Deep Learning Models for Gesture-controlled Drone Operation

Tahajjat Begum, Israat Haque, and Vlado Keselj
Department of Computer Science, Dalhousie University, Halifax, Canada
Email: tahajjat.begum@dal.ca, israat@dal.ca, vlado@cs.dal.ca

***Abstract*—Recently Unmanned Aerial Vehicles (UAVs) or Drones have gained enormous attention in applications like military, agriculture, industry, etc. One approach of controlling the operation of a drone is using hand gestures, which enables designing a low-cost system. However, the accuracy of such a system highly depends on the gesture recognition models. We can use a neural network-based gesture recognition model, which is a widely accepted image recognition scheme. In this work, we first design three deep neural network-based gesture recognition models: simple Convolutional Neural Networks (CNN), VGG-16, and ResNet-50 to uncover the best model for drone control. We evaluate the proposed models over our generated hand-gesture images in terms of their accuracy, precision, and complexity. The analysis reveals that each of the three models has its advantages and disadvantages while balancing between accuracy and complexity. For example, Simple CNN offers 92% accuracy on the testing set validation with the lowest validation loss compared to VGG-16 and ResNet-50. Thus, users can choose one of the proposed models to match their drone application.**

***Index Terms*—Convolutional Neural Networks (CNN), Human-Computer Interaction (HCI), Unmanned Aerial Vehicles (UAV).**

I. Introduction

The usage of drones is not limited to the military use anymore; instead, the popularity is increasing in different applications like aerial photography, shipping-and-delivery, entertainment, law enforcement, wildlife monitoring, search-and-rescue, precision agriculture, disaster management, storm tracking, etc. [1], [2]. According to the Federal Aviation Administration, the drone market will reach 17 billion by 2024, and 7 million drones will reach the sky [3]. Today we are experiencing the evolution of drone technology from one generation to another. The next generation of smart drones called Solo is already in the market to capture visual images [4]. The remote-controlled drones are gradually transforming into semi or completely automated devices exploiting the Artificial Intelligence (AI) based implementation.

Humans usually use hand and body gestures to communicate; e.g., face or hand motion. Thus, gesture recognition can enable man-machine interactions to realize human-controlled drone operation. Controlling drones using hand gestures can give humans an edge to directly interact with the drones and avoid additional hardware like remote control, which incur an additional cost. A hand gesture-controlled drone can also minimize physical labor for humans. As the hand gestures can be comprised of images or live video streaming, we can design a system for vision-based drone control that can reduce cost and avoid any interruptions in the drone operation due to the failure of hardware like a remote control. Furthermore, the vision-based drones will be able to provide better data extraction facility than remote-controlled ones in terms of flexibility and ease of use [5].

We consider a vision-based gesture recognition system to control the operation of a drone. In particular, we focus on applications like navigation, surveillance, and training, where users do not require a remote control. Instead, they can use hand gestures to control the drone operation. For instance, DJI Spark is a hand gesture-controlled drone that helps immature pilots learn flying [6]. The system consists of an image recognizer to capture gesture images to feed to an image processing unit. The drone controller reacts to the image processing unit's outcome, i.e., the controller controls a drone in real-time following users' gestures. Thus, the core component of such a gesture-controlled system is the image processing unit and corresponding accuracy.

A deep neural network, such as a convolution neural network (CNN), is widely recognized as an accurate image classification algorithm because of its automatic feature extraction capability [7]. In particular, CNN learns features by studying complex hidden layers. It can also effectively reduce the growing number of parameters without compromising the model accuracy [8], [9]. Over time researchers have proposed different CNN architectures for better accuracy, processing time, and model complexity. We select three architectures varying in the number of fully connected layers. For example, we consider a basic CNN architecture with fifteen layers, medium-sized VGG-16, and a large neural network Resnet-50. We plan to explore their trade-off in accuracy, complexity, and resource requirement (CPU, memory, etc.), and based on these recommendations; drone users can choose an appropriate model to meet their application requirements and available resources.

The research in neural network-based drone control is still in its infancy except for a couple of schemes. Hu *et al.* [10] present a framework for neural network-based drone control, where different images of hand-gestures are classified using an eight-layer CNN. We extend their work by incorporating VGG-16 and Resnet-50 and a 15-layer CNN in the image classification module of the drone control system to generalize it for various applications. In addition to measuring the accuracy, we consider precision, recall, and F1 value in our model evaluation,

978-3-903176-31-7

which are missing in [10]. Hadri *et al.* [11] use simple CNN with varying number of fully connected layers. We offer the above three different neural networks that users can choose from depending on their application demand. Thus, we present an extensive evaluation and comparison of these three models: simple CNN, VGG-16, and Resnet-50.

We consider drone applications where simple hand gestures like left, right, stop, and forward can control drones for navigation, surveillance, or training (e.g., an immature pilot). However, any required hand gestures can be accommodated in our drone control system. Sensors from a collector (e.g., Leap Motion Controller) can capture the gestures to feed to the image classification module, a neural network in our design. We collect around five hundred and fifty hand gestures from different locations at Halifax (NS, Canada) over twenty days. Seventeen people participated in the image collection phase to generate images in eight different light intensity (day and night) for each gesture. We then use 80% of data to train the above neural networks and the rest of the data for testing. We used different people and their images for the validity of the training and testing dataset to ensure the testing images are entirely different from the images produced for training.

We measure the accuracy, recall, precision, and F1 values of the three CNNs. The accuracy of the training dataset for VGG-16 was 100%; however, the validation accuracy and loss value are worse compared to simple CNN. Based on precision, recall, validation accuracy, and lose value, simple CNN offers a better result. We notice that the validation accuracy of simple CNN increases with the increasing epoch measure while it decreases for the other two architectures. Also, the validation loss is the lowest in simple CNN compared to VGG-16 and Resnet-50. It offers 92.5% accuracy on the testing dataset. We suspect that simple CNN offers good accuracy in the case of a small dataset compared to the other two architectures.

The rest of the paper is organized follows. Section II provides the necessary background. The following section presents the related work. We outline our methodology in Section IV, and the next section presents the evaluation results. We discuss the future research directions in Section VI following the concluding remarks.

II. Background

This section presents the necessary background on the operation of a vision-based drone and three neural networks that we deploy in this work.

Drones Fundamentals. A drone is an Unmanned Aerial Vehicle (UAV) without any onboard pilot. The core component is its flight controller hardware that consists of various sensors like accelerometer and gyroscope and firmware to control the drone movement. The drone control system also requires a ground base station consisting of essential software to install and set up the firmware at the flight controller. It also calibrates different parts of the drone. The base-station and the flight controller communicate over a standard protocol called *Micro Air Vehicle Communication Protocol (MAVLink)* [11].

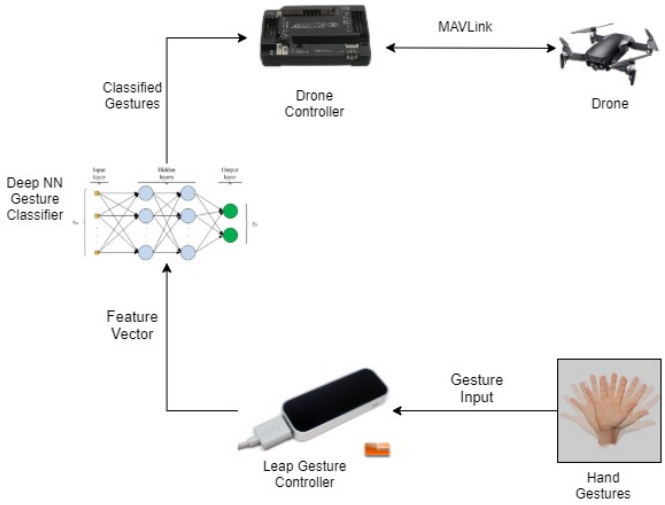

Fig. 1: An example of a gesture-controlled drone system.

In a vision-based drone control system, we need another component to capture and process the gestures. There are two approaches to capture gestures: a front-facing camera mounting on a drone or camera connected to the ground base-station. In this study, we consider the second option to capture gestures. For example, a Leap Motion Controller can capture the gestures to feed to the learning-based image processing unit. A trained CNN model then classifies the captured gestures to send an appropriate control signal from the base station to the flight controller over MAVLink. The entire vision-based drone control system is depicted in Fig 1.

Deep Learning Fundamentals. The deep learning methodology is mostly based on artificial neural networks that are computational models inspired by the human brain structure. The neurons are usually organized into several layers, which are the core entity of a neural network. Each layer has a collection of neuron nodes, where the information processing takes place. The information is transferred from one layer to another over connecting channels called the *weighted channel*, where the neurons include a unique bias. The bias is added to the weighted sum of inputs reaching the neurons, which later pass through an activation function to activate neurons and compute the output value. The output of one layer connects to the next one until it reaches the second last input layer. Each neuron can consist of one or more output connections that process information as signals to the next layer. The weight and bias are adjusted throughout the network to produce a well-trained neural network, which can recognize patterns.

The forward propagation of information through a neuron defines a set of inputs such as x_1, x_2, x_3, ..., x_m in Fig 2, which has corresponding weights w_1 through w_m, respectively. The weighted sum of the input passes through a non-linear activation function to produce the final output y. Bias allows the shift of activation function to the left or right regardless of the input. Sigmoid Function, Hyperbolic Tangent, Rectified Linear Unit, SoftMax, etc. are the common types of non-

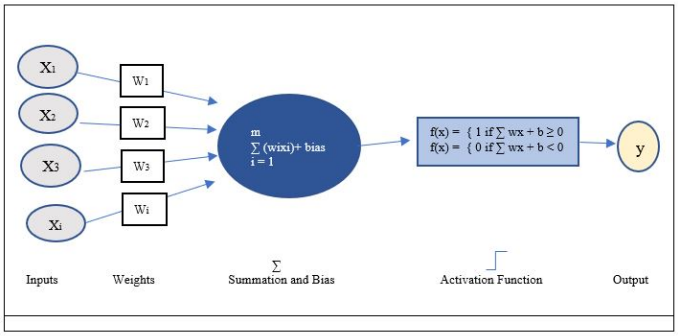

Fig. 2: An example of artificial neuron computation.

linear activation functions in neural networks. Each neuron's value is calculated for each hidden layer based on the weight, bias, and activation function [12]. The activation function introduces non-linearities into the network, allowing approximating complex functions arbitrarily into decision boundaries, making the neural network a powerful method to detect classes. The number and arrangement of layers distinguish neural network architectures. The neural network is the most efficient method to process unstructured data, such as images. As CNN is a widely utilized deep learning method to analyze visual images, this paper focuses on comparing three different CNN architecture for gesture recognition, which can be implemented in the drone control system.

Simonyan *et al.* proposed Very Deep Convolutional Networks for Large-Scale Image Recognition (VGGNet) [13], which outperformed AlexNet, the previous state-of-the-art. The authors showed that having more layers with smaller convolutional kernels can increase a neural network's accuracy. However, optimizing models with many layers is inherently difficult because of the vanishing and exploding gradient problems. The Residual networks (ResNet) [14] address this gradient problem. As the name suggests, residual functions form the building block for these models (see Fig.**??**). Explicitly adding the identity function to deepereach convolutional layers appears to be beneficial during training and improve accuracy. ResNet-50 network achieves this by adding a skip layer and identity mapping to add up the previous layer to approximate the final function $F(x) + x$.

III. Related Work

In this section, we present and discuss existing literature related to our proposed design.

The deep neural network-based image classification for drone operation is getting its momentum in academia and industry. Industries have started implementing drones in their services and proposing to build advanced drone support systems. For instance, Google AI Blog announces a new framework for hand and finger tracking, which can be used for real-time hand perception experiments [15]. They also introduce another solution to train their provided model to recognize images, poses, or sound [16]. However, these designs are not for drone applications.

We notice a couple of drone control systems using neural networks, specifically CNN. Hu and Wang [10] present a state-of-the-art gesture recognition system to control a UAV. In their experiment, the authors focus on three deep neural networks to recognize dynamic hand gestures. They use skeleton data from a leap motion controller and split the data among training, validation, and testing. The former dataset is used to train three neural networks: 2-layer and 5-layer neural networks and an 8-layer convolution neural network. The evaluation results reveal that CNN offers the highest accuracy among the three models that can be used to control a drone in real-time.

Hadri proposes a similar drone control system deploying VGG-16 as the image classifier [11]. In particular, the system uses the Single Shot MultiBox Detector (SSD), which is based on VGG-16, to detect hand gestures. The gestures are grouped as a single to five fingers and a close wrist. The author also discusses different hardware and software-based drone controlling schemes. However, their accuracy is lower than the one we report in this work. We compare and contrast simple and complex CNNs to allow users to select an appropriate model based on their available data volume and resources.

IV. Methodology

This section first describes the data collection process. Then, we focus on presenting data preprocessing and decomposition for model training and testing.

A. Data Collection and Preprocessing

We captured images (dataset) around Halifax, Nova Scotia, Canada, in indoor and outdoor settings, where both long and short distance images are considered to increase posture trajectory. We captured total 544 images in different light conditions, where 17 people participated in the image capturing. The dataset includes four types of gestures: forward, stop, right, and left. Each person provided a total of 32 images in 8 different lights for each gesture. We used both mobile and webcams to capture images, and saved in JPG format. Then, we rescaled the collected images of different sizes into 128×128 pixels as part of the preprocessing. We then split 544 images into 80% and 20% for training and testing, respectively. In particular, we use 344, 88, and 68 images as training, testing, and validation, respectively. Fig 3 represents the sample of forward, stop, right, and left gesture images. Thus, our CNN models need to detect four classes of hand gestures.

B. Deep Learning Models

In our experiment, we started with pre-trained VGG-16 and ResNet-50 models. However, we created simple CNN based on a basic CNN architecture. We calculated model parameters, including the number of epochs, optimizer, number of dense layers, neurons in the dense layers, activation function, and dropout layer to deal with the model overfitting problem. For VGG-16, we started with a pre-trained VGG architecture with 18 weight layers as input, one pooling layer, and three dense

Fig. 3: Examples of different gestures.

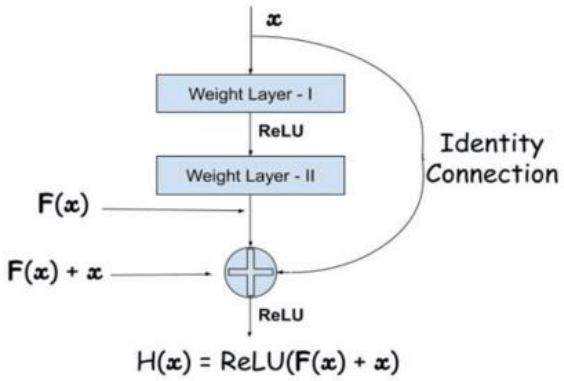

Fig. 4: ResNet-50 architecture.

layers; however, to fit our problem, we changed the output layer into four classes: right, left, forward, and stop. In the original VGG-16 architecture, the width of convolution layers (the number of channels) is relatively small. It starts from 64 in the first layer and then increases by a factor of 2 after each max-pooling layer until it reaches 512. We followed the same procedure for ResNet-50 (see Fig.4), i.e., we used a pre-trained ResNet model and kept the same number of input layers while changed the output layers based on our classification requirement.

C. Training and Testing

We did not transfer images into gray-scale instead used the color ones. After the preprocessing step, which was done using Keras library, the training data is used to build different CNN models. In simple CNN, the 128×128 dimension was set as width and height and used Convo2d model. We used ReLu function as an activation function for hidden layers and the Soft-Max activation function in the final output layer for multiclass problems. We tested a total of 831780 parameters during the training of the simple CNN. A similar process has been followed for the VGG-16 model; however, the default image size for VGG-16 is set to 224×224. VGG-16 is a published pre-trained model that was trained on the ImageNet, which showed excellent accuracy result. We simply imported that model; however, the output layer is modified to fit our problem, four classes. For the VGG-16 model, a total of 527364 trainable parameters were available to learn within the network. Adam optimization algorithm was used to optimize the model and train the deep neural network and used 32 batch sizes to train the model. Likewise, ResNet-50 used a pre-trained model built on ImageNet with 128×128 weight and height and 16 batch size, half of the VGG-16 model, and resulted in 34609156 learnable parameters.

All three models used the same activation functions, and ResNet-50 and VGG-16 use the same optimizer. After training all three deep neural network models, the accuracy of the model is tested using the 88 testing sample, which was separated from the training dataset to ensure the model validity is based on an unknown dataset. However, before examining the model using the testing dataset, the validation dataset is used for VGG-16 and ResNet-50 model to reduce the overfitting problem. Testing the model helped to understand the learning rate or the need for epochs size adjustment. We used 30 epoch size for Simple CNN and VGG-16; however, the epoch size for the ResNet-50 model was adjusted during the experiment for an improved result. Multiple epochs update the weights of the model as network biases, and weights are trained to ascertain the final output layer. The accuracy and validation increased at the end of epoch training for all the models, which will be discussed in the next section.

V. Evaluation

In this section, a complete discussion of three trained models is presented in terms of accuracy, error, and robustness of each model.

A. Model Comparison

It is essential to fine-tune weights to reconcile the difference between the actual and predicted outcomes as weights decide how quickly the activation function will react. We can adjust weights using back-propagation to improve the accuracy prediction by calculating the loss function gradient. The validation and training error are the main two factors to determine the training time. Model training usually continues until these two errors start dropping. The increase in the validation error, however, indicates overfitting. Usually, when validation error starts to increase, the training needs to be terminated. Another factor is the number of epochs, which depends on the dataset, and it controls the number of complete passes of the learning algorithm through the training dataset. The epochs allow the learning algorithm to continue looping until it reaches the minimal model error. In our experiment, the number of epochs is chosen based on the accuracy and validation loss, which is 30. However, this hyperparameter can be tested for different

TABLE I: Comparison of the Three Models

Model	Simple CNN	VGG-16	RestNet50
Training accuracy	0.97	1	0.9389
Training loss	0.119209e-08	0.005770	0.6531
Validation accuracy	0.9250	0.9545	0.8295
Validation loss	0.11921e-08	0.1556	0.7129
Optimizer	rmsprop	adam	adam

TABLE II: Simple CNN Classification Performance.

	Precision	Recall	F_1-Score	Support
Forward	0.70	0.33	0.29	21
Left	0.29	0.43	0.34	23
Right	0.00	0.00	0.00	22
Stop	0.07	0.05	0.05	22
Avg/Total	0.15	0.20	0.17	88

TABLE III: VGG-16 Classification Performance.

	Precision	Recall	F_1-Score	Support
Forward	0.16	0.14	0.15	21
Left	0.27	0.26	0.27	23
Right	0.17	0.18	0.17	22
Stop	0.30	0.32	0.31	22
Avg/Total	0.23	0.23	0.23	88

TABLE IV: ResNet-50 Classification Performance.

	Precision	Recall	F_1-Score	Support
Forward	0.26	0.33	0.45	21
Left	0.58	0.83	0.68	23
Right	0.67	0.55	0.60	22
Stop	0.78	0.67	0.65	22
Avg/Total	0.68	0.67	0.65	88

epochs numbers to compare validation losses for a single model for a better comparison.

Table I provides the accuracy and loss comparison for the three models presented in this paper. The results confirm that Simple CNN offers the lowest training loss, but VGG-16 has 100% or 1.0 accuracy on its training data. In Simple CNN, the accuracy increases with the increasing epoch measure, whereas the validation accuracy decreases, which indicates the model fits the training set better. Furthermore, it has the lowest validation loss compared to the other two models and offers 92.5% accuracy on the testing dataset. Thus, Simple CNN best fits to recognize gesture images better than the other two models in a small dataset like ours.

Furthermore, compared to VGG-15 and ResNet-50, Simple CNN offers better precision and recall value, presented in Table III, Table II, and Table IV. Model accuracy represents the number of accurate predictions; however, model accuracy is not the right predictor to ensure the validity and reliability of the model. We need to dive deeper into the confusion matrix to comprehend other classification matrices. It is very crucial to predict the number of correct and incorrect predictions. We calculate precision to calculate the positive predictions of the model, and recall calculates the model reliability to predict positive outcomes. The combination of precision and recall is represented by F1 score, which provides a weighted average of precision and recall [17]. We can understand the number of actual occurrences of the class in the specified dataset by checking the support. Simple CNN has better precision for forward and stop gestures with an average of 68% error rate to label a negative instance as positive. Table II presents that the Simple CNN results in 67% recall value, which means that in 68% of the cases, Simple CNN can find positive instances compared to the other two models. Table II also shows that the left and stop gestures positive instances are identified by the Simple CNN at the rates of 83% and 95%, respectively.

We also investigated the training and validation accuracy and loss for all three models, where we observe an upward accuracy trend at the end of the epoch for both training and testing data. However, VGG-16 shows the highest accuracy for both testing and training datasets compared to the other

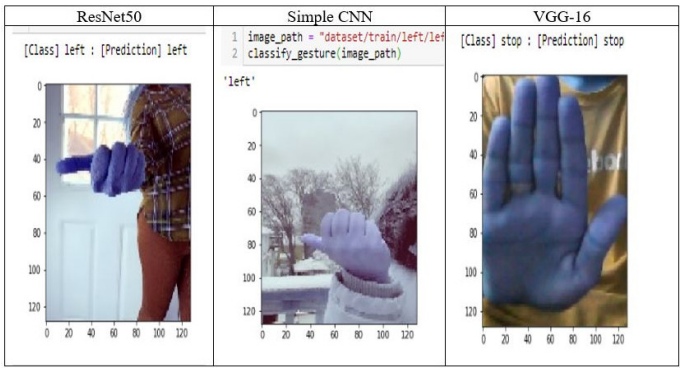

Fig. 5: Gesture prediction of the three models.

two models. The loss presents a fascinating insight, mainly for Simple CNN. The loss value for simple CNN is better than VGG-16 and ResNet-50. These two models have fluctuation in loss value for both training and testing datasets; however, simple CNN presents similar and very lower loss value for training and testing. The validation and accuracy curves can stipulate epoch closure signs if both training and testing values start to depart consistently. These accuracy and loss values also support Simple CNN as a more accurate gesture detection algorithm in our case.

Fig.5 presents the prediction performance of the three models, which indicates that all three models can accurately predict different gestures. In particular, we present the prediction outcome of left and stop gestures, which are predicted correctly.

B. Model Test in a Simulator

In this experiment, we integrate the trained models in a simulator using the Python Turtle library. The Python Turtle module provides a drawing window where shapes can be drawn with simple repetitive moves. Fig.6 presents the simulation process: shapes are moving in the Turtle simulation environment during the implementation process. All the four gestures: right, left, stop, and forward, are identified and labeled by the windows command prompt and Turtle simulation. We also plan to use graphical software in this simulator in the future. Finally, we plan to extend this simulation in a testbed deployment, which we discuss in the following section.

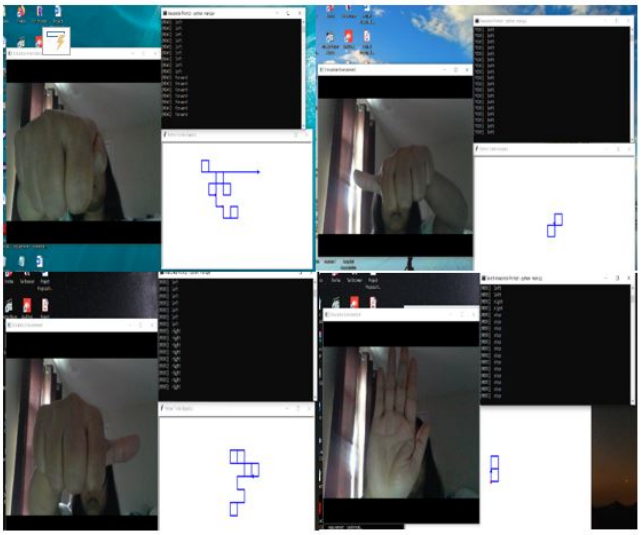

Fig. 6: Gesture prediction in a simulation environment.

VI. Discussion

In this section, we outline a couple of limitations of our work and discussion on how to mitigate those in the future research.

Dataset. The current dataset includes around 550 gestures as we could not collect more because of difficulties reaching out to more participants. We did not use publicly available dataset as our goal is to generate a dataset for the target applications from diverse participants (by gender, age, height, etc.). The CNN models are usually trained on large datasets, especially ResNet-50 and VGG-16 models. Thus, we plan to conduct another data collection cycle to gather thousands of gestures from a wide range of participants. We will then test these architectures to see the impact of the size of the dataset on their performance. Another plan is to explore extensively different hyperparameters and epochs for the three models as they have a different number of hidden layers; they may converge to the optimal solution for different hyperparameters and epochs.

CNN based drone system. We plan to integrate the proposed gesture recognition model in a simulator and real testbed. For example, we can consider a system presented in Fig.7. In that system, a leap motion controller acts as a gesture sensor to capture hand images. These images then feed to the proposed CNN based image recognition module. The model outcome next goes to a Raspberry Pi serving as a ground station. The entire controlling system can be managed using a Python script. For example, Olympe by Parrot developer [18] provides a Python interface to connect and control a drone in a simulator and testbed, so users can write their own control applications. We can use a customized Python script to control a drone (e.g., Parrot AR drone). The drone will have a Raspberry Pi controller attached to it, which will process all the gesture commands via WiFi networks. We plan to test the proposed system with the integrated deep NN based gesture recognition module.

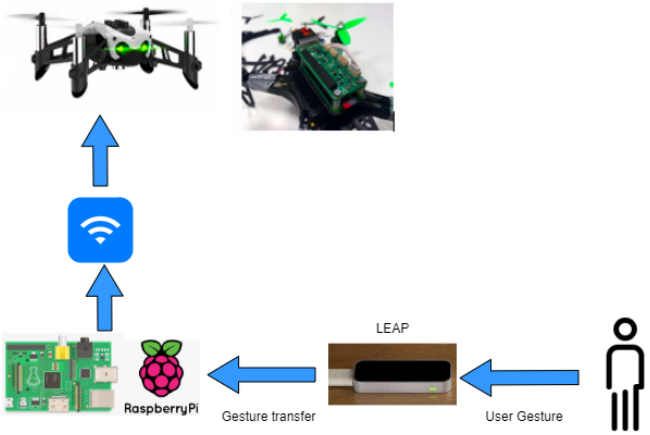

Fig. 7: A drone control system.

VII. Conclusion

In this paper, we have focused on the gesture-controlled drone operation, where gestures are composed of hand images. The core of this drone control system is its gesture recognition component. The accuracy of the recognition system has a crucial impact on the drone movement. CNN is widely recognized as an accurate image recognition solution. Thus, we have implemented and compared the three CNN architectures: simple CNN, VGG-16, and ResNet-50. We have considered the most common hand gestures: right, left, forward, and stop to evaluate the performance of these architectures. The hand gestures are collected from different participants in a diverse environment (e.g., indoors, outdoors). We have tested the trained models over the Turtle simulator to recognize hand gestures in real-time. The evaluation results have shown that simple CNN has the best performance. The accuracy of the training dataset for VGG-16 is 100%, whereas the validation accuracy and loss value are lower compared to the simple CNN. We have also proposed a drone controlling system consists of a leap motion controller, gesture recognition module, and a Raspberry Pi to control an AR drone, which we envision to test as part of the future work.

Acknowledgements

We would like to thank the anonymous AnServApp reviewers for their constructive feedback. Also, we would like to thank Dr. Sageev Oore from Dalhousie University for his useful comments.

References

[1] N. Joshi, "10 stunning applications of drone technology," February 27, 2019. [Online]. Available: https://www.allerin.com/blog/10-stunning-applications-of-drone-technology

[2] D. Joshi, "Drone technology uses and applications for commercial, industrial and military drones in 2020 and the future," December 18, 2019. [Online]. Available: https://www.businessinsider.com/drone-technology-uses-applications

[3] G. Jeremy, "7 reasons why drones are the future of business," May 05, 2018. [Online]. Available: https://www.inc.com/jeremy-goldman/7-reasons-why-drones-are-future-of-business.html

[4] F. Stephen, "The next generation 3DR Solo smart drone takes flight," September 16, 2016. [Online]. Available: https://www.businessinsider.com/drone-technology-uses-applications

[5] A. A. A Kumar. Singha, A. Swarupa and D. Singh, "Vision based rail track extraction and monitoring through drone imagery," *ICT Express*, vol. 5, no. 4, pp. 250–255, December 2019.

[6] F. Jonathan, "7 reasons to choose the DJI spark," May 20, 2020. [Online]. Available: https://dronerush.com/reasons-to-choose-dji-spark-9807/

[7] P. Mishra, "Why are convolutional neural networks good for image classification?"

[8] E. AI and V. Alliance, "Vision processing opportunities in drones," September 15, 2016. [Online]. Available: https://www.edge-ai-vision.com/2016/09/vision-processing-opportunities-in-drones/

[9] A. Bonner, "The complete beginner's guide to deep learning: Convolutional neural networks and image classification," February 2, 2019. [Online]. Available: https://towardsdatascience.com/wtf-is-image-classification-8e78a8235acb

[10] B. Hu and J. Wang, "Deep learning based hand gesture recognition and uav flight controls," *International Journal of Automation and Computing*, vol. 17, pp. 17–29, 2020.

[11] S. Hadri, "Hand gesture for drone control using deep learning," 2018.

[12] N. Kang, "Introducing deep learning and neural networks — deep learning for rookies (1)."

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR 2015*, 2014.

[14] S. R. Kaiming. He, Xiangyu. Zhang and Jian.Sun, "Deep residual learning for image recognition," in *The IEEE conference on computer vision and pattern recognition*, 2016.

[15] V. Bazarevsky and F. Zhang, "Google AI blog," August 19, 2019. [Online]. Available: https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html

[16] Google, "Teachable machine," 2019. [Online]. Available: https://teachablemachine.withgoogle.com/

[17] J. Brownlee, "What is a confusion matrix in machine learning," August 15 2020. [Online]. Available: https://machinelearningmastery.com/confusion-matrix-machine-learning/

[18] P. Developers, "Parrot sdk," n.d. [Online]. Available: https://developer.parrot.com/