampled data, changing compression according to the UAV sensors capabilities (**step 1** and **step 2**). Each revised data sample is fed to the DL application, which outputs a prediction (**step 3**). Finally, predictions are compared with the ground truth, to infer and store the model performance according such data compression (**step 4**).

More formally, the function:  $Q : \mathcal{S} \times \mathcal{L} \rightarrow \mathbb{R}^2$  maps an application scenario in the set  $\mathcal{S}$  and a compression level in the discrete set  $\mathcal{L} = \{1, \dots, 100\}$  a tuple in  $\mathbb{R}^2$  representing the average *accuracy* and *data size*. The function is determined by iterating over the application scenarios, possible compression levels and relative dataset entries. Each sampled image is compressed according the compression level  $l$  by the JPEG compression algorithm [15], and it is fed into the DL model to determine accuracy and data size.

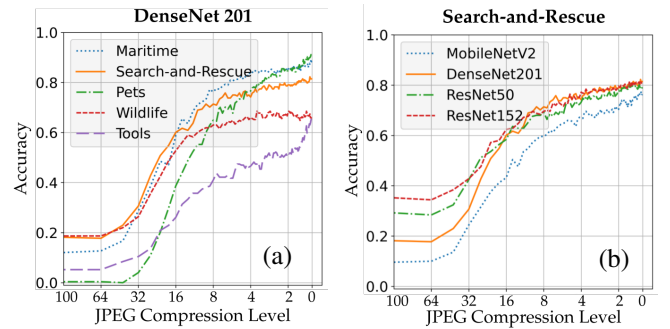


Fig. 4: Accuracy as a function of JPEG compression level.



To give an example, we consider 5 scenarios: **Maritime** (Fireboat, Wreck, Lifeboat, Ocean liner, Speedboat), **Search-and-Rescue (SaR)** (Fire truck, Ambulance, Police van, German shepherd, Pickup truck) **Wildlife** (Kit fox, Polecat, Red wolf, Zebra, Jaguar), **Tools** (Screwdriver, Power drill, Hatchet, Hammer, Chainsaw), **Pets** (Golden retriever, Pomeranian, Guinea pig, Persian cat, Hamster). Figure 4a shows the accuracy for the different scenarios as a function of the compression, while Figure 4b shows the accuracy for the same scenario when different DL models are used. The figures highlight the need of the A<sup>2</sup>-TA. For example, images of *Tools* have low accuracy, constraining the compression at level  $l = 8$  to achieve at least 40% of accuracy, while *Wildlife* can achieve the same accuracy with higher compression  $l = 25$ .

#### D. A<sup>2</sup>-TPP MILP Formulation

We formalize A<sup>2</sup>-TPP as a Mixed Integer Linear Programming (MILP). Hereafter we denote the edge server as  $\sigma$ , the set of targets to monitor as  $\mathcal{T}$ , the set of available UAVs of the fleet as  $\mathcal{U}$ . The A<sup>2</sup>-TPP solver outputs: (i) a connected coverage formation of UAVs, and (ii) the compression level each drone must adopt to capture images when inspecting a target.

**Definition III.1.** A subset of UAVs  $F \subseteq \mathcal{U}$  is deployed according to a **connected coverage formation**, when some of the UAVs in  $F$  are employed to inspect a subset of targets  $M \subseteq \mathcal{T}$ , while being connected to the base station  $\sigma$ , either directly or through a multi-hop sequence of the other UAVs in  $F$ .

**Definition III.2.** A **task** for a drone  $u \in F$  covering a target  $t \in M$ , consists in delivering an image captured from the drone's on-board cameras to the base station  $\sigma$ . It is said to be an **accomplished task** if two conditions hold: (1) when received at  $\sigma$ , the time since the task was created is not superior to a threshold  $\Delta$ ; (2) when the captured image reaches  $\sigma$ , the DL model outputs a correct prediction for it.

We define the UAV sensing range and communication range as  $r_{\text{sens}}$  and  $r_{\text{com}}$ , respectively. We denote with  $p$  the position vector of the entities involved, and to  $(p_u^x, p_u^y)$  for the  $x$  and  $y$  coordinates respectively. In particular  $p_u$  is the position of UAV  $u$ ,  $\forall u \in \mathcal{U}$  at the beginning of the mission;  $p_i$  to the position of the target  $i$ ,  $\forall i \in \mathcal{T}$ ;  $p_\sigma$  the position of the edge server. We denote with  $\hat{p}_u$  the position vector assigned by the solver to drone  $u$ ,  $\forall u \in \mathcal{U}$ . We define the distance traveled for each drone as  $l_u = |p_u - \hat{p}_u|$ ,  $\forall u \in \mathcal{U}$ .

To estimate energy consumption, we define  $\beta_u$  as the energy spent for each distance unit traveled at constant speed, and  $\alpha_u$  as the energy spent in a steady position, for a given unit of time. The values of  $\beta_u$  and  $\alpha_u$  are estimated through on-field experiments or from technical specifications. The overall energy consumption for UAV  $u$  is defined as  $\delta_u = l_u \cdot \beta_u + \alpha_u \cdot \Lambda$ ,  $\forall u \in \mathcal{U}$ , where  $\Lambda$  is an upper-bound of the time required, once reached the targets, to monitor the

targets and complete the mission. We constraint the reachable points according to the UAVs initial energy  $\phi_u$ , as follows:

$$\delta_u + |p_\sigma - \hat{p}_u| \cdot \beta_u \leq \phi_u, \forall u \in \mathcal{U} \quad (1)$$

This constraint defines the positions that are reachable as they let the UAVs with enough energy to come back to the edge-server for recharging operations. We use the binary variables  $\hat{\omega}_{ij}^u \in \{0, 1\}$ , which define if the drone  $u$  monitors the target  $i$  with compression level  $j$  or not. A target  $i$  is monitored if and only if the UAV  $u$  is close enough to the target position  $p_i$ . Formally we want to constraint  $\hat{\omega}_{i,j}^u = 1 \iff |\hat{p}_u - p_i| \leq r_{\text{sens}}$  which becomes  $\forall u \in \mathcal{U}, \forall i \in \mathcal{T}, \forall j \in \mathcal{L}$ :

$$\begin{aligned} r_{\text{sens}} &\geq |\hat{p}_u - p_i| - M_{\text{cost}} \cdot (1 - \hat{\omega}_{i,j}^u) \\ r_{\text{sens}} &\leq |\hat{p}_u - p_i| + M_{\text{cost}} \cdot \hat{\omega}_{i,j}^u \end{aligned} \quad (2)$$

where  $M_{\text{cost}}$  is a big constant number. Next, we enforce that a target is covered by at most one UAV and that each UAV can cover only one target:

$$\sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{L}} \hat{\omega}_{ij}^u \leq 1, \forall i \in \mathcal{T}, \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{L}} \hat{\omega}_{ij}^u \leq 1, \forall u \in \mathcal{U} \quad (3)$$

We introduce an extended node set  $\mathcal{V} = \mathcal{U} \cup \{\sigma\}$  and we consider all the possible communication paths among nodes, i.e., all the edges  $(i, j) \forall i, j \in \mathcal{V}$ . A binary variable  $\hat{\gamma}_{i,j} \in \{0, 1\}$ , indicates if the nodes  $i$  and  $j$  are too distant to communicate. The relation  $|\hat{p}_i - \hat{p}_j| \geq r_{\text{com}} \Rightarrow \hat{\gamma}_{i,j} = 0$  is enforced as follows:

$$|\hat{p}_i - \hat{p}_j| \leq r_{\text{com}} + M_{\text{cost}} \cdot (1 - \hat{\gamma}_{i,j}), \forall i, j \in \mathcal{V} \quad (4)$$

We define the data frame offloading as a network flow formulation. We introduce a set of variables  $\hat{e}_{ij}^s$  defining the amount of data transmitted through the link between UAV  $i$  and UAV  $j$ . We define  $\hat{e}_{ij}^a$  to account for the expected task accuracy at the edge, for each task. We impose that the edge does not generate any outgoing flow, for both data and accuracy flows:

$$\sum_{j \in \mathcal{V}} \hat{e}_{\sigma j}^s \leq 0, \sum_{j \in \mathcal{V}} \hat{e}_{\sigma j}^a \leq 0 \quad (5)$$

We allow a flow only for between neighboring nodes:

$$\hat{e}_{ij}^s + \hat{e}_{ij}^a \leq \hat{\gamma}_{i,j} \cdot M, \forall i, j \in \mathcal{U} \quad (6)$$

The maximum bandwidth allowed between two UAVs is constrained to respect the estimated channel data rate  $\rho_{i,j}$ :

$$\sum_{j \in \mathcal{V}} \hat{e}_{ij}^s \leq \rho_{i,j}, \forall i \in \mathcal{U} \quad (7)$$

We specify that a UAV can transmit only towards another UAV, resulting into a tree rooted at the edge:

$$\sum_{j \in \mathcal{V}} \hat{\gamma}_{ij} \leq 1, \forall i \in \mathcal{U} \quad (8)$$

We impose flow conservation as follows:

$$\sum_{k \in \mathcal{V}} \hat{e}_{uk}^s - \sum_{k \in \mathcal{V}} \hat{e}_{ku}^s = \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{L}} b_{i,j} \cdot \hat{\omega}_{ij}^u, \forall u \in \mathcal{U} \quad (9)$$

which imposes that, for each outgoing edge from  $u$ , the flow is increased by expected data size of the target covered by the UAV  $u$ . We also impose that the edge receives all the data produced by the covered targets:

$$\sum_{k \in \mathcal{V}} \hat{e}_{k\sigma}^s = \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{L}} b_{i,j} \cdot \hat{\omega}_{ij}^k \quad (10)$$

To conclude, we constraint the accuracy of the targets at the edge-server, as follows:

$$\sum_{k \in \mathcal{V}} \hat{e}_{uk}^a - \sum_{k \in \mathcal{V}} \hat{e}_{ku}^a = \sum_{t_j^i \in \mathcal{T}'} a_{i,j} \cdot \hat{\omega}_{ij}^u, \forall u \in \mathcal{U} \quad (11)$$

**Objective Function:** maximize covered targets, DL tasks accuracy, and energy spent by the UAVs:

$$\max \quad \alpha \cdot \sum_{j \in \mathcal{V}} \hat{e}_{\sigma j}^a + \beta \cdot \sum_{i \in \mathcal{T}, j \in \mathcal{L}, u \in \mathcal{U}} \hat{\omega}_{ij}^u - \eta \cdot \sum_{u \in \mathcal{U}} l_u \quad (12)$$

The term  $\alpha$  prioritizes the maximization of the accuracy, while  $\beta$  weights the importance of covering the targets and  $\eta$  minimized the distance traveled by the UAVs.

**Theorem III.1.** *The  $\mathcal{A}^2$ -TPP problem is NP-Hard.*

*Proof.* We show that  $\mathcal{A}^2$ -TPP generalizes the *Steiner tree problem with minimum number of Steiner points and bounded edge-length* STPMSPBEL, a known NP-hard problem [26]. Given a set  $P$  of  $n$  terminal points in a 2-dimensional plane, a positive constant  $R$ , and a non-negative integer  $B$ , STPMSPBEL asks whether it exists a tree spanning a set of points  $P \subseteq Q$  s.t. each edge has a length less than  $R$  and the number of Steiner points (i.e.,  $Q \setminus P$ ) is less than or equal to  $B$ . Any instance of STPMSPBEL can be reduced to an instance of our problem in polynomial time. The set of points  $P$  represents our target set  $\mathcal{T} \cup \{\sigma\}$ , and  $B$  defines the number of available UAVs, with communication range equal to  $R$ . We consider UAVs with unlimited batteries and one compression level (i.e.,  $|\mathcal{L}| = 1$ ). This problem instance finds a solution that maximizes the number of connected targets with the edge server, moving the minimum number of UAVs. If such a solution exists, and covers all the points in  $P$ , then it also exists a tree spanning a set of points  $P \subseteq Q$ , where each edge has length less than  $R$  and the number of Steiner points is less then or equal to  $B$ . The complexity of the above reduction is polynomial, thus we derive that  $\mathcal{A}^2$ -TPP problem is at least as hard as the STPMSPBEL problem [26].  $\square$

#### IV. A POLYNOMIAL TIME HEURISTIC FOR $\mathcal{A}^2$ -TPP

We propose a greedy heuristic to solve  $\mathcal{A}^2$ -TPP in polynomial time. We first introduce the algorithm, and then prove its polynomial time complexity.

##### A. Algorithm Overview

GREEDY- $\mathcal{A}^2$ -TPP outputs a *connected coverage formation* – also referred to as *coverage* for brevity – for the UAVs, and a *compression level assignment* for each covered target. Both coverage and compression need to meet the criteria expressed in Equation [12] that is, optimizing the *number of accomplished*

---

#### Algorithm 1: GREEDY $\mathcal{A}^2$ -TPP

---

**Input:**  $\mathcal{U}$ : set of UAVs,  $\mathcal{T}$ : set of targets

**Output:**  $\Psi$  a connected coverage formation

---

```

1  $\hat{\mathcal{T}}, \Psi, \tau_{\text{par}}, c_{\text{par}} \leftarrow \{\sigma\}, \{\sigma\}, \{\sigma\}, 0$ 
2 while  $\mathcal{T} - \hat{\mathcal{T}} \neq \emptyset$  or  $|V_{\Psi} \cup V_{\text{par}}| < |\mathcal{U}|$  do
3    $t_{\text{best}}, \tau_{\text{best}}, c_{\text{best}} \leftarrow \emptyset, \emptyset, \infty$ 
4   for  $t \in \mathcal{T} - \hat{\mathcal{T}}$  do
5      $\tau_{\text{temp}} \leftarrow \text{TST}(\{t\} \cup \mathcal{C}(\tau_{\text{par}}), r_{\text{com}})$ 
6      $c_{\text{temp}} \leftarrow \text{COST}_{\alpha}(\tau_{\text{temp}}, \tau_{\text{par}}, \Psi, \text{COMPRESSION}(\tau_{\text{temp}}))$ 
7     if  $c_{\text{temp}} < c_{\text{best}}$  and  $|V_{\text{temp}}| - 1 \leq |\mathcal{U}| - |V_{\Psi} \cup V_{\text{par}}|$  then
8        $t_{\text{best}}, \tau_{\text{best}}, c_{\text{best}} \leftarrow t, \tau_{\text{temp}}, c_{\text{temp}}$ 
9   if  $t_{\text{best}} = \emptyset$  then
10     $\Psi \leftarrow \Psi \cup \tau_{\text{par}}$ 
11    break
12   $\tau_{\text{los}} \leftarrow \text{TST}(\{\sigma, t_{\text{best}}\}, r_{\text{com}})$ 
13   $c_{\text{los}} \leftarrow \text{COST}_{\alpha}(\tau_{\text{los}}, \tau_{\text{par}}, \Psi, \text{COMPRESSION}(\tau_{\text{los}}))$ 
14  if  $c_{\text{los}} < c_{\text{best}} - c_{\text{par}}$  then
15     $\Psi \leftarrow \Psi \cup \tau_{\text{par}}$ 
16     $\tau_{\text{par}}, c_{\text{par}} \leftarrow \tau_{\text{los}}, c_{\text{los}}$ 
17  else
18     $\tau_{\text{par}}, c_{\text{par}} \leftarrow \tau_{\text{best}}, c_{\text{best}}$ 
19   $\hat{\mathcal{T}} \leftarrow \hat{\mathcal{T}} \cup \{t_{\text{best}}\}$ 
20  $R, \cdot \leftarrow \text{COMPRESSION}(\Psi)$ 
21 return  $\Psi, R$ 
```

---

tasks. Our approach is to maximize the number of inspected targets, producing a coverage of minimum congestion, and minimizing task misclassification due to low frame resolution. Specifically, a coverage  $\tau = (V, E, W)$  is a Triangular Steiner Tree [27] in which the set of nodes  $V$  represents the positions UAVs must reach to cover the target nodes in  $M$ , while staying connected with the base station  $\sigma$  in a multi-hop manner. The set  $E$  represents the link between UAVs, thus the routes data streams must follow through the network. The function  $W$  maps each  $e \in E$  to a weight that represents link's bandwidth. In our implementation we estimate this value empirically. It is assumed that at the base station, communication happens through dedicated transceivers and does not require actual coverage with a drone. Thus, at any time it holds  $|V| \leq |\mathcal{U}| + 1$ .

##### B. GREEDY- $\mathcal{A}^2$ -TPP

Algorithm 1 returns a coverage formation  $\Psi$ , merging partial coverage formations  $\tau_{\text{par}}$  generated to cover targets using the minimum deployment cost at each iteration. In the initialization phase, we let:  $\hat{\mathcal{T}}$  be the set of covered targets, initially containing only the base station;  $\Psi$  be the coverage archived so far;  $\tau_{\text{par}}$  be the partial coverage iteratively grown that is added to  $\Psi$  when it cannot be further expanded;  $c_{\text{par}}$  be the cost of the partial coverage generated so far (line 1). The while loop iterates until either all the targets are covered  $\mathcal{T} - \hat{\mathcal{T}} \neq \emptyset$  or the number of UAVs used does not exceed the fleet size (line 2). The variables  $t_{\text{best}}, \tau_{\text{best}}, c_{\text{best}}$  contain respectively the best target found at each iteration, the coverage including that target and its cost. A for loop over the uncovered targets  $\mathcal{T} - \hat{\mathcal{T}}$  allows to find the best target to add, building new temporary coverage formations  $\tau_{\text{temp}}$  using the targets already covered by  $\tau_{\text{par}}$  (namely the set  $\mathcal{C}(\tau_{\text{par}})$ ) and adding to them the candidate target  $t$ . Then we evaluate the cost of  $\tau_{\text{temp}}$  (lines 3-6). This cost combines the number of drones needed for the coverage, and the loss in accuracy due to the channel

---

**Algorithm 2: COMPRESSION**


---

**Input:** a coverage formation  $\tau_i$

**Output:**  $R$  vector with compression levels for all targets in  $\tau_i$ ,  $L$  vector with loss in accuracy due to all targets in  $\tau_i$

---

```

1 sort  $t$  by  $Q(S(t), *) \cdot b \forall t \in \mathcal{C}(\tau_i)$  in ascending order
2  $\hat{\mathcal{C}}, L, R \leftarrow \mathcal{C}(\tau_i), \langle \rangle, \langle \rangle$ 
3 for  $t \in \hat{\mathcal{C}}$  do
4    $P \leftarrow \text{SHORTEST-PATH}(\tau_i, \sigma, t)$ 
5    $B \leftarrow \text{BOTTLENECK}(\tau_i, P)$ 
6    $R(t) \leftarrow \arg \max_{l \in \mathcal{L}} Q(S(t), l) \cdot b \leq \min\{B; Q(S(t), *) \cdot b\}$ 
7    $L(t) \leftarrow Q(S(t), *) \cdot a - Q(S(t), R(t)) \cdot a$ 
8    $W_i(e) \leftarrow W_i(e) - Q(S(t), R(t)) \cdot b \quad \forall e \in P$ 
9    $\hat{\mathcal{C}} \leftarrow \hat{\mathcal{C}} - \{t\}$ 
10 return  $R, L$ 

```

---

contention. We will talk in more detail about how this cost is computed when describing Algorithm 2. Then we check if: (i)  $\tau_{\text{temp}}$  has a lower cost than  $\tau_{\text{best}}$  (ii) and if  $\tau_{\text{temp}}$  can be covered with the remaining UAVs. If both checks go through, then  $t$  becomes the best candidate  $t_{\text{best}}$  and the associated candidate coverage  $\tau_{\text{temp}}$  with its cost  $c_{\text{temp}}$  are stored into  $\tau_{\text{best}}$  and  $c_{\text{best}}$  respectively (lines 7-8). In case no target was set as a best candidate, (i.e.,  $t_{\text{best}} = \emptyset$ ) the while loop breaks. This happens only when the second condition at line 7 is not met for any target, that is no coverage formations can stick to the remaining fleet size constraint. Then, the cost paid to cover only  $t_{\text{best}}$  that is  $c_{\text{best}} - c_{\text{par}}$  is compared to the cost  $c_{\text{los}}$  of a new line-of-sight (los) branch  $\tau_{\text{los}}$  grown using only  $t_{\text{best}}$  as target. If  $\tau_{\text{los}}$  costs less than the partial grown tree so far  $\tau_{\text{par}}$ , then  $\tau_{\text{par}}$  is merged with the final tree  $\Psi$ . Then  $\tau_{\text{los}}$  becomes the new partial connected coverage to grow. Otherwise growing  $\tau_{\text{par}}$  is still convenient, so  $\tau_{\text{best}}$  becomes the new partial deployment including the new target  $t_{\text{best}}$  and  $\tau_{\text{los}}$  is discarded (lines 12-19). When the algorithm terminates (line 20) the final coverage  $\Psi$  along with all the compression levels assigned to each target are returned.

### C. Assignment of Compression Levels

Algorithm 2 determines the compression levels for each UAV in  $F$  inspecting targets in  $M$ . The rationale is to increase the compression of data flowing from a target, based on the bandwidth assigned to it, leaving more bandwidth to targets having more to send. The algorithm iterates over the targets  $t$  covered by the candidate input tree, sorted in ascending order based  $Q(S(t), *) \cdot b$ , that is the load produced by the target  $t$  according to the task analyzer  $\mathcal{A}^2\text{-TA}$ , belonging to the application scenario  $S(t)$  and at the minimum compression level (denoted by  $*$ ) (lines 1-3). At each iteration a bottleneck bandwidth  $B$  for the target is computed. This quantity is the bottleneck capacity on the path from the source of flow  $t$ , to the destination  $\sigma$ . This value is influenced by the number of targets  $t$  shares this path with. The bandwidth allocation function can be thought as slight modification of the Depth First Search (DFS) (line 5). To derive the maximum quantity of load that can be transferred from the target  $t$  per unit time, we vary the compression level while remaining subject to the flow

constraint (line 6). We store in the vector  $R$  the compression level for each target. We store the loss in accuracy for  $t$  subject to compression level  $R(t)$ , comparing the accuracy due to the best quality  $Q((S(t), *) \cdot a$  (line 7). The weights of the tree are updated considering the used bandwidth (line 8). Both the compression levels and the loss for each target are returned.

### D. Cost of a Coverage

The cost of a connected coverage formation is parameterized by  $\alpha$ . This exogenous parameter weights the importance given to the accuracy of the tasks. Notice that the importance given to task accuracy opposes to the minimization of the number of UAVs employed. Therefore the cost is a linear combination of the average loss in accuracy, and the percentage of used UAVs to cover the new target in  $\tau_i$  which was not present in the previous formation  $\tau_{i-1}$ , the cost is computed as  $\text{COST}_\alpha$ :

$$\alpha \cdot \frac{\sum_{t \in \mathcal{C}(\tau_i)} L(t)}{|\mathcal{C}(\tau_i)|} + (1 - \alpha) \cdot \frac{|V_i - V_{i-1}| - 1}{|U| - |V_\Psi \cup V_{i-1}|} \quad (13)$$

### E. GREEDY- $\mathcal{A}^2\text{-TPP}$ Example Execution

Figure 5 shows an example of execution of GREEDY- $\mathcal{A}^2\text{-TPP}$ . The gray triangle is the edge server  $\sigma$ . The black dots and red squares represent the target and relay positions, respectively. Figure 5-a shows three temporary coverage  $\tau_{\text{temp}}$ , each covering a different target. The cost of each of the coverage is compared (algo. 1, line 7). Say  $\tau_{\text{temp}}\langle\sigma, t_3\rangle$  is the cheapest coverage among them, that is the tree covering  $\sigma$  and  $t_3$ . At the subsequent iteration shown in Figure 5-b, two Triangular Steiner Trees covering  $\sigma, t_3$  and a new target among the remaining uncovered ones in  $\mathcal{T} - \hat{\mathcal{T}}$  (i.e.,  $t_2$  and  $t_1$ ) are proposed. Say the tree  $\tau_{\text{temp}}\langle\sigma, t_3, t_2\rangle$  is the cheapest coverage among them. In Figure 5-c the cost of  $\tau_{\text{temp}}\langle\sigma, t_3, t_2\rangle$  is compared with a line of sight coverage  $\tau_{\text{los}}\langle\sigma, t_2\rangle$ . The cheapest coverage among the two becomes the one to grow from the subsequent iterations (algo. 1, line 14). Say the cheapest coverage among them is  $\tau_{\text{temp}}\langle\sigma, t_3, t_2\rangle$ . In Figure 5-d we see two grown versions of the tree covering  $t_1$ , whereas in Figure 5-e we see a line of sight coverage of  $t_1$ . Say that comparing the cost of  $\tau_{\text{temp}}\langle\sigma, t_3, t_2, t_1\rangle$ , and  $\tau_{\text{los}}\langle\sigma, t_1\rangle$ , the cheapest is the line-of-sight version. The tree  $\tau_{\text{los}}\langle\sigma, t_1\rangle$  becomes the new

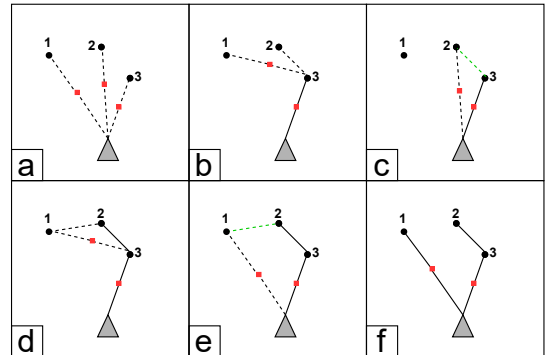


Fig. 5: GREEDY- $\mathcal{A}^2\text{-TPP}$  algorithm example

tree to grow from the subsequent iterations.  $\tau_{\text{temp}}(\sigma, t_3, t_2)$  is archived in  $\Psi$ . There are no more targets to cover.  $\tau_{\text{los}}(\sigma, t_1)$  is archived in  $\Psi$  the algorithm stops returning  $\Psi$ .

#### F. Properties of GREEDY-A<sup>2</sup>-TPP

**Lemma IV.1.** *Computing the compression level assignment has polynomial time complexity of  $O(|\mathcal{U}|^2)$ .*

*Proof sketch.* To measure the cost of a coverage tree  $\tau_i$ , the set of targets in the tree  $\mathcal{C}(\tau_i)$  is sorted by their expected transmission load in ascending order. Sorting requires  $O(|\mathcal{T}| \log |\mathcal{T}|)$  time complexity. The for loop iterates over the targets in  $\tau_i$  first computing the bottleneck bandwidth for  $t$ , having approximately the cost of a Depth First Search and Shortest Path, that is  $O(\log |V_i|)$  for the tree. Iterating over the compression levels to find the highest resolution to fit the bandwidth has constant complexity  $|\mathcal{L}|$  i.e., the cardinality of the discrete set of possible compression levels. Iterating over the edges of the path  $P$  to update the residual bandwidth has cost  $O(\log |V_i|)$ . Other assignments have evident constant complexity. By noticing that  $|V_i| = O(|\mathcal{U}|)$  the overall time complexity of computing the cost of a coverage tree is  $O(|\mathcal{T}| \log |\mathcal{U}|)$ . The complexity further simplifies by considering  $|\mathcal{T}| = O(|\mathcal{U}|)$ , thus resulting in  $O(|\mathcal{U}|^2)$ .  $\square$

**Theorem IV.2** (Time Complexity of GREEDY-A<sup>2</sup>-TPP). *GREEDY-A<sup>2</sup>-TPP with input  $\mathcal{T}$  targets sets has polynomial time complexity of  $O(|\mathcal{U}|^6)$ .*

*Proof sketch.* The while loop is executed, in the worst case, until all the targets in  $\mathcal{T}$  are included in the final solution  $\Psi$ . Within the while loop, a for loop iterates over the set of uncovered targets. For each of them a Triangular Steiner Tree  $t_{\text{temp}}$  is computed, and the time complexity is bounded by  $O(|\mathcal{T}|^4)$  [27]. The cost of each tree is computed with complexity  $O(|\mathcal{U}|^2)$  as shown in Lemma IV.1. Once the best candidate target to cover has been chosen, the Triangular Steiner Tree  $t_{\text{los}}$  of the shortest path towards the target, and the relative cost  $c_{\text{los}}$  are computed. The time complexity to find a stripe can be considered constant in time. The overall time complexity of GREEDY-A<sup>2</sup>-TPP is thus given by:  $O(|\mathcal{T}|(|\mathcal{T}|(|\mathcal{T}|^4 + |\mathcal{U}|^2) + |\mathcal{U}|^2)) = O(|\mathcal{U}|^6)$ .  $\square$

## V. PERFORMANCE EVALUATION

We extensively evaluate A<sup>2</sup>-UAV through simulation (Sect. V-B) as well as real-world experiments (Sect. V-C).

### A. Evaluation Setup

**Application.** We consider a monitoring application where UAVs need to perform image classification or object detection tasks on *target* locations by sampling images at given frame rate (e.g., 24 frames per second (fps)). We adopt (i) *ResNet-50*, a CNN with 50 layers [1]; (ii) *ResNet-152*, an extended version with 152 layers [1]; (iii) *DenseNet* [2], which consists of a Dense Convolutional Network (i.e., each layer is connected to all the other layers in a feed-forward fashion); (iv) *MobileNet-V2* [16], a new neural architecture for mobile devices; (v)

YoloV4, the state-of-the-art model for object detection. All the models were trained on the ImageNet database [14].

**Scenarios.** To emulate common scenarios for UAVs, we use the five scenarios described in Section III-C, i.e., *Maritime*, *Search-and-Rescue*, *Wildlife*, *Tools*, *Pets*. We also design an *Urban* reconnaissance scenario including various objects, such as *wreck*, *fireboat*, *ambulance*, *police van*, *revolver*, *crate*, *packet*, *backpack*, *mountain bike*, *motor scooter*. To ensure repeatability of our experiments, we let the UAVs sample images from a labeled subset of ImageNet. Where not otherwise stated, each target location generates 500 tasks (images) uniformly sampled among these classes.

**Metrics.** We measure the *Percentage of Accomplished Tasks*, defined as the ratio between the number of successfully completed tasks (according to Definition III.2) and the number of the generated tasks. The accomplishment of a task is influenced by its deadline  $\Delta$ . In order to study the performance of A<sup>2</sup>-TPP at varying application scenarios, we let  $\Delta$  vary: low values represent delay critical applications (e.g., intrusion detection), whereas high values, delay tolerant ones (e.g., agriculture). We also measure *Computational Time*, that is the time required by the algorithms to output a connected coverage formation and compression levels for the targets.

**Comparison.** We evaluate A<sup>2</sup>-UAV through real-field experiments and simulation, considering both the optimal solution OPT-A<sup>2</sup>-TPP and the greedy algorithm GREEDY-A<sup>2</sup>-TPP, against STBA [25]. STBA is a state-of-the-art networking-based approach that is the closest to our work. STBA covers a set of targets while providing network connectivity to the edge server. To find a connected tree, STBA uses a node-weighted Steiner tree algorithm, which computes a set of Fermat points to place relays, and then computes a tree among the targets and the edge-server, minimizing the needed UAVs. To allow for a fair comparison, we enhance STBA with data compression in three variants: 1) H-STBA, which does not compress data, but uses the **H**ighest available quality for collected data ( $l = 1$ ); 2) M-STBA which uses the **M**edium compression ( $l = 50$ ); and 3) L-STBA which uses an extreme compression ( $l = 100$ ) resulting in the **L**owest data quality.

### B. Simulation Results

We used the NS-3 network simulator [28], setting most of parameters in line with the devices used in our real-field experiments (e.g., WiFi interface 802.11n at 2.4 GHz), and testbed measured values (UAVs transmission range is 16m, sensing radius 1m, and maximum speed 5m/s [1]). The simulated area is a square of 500 × 500m, with an edge-server positioned in the center of the bottom border. The number of targets varies from 4 to 50, and the number of UAVs from 4 to 50.

1) *Multiple Scenarios:* Figures 6 illustrates the efficacy of our solution for different scenarios, reporting also the theoretical upper bound (blue dotted line). In the most challenging scenario, i.e., *Tools*, with a strict task deadline ( $\Delta = 0.1\text{sec}$ )

<sup>1</sup>The code is available at <https://github.com/flaat/AA-UAV>



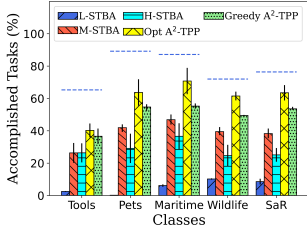


Fig. 6: Accomplished tasks (%),  $\Delta = 0.1\text{sec}$

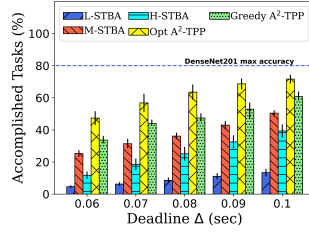


Fig. 7: Accomplished tasks (%) at increasing of  $\Delta$

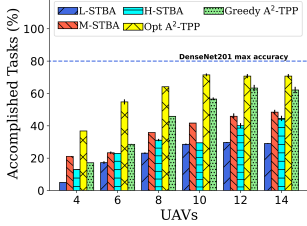


Fig. 8: Accomplished tasks (%) with 6 Targets,  $\Delta = 0.1\text{sec}$

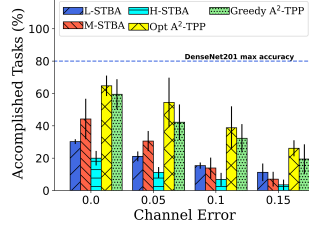


Fig. 9: Accomplished tasks (%) 4 targets,  $\Delta = 0.1\text{sec}$

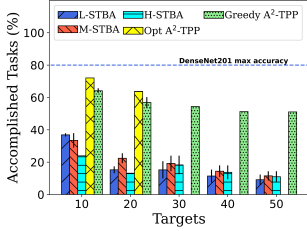


Fig. 10: Accomplished tasks (%), increasing targets

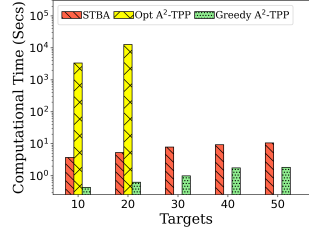


Fig. 11: Computational time (sec)

and DenseNet-201 DL model, **OPT-A<sup>2</sup>-TPP completes 41% of tasks, with an improvement of 52% respect the best STBA variant**, i.e., H-STBA, which completes less than 27% of tasks. The theoretical upper-bound for DenseNet-201 in the same scenario is 60%, meaning that under ideal network conditions of zero latency and no compression, the DL model would correctly classify only 60% of the tasks (Figure 4a show the complexity of predicting tools images, even for uncompressed images). In the case of *Pets* and *Maritime*, OPT-A<sup>2</sup>-TPP reaches the highest percentage of accomplished tasks — 65% and 70% respectively — where the upper-bounds are 90% and 88%. **The improvement with respect to the best STBA variant, i.e., M-STBA, is 55% and 50%.** *Pets* require a compression level lower than  $l = 50$  (see Figure 4a) to achieve satisfactory performance, forcing both OPT-A<sup>2</sup>-TPP and GREEDY-A<sup>2</sup>-TPP to select a medium compression level, more similarly to M-STBA.

In *Wildlife* and *Search-and-Rescue* (SaR), the gap between both the A<sup>2</sup>-TPP versions and STBA variants increases significantly. OPT-A<sup>2</sup>-TPP and GREEDY-A<sup>2</sup>-TPP complete respectively 60%, 63% and 49% and 54% of tasks, against 39% and 37% of M-STBA. The motivation behind this sharp improvement is the use of the A<sup>2</sup>-TA, which understands that even high compressed images can achieve satisfactory performance. Therefore, both our solutions can achieve high

accuracy with low network usage, executing the tasks within their deadline  $\Delta = 0.1$  seconds. GREEDY-A<sup>2</sup>-TPP completes 20% and 31% more tasks than M-STBA.

2) *Urban Scenario*: Figure 7 shows the performance in the *Urban* scenario as a function of task deadline  $\Delta \in \{0.06, 0.07, 0.08, 0.09, 0.1\}$ , when DenseNet-201 is employed. OPT-A<sup>2</sup>-TPP accomplishes tasks up to 72% in the case of  $\Delta = 0.1\text{sec}$ , while the best variant M-STBA achieves only 48% of tasks at the same  $\Delta$ . **OPT-A<sup>2</sup>-TPP accomplishes 58% more tasks than M-STBA with the tightest deadline**, as it adapts the compression of images to meet the latency constraint. The plot also confirms the performance of OPT-A<sup>2</sup>-TPP that outperforms the network-based approaches (i.e., M-STBA) up to 45 – 50%. **GREEDY-A<sup>2</sup>-TPP follows the OPT-A<sup>2</sup>-TPP trend always remaining widely above the performance of STBA solutions.** Figure 8 shows the percentage of accomplished tasks as function of the number of UAVs, with 6 targets randomly distributed in the area. We employ DenseNet-201, which achieves a maximum accuracy of 80%, and set  $\Delta = 0.1\text{sec}$ . Both OPT-A<sup>2</sup>-TPP and GREEDY-A<sup>2</sup>-TPP outperform the STBA variants, as they cover all the targets with only 8 UAVs. Conversely, the STBA variants require at least 10 UAVs to cover all the targets, and achieve lower performance. OPT-A<sup>2</sup>-TPP covers 15 – 20% more targets than STBA algorithms, in all the scenarios, completing 69% of tasks (using 10 UAVs), while the best variant M-STBA accomplishes only 42% of tasks with the same number of UAVs. We can notice how GREEDY-A<sup>2</sup>-TPP performs better as quickly as number of UAVs grow reaching similar performance of OPT-A<sup>2</sup>-TPP.

3) *Robustness to Channel Errors*: In Figure 9 we plot the percentage of accomplished tasks by varying the probability of channel error  $\psi \in \{0, 0.05, 0.1, 0.15\}$ , in a setting with 20 UAVs and 4 targets. Both OPT-A<sup>2</sup>-TPP and GREEDY-A<sup>2</sup>-TPP are the most robust algorithms, increasing their improvement with respect to STBA variants. **OPT-A<sup>2</sup>-TPP completes up to 170% more tasks than the other approaches.** On the other hand, M-STBA and H-STBA experience severe delays and drastic performance reduction due to frequent TCP re-transmissions, which introduces additional data in the network, further overloading communication links.

4) *Scalability*: Figure 10 investigates the percentage of accomplished tasks in a scenario with 50 UAVs, varying the number of targets from 10 to 50. We do not include the OPT-A<sup>2</sup>-TPP when the targets are more than 20, due to prohibitive computational time. This result underlies the huge benefit introduced by the polynomial time solution GREEDY-A<sup>2</sup>-TPP, which scales gracefully when the problem instance grows in complexity. The figure shows that GREEDY-A<sup>2</sup>-TPP has near optimal performance with 10 targets, accomplishing 63% of the tasks, while L-STBA accomplishes only 38% of them. All the algorithms have a slightly decreasing trend as the number of targets increases, as the UAVs have to offload more tasks with possible network congestion and missed deadlines. The STBA variants quickly drop their performance due to



congestion and long delays, while GREEDY-A<sup>2</sup>-TPP is able to keep satisfactory performance around 50%, trading off compression and accuracy to cover all targets and offload their data. With 50 targets GREEDY-A<sup>2</sup>-TPP accomplished 5 times the tasks of the best STBA variant. Finally, in Figure 11 we investigate the computational time. We restrict the time to a maximum of 5 hours (18000 seconds), and we consider no solutions after that time. We consider a general STBA instance without compression levels, as they do not affect the execution time. While OPT-A<sup>2</sup>-TPP has very huge computational times even with 10 targets, **GREEDY-A<sup>2</sup>-TPP is 15x faster than the STBA solutions.**

### C. Experimental Testbed Results

We now evaluate the performance through our experimental testbed. **The testbed is composed of 4 UAVs and an edge server with dedicated GPU.** We emulate missions with up to 4 targets. We run 10 experiments for each scenario and we average the results. Each UAV includes a DJI Mavic Air 2 drone, mounting a Raspberry PI 4 model B and a power bank, as shown in Figure 12. The on-board Raspberry PI, powered by the power bank is used to generate and pre-process tasks, and to offload them to the edge according to the optimization plan. For repeatability and to emulate different scenarios, we sample images from the ImageNet dataset [29].

Field	*****
Time	9:00-18:30 a.m.
Temperature	+4 - 15°C
Wind Speed	0.0 to 4.3 m/s
Field Size	65x35(meters)
Nr. Of UAVs	4
Nr. Of Targets	Variable (from 1 to 4)
Humidity	60% - 77%
AMSL	2 meters

TABLE II: Experimental Setting

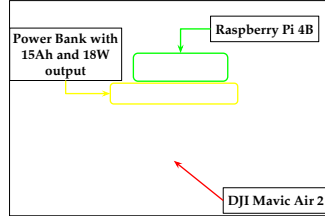


Fig. 12: UAV implementation.

The edge server is a Jetson Nano board, used to run the DL models and execute tasks. It mounts a Raspberry PI for computation and communication. TCP links are established for reliable connectivity. Considering the limited capabilities of the edge server, we execute only ResNet-50 and MobileNet-V2 on the Jetson Nano, which have approximately 0.03 seconds of inference time [30]. For DensNet-201, ResNet-152, and YoloV4, we used a laptop with an NVIDIA RTX-2060 Graphics Processing Unit (GPU). In the experiments, we consider up to 4 targets placed at around 15 meters from the edge. Table 11 reports the experimental settings in the *Urban* scenario. The first set of experiments evaluates the impact of increasing the number of targets (from 1 to 4), with MobileNet-V2. Figure 13 shows the percentage of accomplished tasks at the edge-server with a task deadline of  $\Delta = 0.4\text{sec}$ . The plot shows that the OPT-A<sup>2</sup>-TPP finds the best trade-off between accuracy and data compression. It completes more than 67% of the tasks, independently of the number of targets. This is close to the maximum performance achievable with the DL model (i.e., 78%), represented by the blue horizontal line. **GREEDY-A<sup>2</sup>-TPP instead reaches up to 57% accomplished task, with**

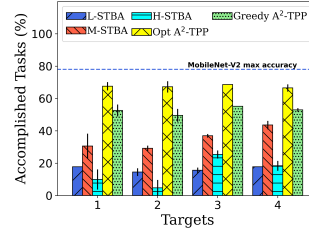


Fig. 13: Accomplished Tasks (%) at increasing targets, MobileNet-V2,  $\Delta = 0.4\text{sec}$

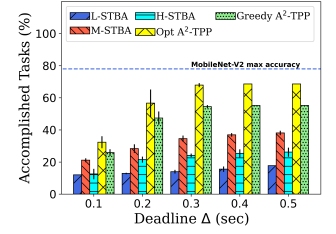


Fig. 14: Accomplished Tasks (%) at increasing of  $\Delta$ , using 3 targets and MobileNet-V2

**a 20% average improvement over the best STBA version (M-STBA).** Conversely, the best STBA variant (i.e., M-STBA) does not complete more than 46% of tasks, independently of the number of targets. In particular, with 2 targets, all STBA variants perform very poorly, completing less than 30% of tasks. The superiority of A<sup>2</sup>-UAV in both the approaches (Opt and Greedy) is confirmed by results on the percentage of accomplished tasks by varying the deadline  $\Delta \in [0.1, 0.5]$  (see Figure 14). OPT-A<sup>2</sup>-TPP reaches an improvement over the percentage of executed tasks with respect to M-STBA up to 76% when  $\Delta = 0.3\text{sec}$ . The GREEDY-A<sup>2</sup>-TPP approach instead improves M-STBA results (when  $\Delta=0.3\text{sec}$ ) around 50% upholding our intuition. We investigated the performance of GREEDY-A<sup>2</sup>-TPP and OPT-A<sup>2</sup>-TPP also when other DL models are applied. Table 11 summarizes the results in the case of  $\Delta = 0.3\text{sec}$  and 4 targets, for ResNet50, MobileNet-V2 (executed on the Jetson Nano) and ResNet152, DenseNet201 and YoloV4 (executed on a laptop with a dedicated GPU). The results show that both our solutions outperforms all STBA variants independently of the applied model. In particular, with DenseNet201 OPT-A<sup>2</sup>-TPP has the best performance.

DL Model	OPT-A <sup>2</sup> -TPP	GREEDY-A <sup>2</sup> -TPP	L-STBA	M-STBA	H-STBA
ResNet50	<b>66.64</b>	62.88	25.42	36.14	21.86
ResNet152	<b>67.89</b>	65.27	29.93	37.96	24.70
DenseNet201	<b>70.17</b>	68.33	32.92	41.87	26.99
MobileNet-v2	<b>69.29</b>	57.36	18.37	38.94	21.91
YoloV4	<b>59.32</b>	51.3	15.22	31.52	17.18

TABLE III: Percentage of Completed Tasks,  $\Delta = 0.3\text{sec}$

## VI. CONCLUSIONS

In this paper we proposed A<sup>2</sup>-UAV, a novel *application-aware* optimization framework for reliable and effective DL task offloading in multi-hop UAVs networks. For the first time, we considered the *accuracy* and *delay* requirements of the specific DNN tasks, to jointly optimize task assignment and offloading. Through extensive simulation, we demonstrated that A<sup>2</sup>-UAV is able to deal with different network conditions, maximizing the application performance at the edge. A<sup>2</sup>-UAV outperforms existing approaches, getting an average improvement w.r.t. the state-of-the-art algorithm of 38%. We finally validated our solution through real-field experiments, considering four DJI Mavic Air 2 UAVs and a Jetson Nano board as edge server. We share datasets and code with the research community to allow reproducibility.

## VII. ACKNOWLEDGEMENT OF SUPPORT AND DISCLAIMER

This work is funded in part by the G5828 "SeaSec: DroNets for Maritime Border and Port Security" project under the NATO's Science for Peace Programme, by the National Science Foundation (NSF) grant CNS-2134973 and CNS-2120447, as well as by an effort sponsored by the U.S. Government under Other Transaction number FA8750-21-9-9000 between SOSSEC, Inc. and the Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory, the U.S. Government, or SOSSEC, Inc.

## REFERENCES

- [1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [2] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
- [3] M.-A. Messous, S.-M. Senouci, H. Sedjelmaci, and S. Cherkaoui, "A game theory based efficient computation offloading in an UAV network," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4964–4974, 2019.
- [4] J. Scherer and B. Rinner, "Multi-robot persistent surveillance with connectivity constraints," *IEEE Access*, vol. 8, pp. 15 093–15 109, 2020.
- [5] N. Bartolini, A. Coletta, M. Prata, and C. Serino, "On connected deployment of delay-critical fanets," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 9720–9727.
- [6] M. Samir, C. Assi, S. Sharafeddine, D. Ebrahimi, and A. Ghreyeb, "Age of information aware trajectory planning of UAVs in intelligent transportation systems: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 12 382–12 395, 2020.
- [7] Y. Chen, N. Zhao, Z. Ding, and M.-S. Alouini, "Multiple UAVs as relays: Multi-hop single link versus multiple dual-hop links," *IEEE Transactions on Wireless Communications*, vol. 17, no. 9, pp. 6348–6359, 2018.
- [8] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109–2121, 2018.
- [9] S. Hosseinalipour, A. Rahmati, and H. Dai, "Interference avoidance position planning in dual-hop and multi-hop UAV relay networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 11, pp. 7033–7048, 2020.
- [10] S. Chuprov, L. Reznik, A. Obeid, and S. Shetty, "How degrading network conditions influence machine learning end systems performance?" in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2022, pp. 1–6.
- [11] B. Yang, X. Cao, C. Yuen, and L. Qian, "Offloading Optimization in Edge Computing for Deep Learning Enabled Target Tracking by Internet-of-UAVs," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9878–9893, 2020.
- [12] D. Callegaro, M. Levorato, and F. Restuccia, "SeReMAS: Self-Resilient Mobile Autonomous Systems Through Predictive Edge Computing," *arXiv preprint arXiv:2105.15105*, 2021.
- [13] W. Chen, Z. Su, Q. Xu, T. H. Luan, and R. Li, "VFC-based cooperative UAV computation task offloading for post-disaster rescue," in *IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2020, pp. 228–236.
- [14] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 248–255.
- [15] A. C. Bovik, *Handbook of image and video processing*. Academic press, 2010.
- [16] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," *CoRR*, vol. abs/1801.04381, 2018.
- [17] A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, 2020.
- [18] L. Bertizzolo, S. D'oro, L. Ferranti, L. Bonati, E. Demirors, Z. Guan, T. Melodia, and S. Pudlewski, "Swarmcontrol: An automated distributed control framework for self-optimizing drone networks," in *IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2020, pp. 1768–1777.
- [19] M. T. Rashid, D. Y. Zhang, and D. Wang, "Socialdrone: An integrated social media and drone sensing system for reliable disaster response," in *IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2020, pp. 218–227.
- [20] N. Bartolini, A. Coletta, G. Maselli *et al.*, "A multi-trip task assignment for early target inspection in squads of aerial drones," *IEEE Transactions on Mobile Computing*, vol. 20, no. 11, pp. 3099–3116, 2021.
- [21] X. Wang and L. Duan, "Dynamic pricing and capacity allocation of UAV-provided mobile services," in *IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2019, pp. 1855–1863.
- [22] T. Kimura and M. Ogura, "Distributed collaborative 3d-deployment of UAV base stations for on-demand coverage," in *IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2020, pp. 1748–1757.
- [23] E. Natalizio, N. R. Zema, E. Yanmaz, L. D. P. Pugliese, and F. Guerriero, "Take the field from your smartphone: Leveraging UAVs for event filming," *IEEE Transactions on Mobile Computing*, vol. 19, no. 8, pp. 1971–1983, 2019.
- [24] D. Tateo, J. Banfi, A. Riva, F. Amigoni, and A. Bonarini, "Multiagent connected path planning: PSPACE-Completeness and how to deal with it," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [25] T. N. Nguyen, B.-H. Liu, and S.-Y. Wang, "On new approaches of maximum weighted target coverage and sensor connectivity: Hardness and approximation," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, pp. 1736–1751, 2019.
- [26] G.-H. Lin and G. Xue, "Steiner tree problem with minimum number of steiner points and bounded edge-length," *Information Processing Letters*, vol. 69, no. 2, pp. 53–57, 1999.
- [27] F. Senel and M. Younis, "Relay node placement in structurally damaged wireless sensor networks via triangular steiner tree approximation," *Computer Communications*, vol. 34, no. 16, pp. 1932–1941, 2011.
- [28] nsnam, "Network Simulator-3 (NS-3)," 2021. [Online]. Available: <http://www.nsnam.org/>
- [29] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [30] Nvidia, "Jetson nano: Deep learning inference benchmarks," 2021. [Online]. Available: <https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks>