Applikation

Zur Ausführung den *Application* Ordner mit Folgendem Inhalt auf den Raspberry Pi kopieren:

```
└── Application/
    ├── models/
    │   └── animals_faster_rcnn_inception/
    │       ├── frozen_inference_graph.xml
    │       └── ...
    ├── launcher.sh
    ├── main.sh
    ├── detection.py
    └── connection.py
```

Das Script *main.py* ist die Hauptanwendung, welche *detection.py* und *connectoin.py* verwendet.

Aufgerufen wird das *main.py* Script über das *launcher.sh* Script

Im *models* Ordner sind OpenVino Modelle die auf folgende Datensätze trainiert wurden:

- Animals: 9 Wildtierklassen:
- Samples: Alltagsgegenstände zum testen der Anwendung
    - Klassen sind in *classes.txt* der jeweiligen Modell-Ordner definiert

Diese enthalten jeweils folgende Modelle:

- Faster R-CNN: wird von der Applikation verwendet
- SSD: wird verwendet, wenn beim Laden des Faster R-CNN ein Fehler auftritt

Einstellungen

Die Anwendung kann mit oder ohne Senden der Daten ausgeführt werden.
Um die Daten senden zu können wird ein remote.it Account benötigt, wie im Abschnitt **Connection** beschrieben wird.

Dafür müssen folgende Einstellunge vorgenommen werden:

```
# Gibt mit True/False an ob erkannte Bilder gesendet werden sollen
send_results = True

# User-Name des Gerätes, an das die Bilder gesendet werden sollen
remote_user = ''

# Device-Name des Gerätes in remote.it
remote_divice_name = ''

# E-Mail Adresse des remot.it Accounts
remote_it_email = ''
```

```
# Passort des Gerätes, an das die Bilder gesendet werden sollen
password_remote_divece = ''

# Passort des remot.it Accounts
password_remoteit = ''

# Pfad im Zielgerät, in dem die Bilder gespeichert werden sollen
remote_output_dir = os.path.join('/home', remote_user)

# (optinal) E-Mail Adresse an die eine Benachrichtigung gesendet werden
soll
send_email = None
```

Wenn send_results = False müssen die Einstellungen nicht gemacht werden und die Bilder werden lokal gespeichert.

Autostart

Um die Anwendung beim booten des Raspberry's automatisch zu starten:

Im *launcher.sh* den absoluten Pfad in dem das *main.py* Script liegt definieren.
Anschließend einen Service anlegen, der launcher.sh beim booten ausführt

```
sudo nano /lib/systemd/system/my_init.service
```

und folgendes schreiben:

```
[Unit]
Description=init
After=multi-user.target

[Service]
User=pi
Group=pi
Type=idle
ExecStart=bash /pfad/zu/launcher.sh &

[Install]
WantedBy=multi-user.target
```

Zugriffsrechte geben:

```
sudo chmod 644 /lib/systemd/system/my_init.service
```

Daemon neu laden (nach jeder änderung)

```
sudo systemctl daemon-reload
```

manuelles starten/stoppen des Service

```
sudo systemctl start my_init.service
sudo systemctl stop my_init.service
```

automatisches starten beim Boot aktivieren/deaktivieren

```
sudo systemctl enable  myscript
sudo systemctl disable  myscript
```

Status abfragen (gibt Konsolenausgabe von main script aus)

```
sudo systemctl status my_init.service
```

Connection

Verwendet remote.it zum senden von Daten mit scp über proxy ssh Verbindung.

1. Remote.it installieren

Auf dem Remote Gerät (an dass Daten gesendet werden sollen)

installieren

```
curl -LkO
https://raw.githubusercontent.com/remoteit/installer/master/scripts/auto-
install.sh
chmod +x ./auto-install.sh
sudo ./auto-install.sh
```

und auführen

```
sudo connectd_installer
```

Mit remote.it Konto anmelden, oder neuen account erstellen.
Dann neues Gerät (z.B. *remote-Pc*) mit SSH Service (z.B. *ssh-Pc*) anlegen

2. Verwendung

(wird von der Applikation implementiert)

In *ssh_connection.py*

- in init(): Developer API Key anpassen
- in main():
    - *email* von remote.it
    - *password_remoteit* von remote.it
    - *password_remote_divece* passwor von remote gerät
    - *remote_user* username des remote geräts
    - *remote_divice_name* mit dem gerät über *connectd* angemeldet wurde
    - *file_path* datei die gesendet werden soll
    - *remote_output_dir* directory auf zielgerät

einloggen und verbinden:

```
conn = SSHConnect(email, password_remoteit)
logged_in = conn.login(remote_divice_name)
if logged_in:
    ret = conn.connect()
```

wenn erfolgreich, enthällt ret *server* und *port*, sonst false

senden

```
server, port = ret
conn.send(server, port, remote_user, password_remote_divece,
        file_path, remote_output_dir)
```

verbindung trennen

```
conn.disconnect()
```

script zum testen ausführen

```
python3 connection.py
```

Email Connection

mit

```
conn.send_mail('ziel@addresse.com', 'textmessage')
```

von der in *SSHConnect(email, password)* verwendeten email aus an *zieladresse* die *textmessage* senden.

OpenVino Installationen

Ubuntu

1. OpenVino herunterladen und entpacken

```
cd ~/Downloads
tar -xvzf l_openvino_toolkit_p_<version>.tgz
```

2. OpenVino und dependencies installieren

```
cd l_openvino_toolkit_p_<version>
sudo ./isntall_GUI.sh
cd /opt/intel/openvino/install_dependencies
sudo -E ./install_openvino_dependencies.sh
```

3. Environment Variablen setze

```
source /opt/intel/openvino/bin/setupvars.sh
```

(in .bashrc hinzufügen für permanent)

4. ModelOptimizer installieren

```
cd
/opt/intel/openvino/deployment_tools/model_optimizer/install_prerequisites
sudo ./install_prerequisites.sh
```

5. NCS2 einrichten

```
sudo usermod -a -G users "$(whoami)"
sudo cp /opt/intel/openvino/inference_engine/external/97-myriad-
usbboot.rules /etc/udev/rules.d/
sudo udevadm control --reload-rules
sudo udevadm trigger
sudo ldconfig
sudo reboot
```

Raspberry Pi

1. Raspberry Pi Setup

Rasbian herunterladen und img sd karte mit zb BalenaEtcher erstellen.

Dann ssh datei auf sd karte anlegen:

```
cd media/<user>/boot
touch ssh
```

SD Karte in Raspberry und Verbindung über Ethernet Kabel herstellen.
Dann:

```
ssh pi@raspberrypi.local
```

passwort: *raspberry* eingeben.

2. OpenVino auf Raspberry Pi

OpenVino und OpenCV installieren.

Mobiles Internet

Huawei E3531 SurfStick

1. prüfen ob stick erkannt wird:

```
lsudb
```

ausgabe:

```
Bus 001 Device 004: ID 12d1:14dc Huawei Technologies Co., Ltd. E33372
LTE/UMTS/GSM HiLink Modem/Networkcard
```

2. Browser öffnen und auf 129.168.8.1 gehen

3. Pin eingeben (Pin prüfung deaktivieren)

als standart Verbindung fenstlegen, da sonst verbindungswechsel stattfinden kann und dann connection script evtl nicht mehr läuft.

```

```

```
route -n # aktuelle metrik rausfinden
```

```
ifmetric eth1 100 # niedriger als die anderen für höhere proi
```

für permant:
in

```
sudo nano /etc/dhcpcd.conf
```

hinzufügen:

```
interface eth1
metric 200 # niedrigste
static ip_address=192.168.8.1
```