

Automatic Rat Detection using Edge Computing

Using Raspberry PI and K3S cluster

Team #6 Members

Anish Pokhrel(1394715)
Ashlesh Mithur(1386367)
Deepak Kumar(1400489)
Nidhi Nayak(1404524)
Shobhit Tiwari(1387366)
Pushpita Sarkar(1384152)
Arpan Kumar(1378650)

Under the guidance of

Prof. Dr. Christian Baun
Frankfurt University of Applied Sciences

Agenda

- Introduction
- Architecture
- Sensor Nodes
- Machine Learning Model
- Setting up K3S Cluster
- Rest API
- Web Application
- Slack Notification Service
- Deployments
- Demo
- Results

Introduction

- Developed edge computing solution to detect rats at sensor node
- Stored those results in cloud

Team Organization

Task	Contributors
Initial Hardware Setup ,Testing	Ashlesh Mithur, Arpan Kumar
K3S cluster, Sensor Node setup	Ashlesh Mithur
ML Model, Training, Sensor Node Deployment	Deepak Kumar
API development & DB setup	Anish Pokhrel
User Interface development	Shobhit Tiwari, Nidhi Nayak
Notification and Alerts	Anish Pokhrel, Pushpita Sarkar
Sensor Node and Cluster Integration	Deepak Kumar, Ashlesh Mithur
Project Integration	Anish Pokhrel, Ashlesh Mithur, Deepak Kumar
Documentation	Anish Pokhrel, Ashlesh Mithur, Deepak Kumar, Shobhit Tiwari

Software Stack

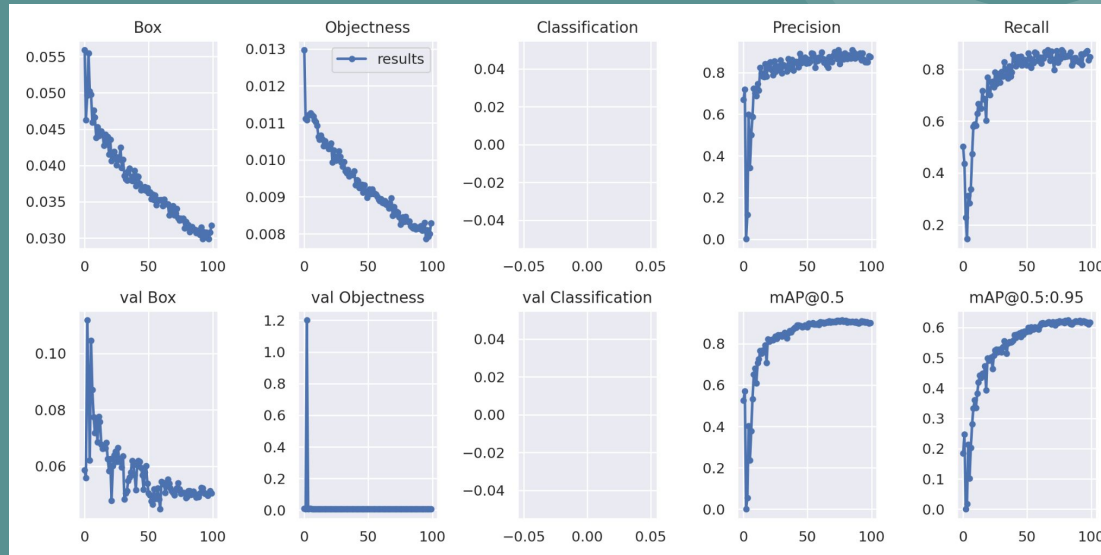
- Communication: Discord
- Notification Service: Slack
- Repository: Git <https://github.com/dpk0811/Rat-Detection>
- Scrum Board: Trello <https://trello.com/b/9EJAa3ZV/cloud-computing-project>
- ML framework: YOLOv7
- Backend Service: REST API based on Spring Boot
- Database: Postgres
- Training data: Scrapped data from internet
- Operating system: Raspberry Pi OS

Sensor Node

- Raspberry Pi 4 single board computer (SBC) with attached Raspberry Pi camera module 2
- Used as an edge computing device to detect rats
- Used to build relevant data and send it over to the REST API running on K3S cluster

Machine Learning Model

- Trained a machine learning model using YOLOv7 framework for automatic rat detection on the sensor node
- Used Google Colab to train machine learning models as it offers free computing resources
- Used 4500+ rat images and their labels to train model
- Splitted whole Dataset in a 70:30% ratio



K3S Cluster

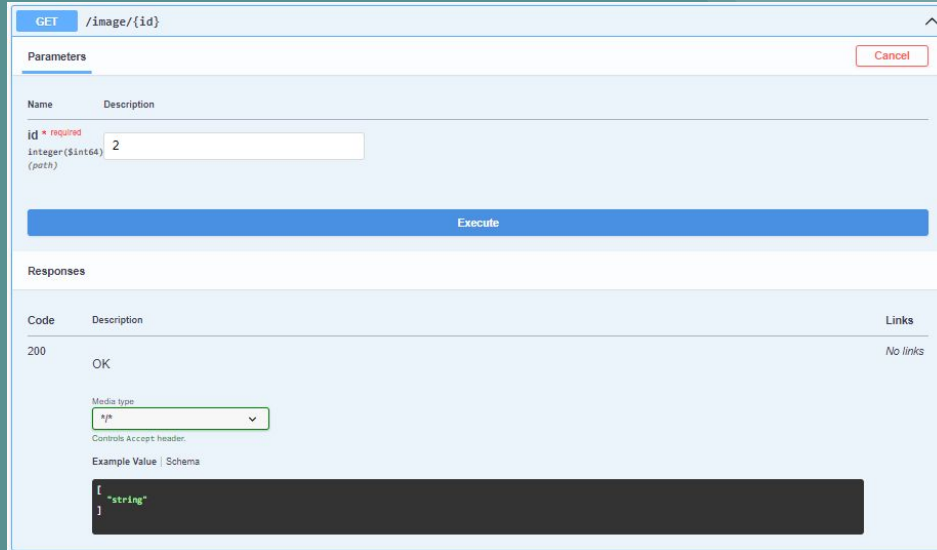
- Used 4 different Raspberry Pi 3 SBCs' to setup light weight Kubernetes cluster
- Created K3S cluster with 1 master node and 3 worker nodes
- Memory of each Raspberry Pi 3 ⇒ 32GB (SD Card)
- Flashed 32-bit Raspberry Pi OS manually by using Raspberry Pi Imager v1.7.3.
- Connected to the network via LAN switch.
- Generated token from master, which is used by all agent nodes while creating the K3S cluster.

```
PS C:\Users> kubectl get nodes -o wide
```

NAME	STATUS	ROLES	AGE	VERSION	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE	KERNEL-VERSION	CONTAINER-RUNTIME
k3smaster	Ready	control-plane,master	40m	v1.25.6+k3s1	192.168.0.10	<none>	Debian GNU/Linux 11 (bullseye)	5.15.61-v8+	docker://20.10.5+dfsg1
ksworker1	Ready	<none>	13m	v1.25.6+k3s1	192.168.0.11	<none>	Debian GNU/Linux 11 (bullseye)	5.15.61-v8+	docker://20.10.5+dfsg1
ksworker2	Ready	<none>	13m	v1.25.6+k3s1	192.168.0.12	<none>	Debian GNU/Linux 11 (bullseye)	5.15.61-v8+	docker://20.10.5+dfsg1
ksworker3	Ready	<none>	13m	v1.25.6+k3s1	192.168.0.13	<none>	Debian GNU/Linux 11 (bullseye)	5.15.61-v8+	docker://20.10.5+dfsg1

REST API

- Used to
 - process,
 - save, and
 - retrieve
 detection images
- Used to trigger notification service upon successful detections
- Technologies used (Java, Maven, Hibernate, Spring Boot, Docker, Postgres , REST API, Slack)



GET /image/{id}

Parameters Cancel

Name	Description
id * required	
integer(\$int64)	
(path)	

Execute

Responses

Code	Description	Links
200	OK	No links

Media type: ▼

Controls Accept header:

Example Value | Schema



```
[
  "string"
]
```



Slack



- Implemented by integrating Slack WebHook in the REST API
- REST API triggers notification to slack channel on receiving images
- Notification contains details such as
 - confidence level,
 - Timestamp, and
 - number of rats detected





Notification Bot APP 7:21 PM

 ALERT!!! 2 rats Detected with Confidence Level : 0.65 at 2023-01-29 19:21:08.933354. Check the website for more details. 

 ALERT!!! 1 rat Detected with Confidence Level : 0.66 at 2023-01-29 19:21:14.018732. Check the website for more details. 

 ALERT!!! 1 rat Detected with Confidence Level : 0.52 at 2023-01-29 19:21:25.715930. Check the website for more details. 

 ALERT!!! 1 rat Detected with Confidence Level : 0.71 at 2023-01-29 19:21:38.093022. Check the website for more details. 

Web Application

- Fetch data from REST API
- Display data into the website.
- Technologies used (Python, Flask, Jinja, Docker)

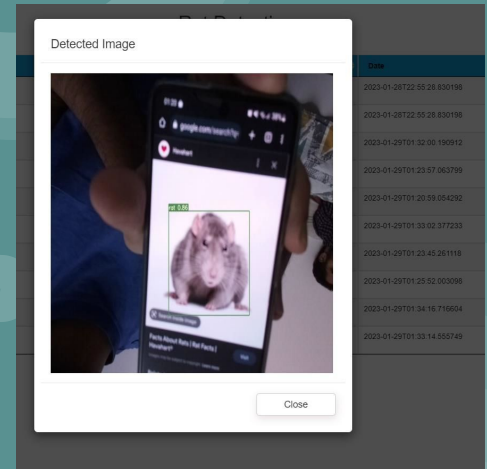
← → ↻ 🏠 ⚠ Not secure | 192.168.0.10:5000/rat_page

Rat Detection

Show entries Search:

S.No	Name	Type	Confidence Level	Date	View
105	Image2023-01-29 01:32:00.150912.jpg	image/jpg	0.86	2023-01-29T01:32:00.190912	view
106	Image2023-01-29 01:32:11.958397.jpg	image/jpg	0.83	2023-01-29T01:32:11.958397	view
107	Image2023-01-29 01:33:02.377233.jpg	image/jpg	0.52	2023-01-29T01:33:02.377233	view
108	Image2023-01-29 01:33:14.555749.jpg	image/jpg	0.91	2023-01-29T01:33:14.555749	view
109	Image2023-01-29 01:34:16.716604.jpg	image/jpg	0.77	2023-01-29T01:34:16.716604	view
110	Image2023-01-29 19:21:08.933354.jpg	image/jpg	0.65	2023-01-29T19:21:08.933354	view
111	Image2023-01-29 19:21:14.018732.jpg	image/jpg	0.66	2023-01-29T19:21:14.018732	view
112	Image2023-01-29 19:21:25.715930.jpg	image/jpg	0.52	2023-01-29T19:21:25.715930	view
113	Image2023-01-29 19:21:38.093022.jpg	image/jpg	0.71	2023-01-29T19:21:38.093022	view
114	Image2023-01-29 19:21:50.260188.jpg	image/jpg	0.76	2023-01-29T19:21:50.260188	view

Showing 1 to 10 of 46 entries Previous Next



ML model deployment on Sensor Node

```
python detect.py --weights best.pt --conf 0.5 --source 0 --no-trace --exist-ok
/home/pi/.local/lib/python3.9/site-packages/torchvision/io/image.py:13: UserWarning: Failed t
o load image Python extension:
  warn(f"Failed to load image Python extension: {e}")
Namespace(weights=['best.pt'], source='0', img_size=640, conf_thres=0.5, iou_thres=0.45, devi
ce='', view_img=False, save_txt=False, save_conf=False, nosave=False, classes=None, agnostic_
nms=False, augment=False, update=False, project='/home/pi/Desktop/Rat_Detection/Detections',
name='exp', exist_ok=True, no_trace=True)
YOLOv0.1-115-g072f76c torch 1.13.1 CPU

Fusing layers...
RepConv.fuse_repvgg_block
RepConv.fuse_repvgg_block
RepConv.fuse_repvgg_block
/home/pi/.local/lib/python3.9/site-packages/torch/functional.py:504: UserWarning: torch.meshg
rid: in an upcoming release, it will be required to pass the indexing argument. (Triggered in
ternally at /root/pytorch/aten/src/ATen/native/TensorShape.cpp:3190.)
  return _VF.meshgrid(tensors, **kwargs) # type: ignore[attr-defined]
Model Summary: 306 layers, 36479926 parameters, 6194944 gradients, 103.2 GFLOPS
1/1: 0... success (640x480 at 30.00 FPS).

0: Done. (5121.2ms) Inference, (1.0ms) NMS
0: Done. (4865.5ms) Inference, (0.8ms) NMS
0: 1 rat, Done. (4997.7ms) Inference, (5.8ms) NMS
```

```
pi@raspberrypi:~/Desktop/yolov7 $ make dispatch
python dispatch.py
http://192.168.0.11:8083/image/upload
```

Postgres Deployment on K3S Cluster

```
PS C:\Users\ashle\k3s\db-postgres> kubectl apply -f postgresconfig.yaml
configmap/postgres-config created
PS C:\Users\ashle\k3s\db-postgres> kubectl apply -f postgrespvcpv.yaml
persistentvolume/postgres-pv-volume created
persistentvolumeclaim/postgres-pv-claim created
PS C:\Users\ashle\k3s\db-postgres> kubectl apply -f postgresdeployment.yaml
deployment.apps/postgres created
PS C:\Users\ashle\k3s\db-postgres>
```

```
PS C:\Users\ashle\k3s\db-postgres> kubectl apply -f postgreservice.yaml
service/postgres created
```

```
PS C:\Users\ashle\k3s\db-postgres> kubectl get all
```

NAME	READY	STATUS	RESTARTS	AGE
pod/postgres-654ddd49b4-tmp4n	1/1	Running	0	32m

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/kubernetes	ClusterIP	10.43.0.1	<none>	443/TCP	98m
service/postgres	LoadBalancer	10.43.161.198	192.168.0.10,192.168.0.11,192.168.0.12,192.168.0.13	5432:31595/TCP	2m21s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/postgres	1/1	1	1	32m

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/postgres-654ddd49b4	1	1	1	32m

REST API Deployment on K3S Cluster

```

pi@ksworker1: ~$ sudo docker pull anishpokhrel/cloud-tag:latest
latest: Pulling from anishpokhrel/cloud-tag
114ba63dd73a: Already exists
bc0b8a8acead: Already exists
a4ea641ee679: Already exists
04e9e95aca68: Already exists
433ac3e3efad: Already exists
e4bbe8c34c85: Already exists
af7e5c7f7eec: Already exists
5980d37bc869: Pull complete
Digest: sha256:4d6b5e6ef0daf809f7b1fff6bc4a01cd71a0fa759dc6bf37c265ae00f3e5cefe7
Status: Downloaded newer image for anishpokhrel/cloud-tag:latest
docker.io/anishpokhrel/cloud-tag:latest
  
```

```
PS C:\Users> kubectl get all
```

NAME	READY	STATUS	RESTARTS	AGE
pod/postgres-654ddd49b4-tmp4n	1/1	Running	0	3h44m
pod/cloud-app-57d856849-2qlg8	1/1	Running	0	31m

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/kubernetes	ClusterIP	10.43.0.1	<none>	443/TCP	4h50m
service/cloud-app	LoadBalancer	10.43.12.165	192.168.0.10,192.168.0.11,192.168.0.12,192.168.0.13	8083:30656/TCP	25m
service/postgres	LoadBalancer	10.43.161.198	192.168.0.11,192.168.0.12,192.168.0.13	5432:31595/TCP	3h14m

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/postgres	1/1	1	1	3h44m
deployment.apps/cloud-app	1/1	1	1	31m

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/postgres-654ddd49b4	1	1	1	3h44m
replicaset.apps/cloud-app-57d856849	1	1	1	31m

Web Application Deployment

```
PS C:\Users\ashle> kubectl get all
```

NAME	READY	STATUS	RESTARTS	AGE
pod/flask-app-587d4c74d7-sqkb9	1/1	Running	2 (9d ago)	9d
pod/postgres-654ddd49b4-tmp4n	1/1	Running	3 (9d ago)	10d
pod/cloud-app-57d856849-xm9js	1/1	Running	14 (4d2h ago)	10d

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/kubernetes	ClusterIP	10.43.0.1	<none>	443/TCP	10d
service/postgres	LoadBalancer	10.43.161.198	192.168.0.10,192.168.0.11,192.168.0.12,192.168.0.13	5432:31595/TCP	10d
service/flask-app	LoadBalancer	10.43.40.195	192.168.0.10,192.168.0.11,192.168.0.12,192.168.0.13	5000:30905/TCP	9d
service/cloud-app	LoadBalancer	10.43.169.236	192.168.0.10,192.168.0.11,192.168.0.12,192.168.0.13	8083:30276/TCP	10d

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/flask-app	1/1	1	1	9d
deployment.apps/postgres	1/1	1	1	10d
deployment.apps/cloud-app	1/1	1	1	10d

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/flask-app-587d4c74d7	1	1	1	9d
replicaset.apps/postgres-654ddd49b4	1	1	1	10d
replicaset.apps/cloud-app-57d856849	1	1	1	10d

```
PS C:\Users\ashle>
```


DEMO

Result

- Successful live detection of rats at the edge/sensor node
- Dispatch the detected rat image to K3S cluster via the REST API
- REST API processes and saves the image in postgres DB
- Slack notification alerts are sent
- Full details about detection can be seen on the web application
- Detection history also available on the web application

Questions!!



Image source:
<https://www.google.com/url?sa=i&url=https%3A%2F%2Fimgflip.com%2F%2F4ivf7u&psig=AOvVaw1aKUTEyly3nZ9g5DYK2Ojx&ust=1675981813100000&source=images&cd=vfe&ved=0CBCEQjhqFwoTCMJPuPb8hv0CFQAAAAAdAAAAABAE>