AI & SUSTAINABILITY
IN VET EDUCATION

AI-VET MOOC - шаблон за поднесување проект

Funded by
the European Union

Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

1. Наслов на проектот:

„Визуелен преведувач на гестовен јазик“

2. Категорија на проектот:

☐ **Социјално влијание и инклузивна вештачка интелигенција**

3. Опис на проектот:

Овој систем, базиран на технологии на машинско учење, овозможува автоматско препознавање и преведување на гестовен јазик на глувонемите лица преку анализа на видео-сигнал од камера. Притоа, не е потребна употреба на специјализирана опрема, како што се ракавици или дополнителни сензори како во десега развиените технологии за преведување на гестовен јазик, што ја прави технологијата практична, достапна и лесна за примена.

4. Клучни зборови:

преведувач, гестовен, визуелен, камера, глувонеми

5. Целна публика:

„И едукатори и студенти“

6. Цели на проектот:

Целта на овој проект е создавање на современо технолошко решение за подобрување на комуникацијата на глувонемите лица со останатите кои не го знаат знаковниот јазик а со тоа и подобрување на нивниот социјален живот. Со овој проект се учи процесот на развој и имплементацијата на визуелен преведувач на гестов јазик, фокусирајќи се на техничките и теоретските аспекти што придонеа во негова реализација. Со развојот на овој проект се учи имплементација на методите на машинско учење.
Целта на овој проект е да се стекнат знаења за работа со Raspberry Pi, програмирање во Python и вметнување на библиотеки.
Со оваа вежба се учи:

1. Да се инсталира оперативен систем за Raspberry Pi
2. Хардверско поврзување на елементите
3. Разбирање на интеграцијата на хардвер и софтвер (Python, библиотеки).
4. Да се демонстрира интеракција хардвер-софтвер со користење на Python.

7. Временска рамка на проектот:

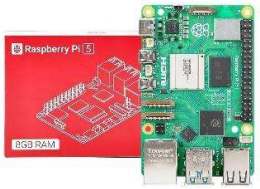
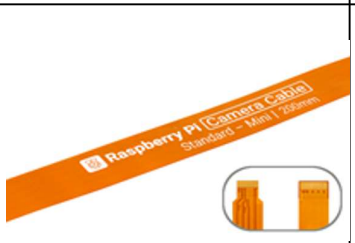
- Датум на почеток: 23.3.2026
- Датум на завршување: 1.4.2026

Funded by the European Union

Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

8. Користени алатки и технологии:

1. При изработка на овој модел за визуално преведување на знаковен јазук се употребани следниве компоненти од комплетот AI-VET:

	Компонента	picture	Функција
1.	Raspberry Pi 5 – 8GB		Се користи како централна управувачка единица („мозок“) на системот. По добивањето на резултатите од камерата, ги обработува и ги пренесува до дисплејот за прикажување.
2.	Raspberry camera module 2		Претставува камера-модул на системот. Го снима гестот прикажан пред неа и, преку вградениот код, го препознава и испраќа резултатот за понатамошна обработка.
3.	FPC кабел за камера 200mm		Се користи за поврзување на камера-модулот со главната плочка, овозможувајќи пренос на податоци и сигнали помеѓу нив.
4.	micro HDMI to HDMI кабел 1m		Се користи за поврзување на системот со дисплеј, пренесувајќи видео и аудио сигнали.
5.	27W USB Напојување од тип C за Raspberry Pi 5		Кабел за напојување

Funded by the European Union

Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

6.	Micro SDHC картица 64GB		Се користат за сериско поврзување и напојување на сите наведени компоненти, овозможувајќи нивна меѓусебна комуникација и синхронизирана работа.

2. AI-VET-Прирачник за склопување со упатства чекор-по-чекор

3. Python код и библиотеки

4. Собирање на фото материјали како база за машинско учење - беше потребно да се обезбеди доволен број слики за секоја класа. Собраните слики се внесуваат во платформата и рачно се обележуваат, со што прецизно му се укажува на моделот што претставува секој поединечен гест.

9. План за имплементација чекор-по-чекор:

Првата фаза од процесот на развој на функционален прототип на визуелниот преведувач на гестовен јазик опфати истражување, при што беа анализирани постојни технолошки решенија и методи од областа на машинското учење и компјутерската визија, со цел поставување јасни цели и избор на соодветни алатки и пристапи.
Втора фаза следуваше детално планирање на шемите и архитектурата на системот, со прецизно дефинирање на неговите компоненти и нивната меѓусебна поврзаност, по што започна развојот на кодот и имплементацијата на алгоритми за обработка на слика и препознавање на гестови.
Трета фаза беше прибирање слики од гестови за тренирање на моделот за машинско учење, при што беше искористена платформата Edge Impulse со функција за препознавање на објекти (object recognition), преку која системот се обучуваше да ги класифицира гестовите.
Четврта-завршната фаза беше склопување на сите компоненти во единствен систем, по што следуваше тестирање и практична проверка на функционалноста, со цел обезбедување точност и стабилност во неговото работење.

1. **План-идеа што сакаме да изработиме:** Визуалниот преведувач на гестов јазик
2. **Истражување за технолошко решение и избор на соодветни компоненти.**

Funded by the European Union

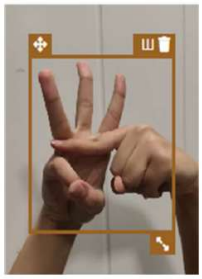
Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

3. Подготовка на хардверските компоненти: собирање на сите потребни материјали - Raspberry Pi 5, кабли за напојување, камера и потоа следи поврзување.

4. Склопување на Raspberry Pi 5 и кутија за Pi 5 со вентилатор со контролирана температура.

5. Инсталирање на Raspbian OS на SD картичката и конфигурирање пристап до мрежата преку VNC или SSH и ажурирајте ги системските пакети.

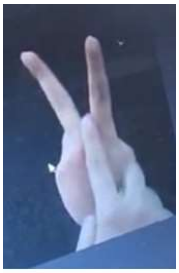
6. Собирање на фото материјали како база за машинско учење.

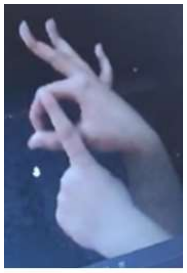


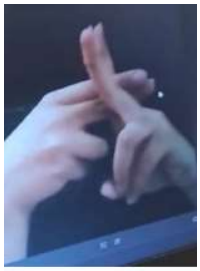
Слика 1: Буква Ш



Слика 2: Буква В


Слика 3: Буква Ц


Слика 4: Буква Ф


Слика 5: Буква Ч

По прибирањето и означувањето на податоците, следува најзначајната фаза – обучување на моделот за машинско учење. Во текот на обуката се формира координатен (просторен) систем во кој секоја класа, односно секој гест, е претставена со посебна боја. Колку што примероците од истата класа се погрупирани и појасно одвоени од останатите, толку е поголема способноста на моделот правилно да ги препознава гестовите и да избегнува нивно меѓусебно мешање.



По успешно завршената обука, системот овозможува тестирање на моделот и прикажување на процентот на точност. Овој показател ја претставува успешноста на моделот, односно колку пати од вкупниот број тестирања (на пример 150 или друг зададен број) машината дала точен резултат. На тој начин се добива јасна слика за ефикасноста и сигурноста на развиениот систем.

7. **Инсталирање на наредбите потребни за имплемингтација**

sudo apt update

curl -sL https://deb.nodesource.com/setup_20.x | sudo bash -

sudo apt install -y gcc g++ make build-essential nodejs sox gstreamer1.0-tools gstreamer1.0-plugins-good gstreamer1.0-plugins-base gstreamer1.0-plugins-base-apps

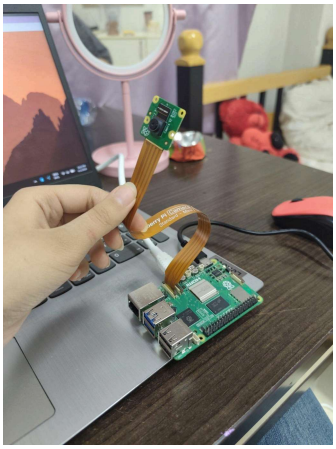
Funded by the European Union

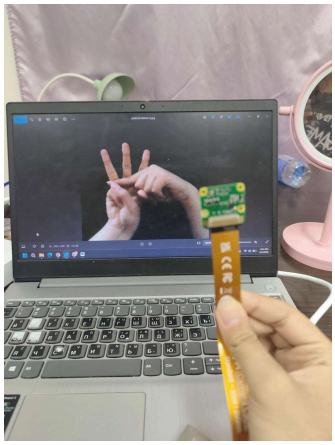
Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

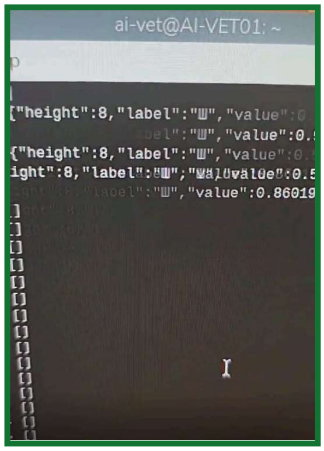
sudo npm install edge-impulse-linux -g --unsafe-perm
sudo npm install -g npm@10.8.1
sudo apt update
sudo apt install -y docker.io
sudo systemctl enable docker
sudo systemctl start docker
sudo docker run -it --rm --privileged --network=host -v /dev/:/dev/ --env UDEV=1 --device /dev:/dev --entrypoint /bin/bash ubuntu:20.04
apt-get update
apt-get install wget -y
wget https://deb.nodesource.com/setup_20.x
bash setup_20.x
apt install -y gcc g++ make build-essential nodejs sox gstreamer1.0-tools gstreamer1.0-plugins-good gstreamer1.0-plugins-base gstreamer1.0-plugins-base-apps vim v4l-utils usbutils udev
apt-get install npm –y
edge-impulse-linux
edge-impulse-linux-runner

8. Програмирање на кодот

9. Тестирање







10- Код

```
#!/usr/bin/env python

import device_patches       # Device specific patches for Jetson Nano (needs to be before importing cv2)

try:
    import cv2
except ImportError:
    print('Missing OpenCV, install via `pip3 install "opencv-python>=4.5.1.48,<5"`')
    exit(1)
```

Funded by the European Union

Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

```
import os
import sys, getopt
import signal
import time
from edge_impulse_linux.image import ImageImpulseRunner

runner = None
# if you don't want to see a camera preview, set this to False
show_camera = True
if (sys.platform == 'linux' and not os.environ.get('DISPLAY')):
    show_camera = False

def now():
    return round(time.time() * 1000)

def get_webcams():
    port_ids = []
    for port in range(5):
        print("Looking for a camera in port %s:" %port)
        camera = cv2.VideoCapture(port)
        if camera.isOpened():
            ret = camera.read()[0]
            if ret:
                backendName =camera.getBackendName()
                w = camera.get(3)
                h = camera.get(4)
                    print("Camera %s (%s x %s) found in port %s " %(backendName,h,w,
port))
                port_ids.append(port)
            camera.release()
    return port_ids

def sigint_handler(sig, frame):
    print('Interrupted')
    if (runner):
        runner.stop()
    sys.exit(0)

signal.signal(signal.SIGINT, sigint_handler)

def help():
     print('python classify.py <path_to_model.eim> <Camera port ID, only required
when more than 1 camera is present>')

def main(argv):
    try:
        opts, args = getopt.getopt(argv, "h", ["--help"])
    except getopt.GetoptError:
```

Funded by the European Union

Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

```
        help()
        sys.exit(2)

    for opt, arg in opts:
        if opt in ('-h', '--help'):
            help()
            sys.exit()

    if len(args) == 0:
        help()
        sys.exit(2)

    model = args[0]

    dir_path = os.path.dirname(os.path.realpath(__file__))
    modelfile = os.path.join(dir_path, model)

    print('MODEL: ' + modelfile)

    with ImageImpulseRunner(modelfile) as runner:
        try:
            model_info = runner.init()
            # model_info = runner.init(debug=True, timeout=60) # to get debug print out
and set longer timeout
                print('Loaded runner for "' + model_info['project']['owner'] + ' / ' +
model_info['project']['name'] + '"')
            labels = model_info['model_parameters']['labels']
            if len(args)>= 2:
                videoCaptureDeviceId = int(args[1])
            else:
                port_ids = get_webcams()
                if len(port_ids) == 0:
                    raise Exception('Cannot find any webcams')
                if len(args)<= 1 and len(port_ids)> 1:
                    raise Exception("Multiple cameras found. Add the camera port ID as a
second argument to use to this script")
                videoCaptureDeviceId = int(port_ids[0])

            camera = cv2.VideoCapture(videoCaptureDeviceId)
            ret = camera.read()[0]
            if ret:
                backendName = camera.getBackendName()
                w = camera.get(3)
                h = camera.get(4)
                    print("Camera %s (%s x %s) in port %s selected." %(backendName,h,w,
videoCaptureDeviceId))
                camera.release()
            else:
```

Funded by the European Union

Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

```
            raise Exception("Couldn't initialize selected camera.")

        next_frame = 0 # limit to ~10 fps here

        for res, img in runner.classifier(videoCaptureDeviceId):
            if (next_frame > now()):
                time.sleep((next_frame - now()) / 1000)

            # print('classification runner response', res)

            if "classification" in res["result"].keys():
                print('Result (%d ms.) ' % (res['timing']['dsp'] + res['timing']['classification']), end='')
                for label in labels:
                    score = res['result']['classification'][label]
                    print('%s: %.2f\t' % (label, score), end='')
                print('', flush=True)
            elif "bounding_boxes" in res["result"].keys():
                print('Found %d bounding boxes (%d ms.)' % (len(res["result"]["bounding_boxes"]), res['timing']['dsp'] + res['timing']['classification']))
                for bb in res["result"]["bounding_boxes"]:
                    print('\t%s (%.2f): x=%d y=%d w=%d h=%d' % (bb['label'], bb['value'], bb['x'], bb['y'], bb['width'], bb['height']))
                    img = cv2.rectangle(img, (bb['x'], bb['y']), (bb['x'] + bb['width'], bb['y'] + bb['height']), (255, 0, 0), 1)
            elif "freeform" in res['result'].keys():
                print('Result (%d ms.)' % (res['timing']['dsp'] + res['timing']['classification']))
                for i in range(0, len(res['result']['freeform'])):
                    print(f'    Freeform output {i}:', ', '.join(f'{x:.4f}' for x in res['result']['freeform'][i]))

            if (show_camera):
                cv2.imshow('edgeimpulse', cv2.cvtColor(img, cv2.COLOR_RGB2BGR))
                if cv2.waitKey(1) == ord('q'):
                    break

            next_frame = now() + 100
        finally:
            if (runner):
                runner.stop()

if __name__ == "__main__":
    main(sys.argv[1:])
```

Funded by the European Union

Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.

11. Критериуми за евалуација:

- **Функционалност на системот – 70%**
  Се оценува дали проектот работи правилно и дали ги исполнува сите зададени барања и цели.
- **Естетика – 10%**
  Се оценува визуелниот изглед и општиот дизајн на крајниот продукт.
- **Идејно решение за поставеност на елементите – 10%**
  Се мери едноставноста, јасноста и практичноста на аранжманот на компонентите за лесна употреба.
- **Временска рамка – 10%**
  Се разгледува дали проектот е завршен во предвидениот рок и временска рамка.

12. Кредити

Василиса Курзакова - Работеше самостојно од идеа, истражување, склопување, инсталирање, хардверско решение, поврзување и софтверско програмирање.

Професор Гордана Наумовска - Обезбеди насоки и поддршка во текот на работењето на проектот.

13. Линк до видео презентација (макс. 5 минути)

https://drive.google.com/file/d/1SQzJSp773lVl_t9fMmEftGkGcFyOxRt7/view?usp=drive_link

14- Изјави

Изјавувам дека ги прочитав и се согласувам со следните изјави:

1- Проектот е оригинален и развиен од наведените членови на тимот.
2- Сите технологии и извори се правилно цитирани.
3- Проектот е развиен со користење на компоненти од комплетот AI-VET.
4- Ние го овластуваме ширењето на проектот за образовни и промотивни цели (неексклузивни права).

Funded by the European Union

Овој проект е финансиран со поддршка од Европската комисија. Оваа комуникација ги одразува само ставовите на авторот, а Комисијата не може да се смета за одговорна за каква било употреба што може да се направи на информациите содржани во неа.