A Case Study for Scoliosis: How MLOps Can Help Reduce AI Challenges in Health Care?

Gábor György Gulyás
Vitarex Stúdió Ltd
Budapest, Hungary
gabor@gulyas.info

Janis Lapins
Data Science
Spicetech Gmbh
Stuttgart, Germany
janis.lapins@spicetech.de

Attila Csaba Kiss
Vitarex Stúdió Ltd
Budapest, Hungary
kiss.csaba@vitarex.hu

***Abstract*—The integration of Artificial Intelligence (AI) into healthcare diagnostics represents a significant advancement, particularly in the screening for conditions, such as scoliosis. This paper discusses the development, implementation, and evaluation of the Posture Buddy (PB) device, a machine vision driven tool designed to enhance the efficiency of scoliosis screening among school-aged children within the Hungarian health visitor system. Through the lens of Machine Learning Operations (MLOps) practices, our case study demonstrates the pivotal role of MLOps in overcoming operational hurdles at the intersection of eHealth and AI. The field-testing of PB revealed that within the context of low light conditions and slight side viewing angles the device performance decreases. In a later phase of the project, the pose estimation model of the device was put through model validation, observing the same flaw. Through these findings, the importance of proactive validation of AI models in healthcare is highlighted, whereas it also underscores the need to use MLOps to enable continuous deployment through the lifecycle of ML-based medical tools.**

***Index Terms*—machine learning, edgeML, health care, MLOps, pose estimation.**

I. Introduction

The integration of Artificial Intelligence (AI) in healthcare has opened new frontiers in diagnostics, treatment planning, and patient care, offering the potential to significantly enhance the accuracy and efficiency of medical services. However, the adoption of AI technologies in healthcare is fraught with challenges such as the reliability, validation, and operationalization of AI models. Scoliosis, a condition characterized by an abnormal lateral curvature of the spine, affects millions worldwide, and traditional screening methods, while effective, are labor-intensive. This underscores the need for automated AI-driven solutions that could improve the speed of screening and also allow better documentation of results. Eventually, such solutions could become more accurate than traditional screening.

This paper presents a case study focused on the development of a Machine Learning (ML) based device for scoliosis screening, highlighting the pivotal role of Machine Learning Operations (MLOps) practices in overcoming the prevalent challenges. The study is anchored in the development and field-testing of Posture Buddy (PB), a device aimed at enhancing the screening process for spinal abnormalities among school-aged children. While machine learning, and machine vision in particular, have a whole host of potential uses in health care, it is crucial that malfunctions are handled, or avoided in advance.

During the field trial of PB, it emerged that its performance decreased under poor lighting conditions and when looking at the patient from aside. Had this been identified beforehand, the users could have been notified in advance, or even deployed an updated model version to mitigate this issue. However, it was not identified earlier, but at an advanced phase, a virtual validation tool called VALICY enabled us to identify these same shortcomings in the model (this validation was an international collaboration in the IML4E [Industrial Grade Machine Learning] project [6], for which PB served as a testing ground). This experience underscores the importance of thorough validation before the first deployment, and the utilization of MLOps infrastructure, to preemptively correct such errors before they impact users.

The paper is structured as follows. In Section II, the Hungarian health visitor system is described to the point of outlining their work in scoliosis screening. Following that, in Section III, PB is presented, a device that was developed to help and enhance the screening process. Its details considering hardware, software and machine learning, and how it was evaluated in our field study are described. In Section IV, the MLOps pipeline is presented, along with model evaluation methods. Section V provides validation details with a black-box validation tool, and the current work is concluded in Section VI.

II. Health Visitors and Scoliosis Screening

Health visitors play a crucial role in public health in Hungary. They provide essential services and support to individuals and families, particularly in the realm of preventive healthcare and early intervention, with a focus on the health of children.

A. Their History and Current Service

The health visitor service was established in 1915 under the name Stefánia Association, with its main goal being the protection of mothers and infants [21], [22]. The Green Cross Health Service started in 1927, whose health visitor service operated from 1930 to 1944, with its scope of activities extending to school-age groups. From then on, health visitors were

Copyright (c) IARIA, 2024. ISBN: 978-1-68558-167-1

systematically involved in school health care in educational institutions. After World War II, the two health visitor services merged, and their current work is based on The Law on Public Education (1993). This provides ground for students to have the right to receive regular health supervision and care.

There are two types of health visitors: school and district health visitors. School health visitors primarily focus on providing healthcare services within educational institutions. On the other hand, district health visitors operate within communities, offering healthcare services directly to families, particularly focusing on maternal and child health, preventive care, and health education. Both play important roles in public health, with school health visitors emphasizing school-based interventions and district health visitors focusing on community-based health promotion and support.

B. Posture Screening in Schools

Health visitor screening examinations are conducted for children aged 3-18 years, where they collaborate in the provision of school health tasks [22]. By law, school health care contains tasks to be performed independently by the health visitor, including screening examinations for specific age groups, such as assessing height, weight, physical development, identifying psychological, motor, mental, and social development and behavioral problems, among others. In addition, it is important to document the tasks performed.

All these circumstances enable the testing of planned tools with the involvement of health visitors. Our project targeted musculoskeletal screening examinations, with a particular focus on spinal disorders. Considering that screening examinations are carried out by multiple professionals in various age groups every two years, the introduction of a digital measurement application could enhance documentation and comparison of results.

C. Scoliosis Screening Protocol Followed by Health Visitors

The procedure for musculoskeletal examination and the positions and movements to be considered during the examination setup are outlined based on the guidelines provided in the school health manual edited by Dr. Anna Aszmann [1]:

1) The student stands with their back to the examiner. They wear only underwear on their upper body. The examiner observes the student's posture, focusing primarily on the trunk-arm triangle, shoulder deviations, and spinal curvatures.
2) The examiner asks the student to raise both arms towards the ceiling. They observe whether the student can compensate for any observed abnormalities with the back muscles.
3) The student leans forward with extended arms towards the floor. Here, the health visitor assesses the strength of the back muscles and the ability to compensate for abnormalities.
4) The examiner instructs the student to straighten up and lower their arms to their sides loosely, then to turn to one side (with the legs as well). With closed eyes, the student raises both arms horizontally and holds them for about 30 seconds. Any changes in posture during this test, such as leaning back or tilting the pelvis forward, may indicate postural problems or weakness.
5) The student lowers both arms to their sides and faces the examiner, allowing any potential pelvic abnormalities and chest deformities to be observed.

During this protocol, health visitors are looking for postural disorders and signs of scoliosis.

III. The *Posture Buddy* Device

The development of this device was done within the IML4E project as a use case of the project [6] (2021-2024). The purpose behind developing and validating the PB tool is to enhance preventive screenings for musculoskeletal abnormalities (such as spinal and other postural disorders) among students, moving beyond the current method of assessment based on visual inspection, lacking visual documentation and an objective basis for comparison.

A. Hardware and Appearance

Initially, a decision on the platform was needed. Developing a mobile application seemed like a natural solution, since the majority of potential users already own one. However, the variety of OS (Operating System) versions, lacking sufficient support to properly run the deep learning based apps, posed a challenge. Additionally, at the project's inception, except for the top-tier mobile phones (which were not widespread in Hungary), our algorithms ran slowly on mobile devices, typically less than one frame per second.

Since health visitors in rural areas have unstable network connection, the posture analysis program needed to function without a network connection. Moreover, for privacy reasons, it was desirable for the evaluation to be performed directly on the device (edge ML); making it impossible to move the evaluation behind a secured API (Application Programming Interface) in the cloud. These constraints necessitated storing and running the machine learning model on the device itself. However, running larger models or updating models on mobile devices is rather difficult, thus prompting exploration of alternative options.

Alternatively, the application could be developed for single-board computers, such as the Raspberry Pi [8]. Such devices are small-sized, low-power, relatively inexpensive, and possess much of the functionality of a traditional computer. Therefore, they offer much more flexibility, but also bring in additional challenges (e.g., providing a proper housing, screen and peripherials). Besides the widely known Raspberry Pi, the Nvidia Jetson Nano system [7] seemed the most suitable for the task, as it features GPU-like (Graphics Processing Unit) hardware acceleration for machine learning (with 128 CUDA [Compute Unified Device Architecture] cores). But, as Nvidia Jetsons were globally unavailable in 2021, we opted for the Raspberry Pi 4.

This all resulted in a uniquely designed, compact-sized computer with its own custom 3D-printed housing developed

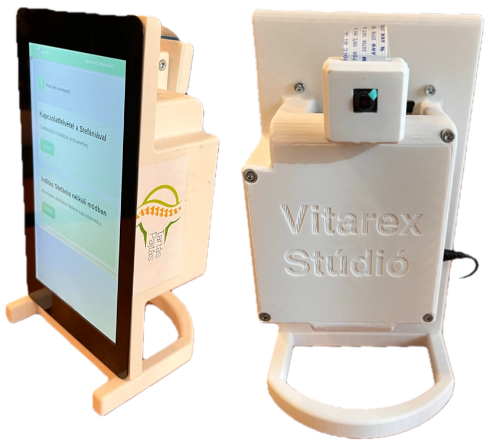


Fig. 1: Two photos of the Posture Buddy device.

by Vitarex Stúdió Ltd, depicted in Figure 1. Its main board is a Raspberry Pi 4 (4GB or 8GB), with an appropriate camera, a 7-inch touchscreen, and a USB-C type charger. The housing is designed to leave the Raspberry Pi ports accessible, allowing peripherals, such as external screens, keyboards, and mice to be connected, enabling the saving of results to a USB drive if needed. When turned on, the posture analysis program starts automatically.

B. Software

The posture analysis is a full screen application that loads automatically after turning the device on. It follows a five-step examination procedure based on the spinal examination protocol described in Section II-C. During the examination, students need to position themselves in front of the camera according to specific instructions displayed on the screen, then the health visitor captures the image. The software automatically detects and displays key points of the human figure in the images (when possible), allowing manual correction if needed. Throughout the five steps, the system continuously calculates and records data indicating the extent of spinal curvature deviation. Upon completion of the examination, it displays all captured images and analytical results.

The software architecture is designed to leverage the full potential of the open computing platform: PB is a lightweight web application (running fully locally). This enables an integrated client-server architecture within a single device, it is easy to change and to update, even remotely. The client side uses web technologies, such as HTML (HyperText Markup Language), CSS (Cascading Style Sheets), and JavaScript, and the server side is a Flask Python app [10], running machine vision based on OpenCV [11] and other libraries.

The server-side has a modular structure, separating different functional units to promote maintainability and clarity of the codebase. For example, the data submission module (for MLOps) is responsible for anonymizing images generated during examination. The metrics module performs calculations related to individual examination steps. Another module handles the integration with the Stefánia Registration System [12] for health visitors (which is used by the majority of health visitors in Hungary), enabling the recording of examination images within the system. Another module is responsible for exporting examination summaries to a USB drive.

In order to deliver fixes to errors, a software update mechanism was implemented, enabling the distribution of new software versions. The new software version is uploaded as a Github Release Asset and the software periodically checks if a new version is available. New versions are downloaded with dependencies attached, and are installed. Finally, the device is restarted. This mechanism ensures the PB device remains functional and up-to-date, enhancing user experience and device reliability.

C. Machine Vision Algorithms

The software solution utilizes multiple machine learning models. The first model determines the coordinates of key points of the human figure in the images (if possible), then additional models serve to separate students from the background and separate the shape and contour of the student's figure (if needed).

1) Pose Estimation: Before entering into the development, models are compared based on their accuracy (by considering their MSE; Mean Square Error) and their processing speed in terms of FPS (Frame Per Sec). At the time, the most widespread model was PoseNet [2]. The output stride (with possible values of 8, 16, 32), regulates the model's processing accuracy, where higher values result in faster but less precise processing. The model also has a multiplier parameter (with possible values of 1.01, 1.0, 0.75, and 0.50) that controls the depth of convolutional processing, where higher values offer more precision but slower processing. Two settings were examined: a rapid but less accurate setup with output stride = 16 and multiplier = 0.5, and a slower yet more precise configuration with output stride = 32 and multiplier = 1.01. OpenPose [3] offered multiple models, the one with 25 keypoints were selected, as that proved to be the most efficient. Due to video memory constraints, it was tested with resolutions of 128x128 and 160x160. For the TRT_Pose system [4], the ResNet-18 model (18 layers deep) was used as it yielded better results based on our preliminary measurements.

In the measurements, it was observed that the TRT_Pose model delivered the best performance by simultaneously achieving the highest image processing speed while producing the fewest errors (5.6 FPS, 13 MSE). The OpenPose models also performed well with minimal errors, albeit at significantly lower image processing speeds (3,6 FPS with MSE around 17). PoseNet lagged behind in both evaluation criteria compared to these results (2.1 FPS and MSE 78; 3.7 FPS and MSE 108).

In each step of the pose estimation process, after taking the picture, PB allows health visitors to improve the results. That is, a screen is loaded where all detected keypoints are displayed over the photo. The health visitor is then allowed to move these points for correction. In the case of a correction, the provided information could be used for improving the model performance.

2) Machine Vision Algorithms: As described in Section II-C, five different metrics have been used for each step in the protocol. All photos are taken in a standardized setting, allowing asymmetries to emerge.

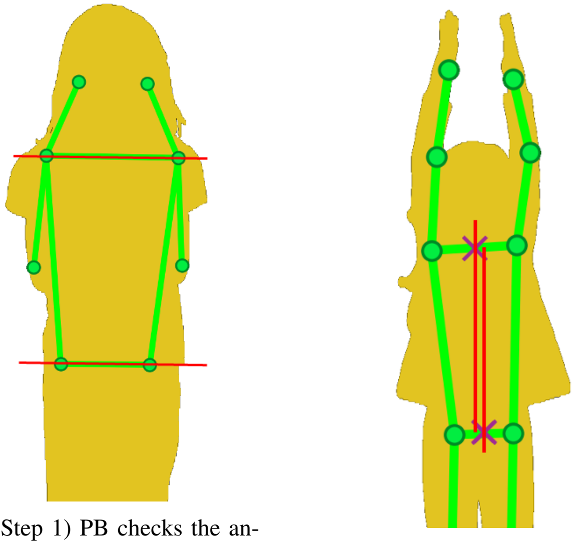
Standing backwards. First, the keypoints are calculated with TRT_Pose, then check the angle defined by the keypoints for the hips and shoulders (horizontal asymmetry). Greater angles reported by this metric can mean a more serious case of scoliosis. (cf. Figure 2a)

Standing backwards, hands raised. In this step, the keypoints are also calculated, and then measure vertical asymmetries between shoulder and hip points. Deviations are measured by the visible tilt of these bodypoints. (cf. Figure 2b)

Leaning forwards. In this step, the student bends forward with their arms ahead, leaving insufficient visibility for keypoint-based measurements. Instead, the silhouette of the back is determined, and its inclination angle is calculated by polygon fitting. The larger the angle, the higher the chance of more advanced scoliosis.

Turned to side, hands raised in front. When they are turned to their side, keypoints can be used again. In this case, keypoints are used combined with polynomial silhouette mapping to determine how bent the spine is.

Standing in front of the camera. This step uses keypoints to determine the vertical asymmetry of keypoint pairs on each side: it gives an asymmetry score about how the distances of keypoints are proportional to each other.



(a) (Step 1) PB checks the angle of the visualized two lines. In this case, the lines are almost in parallel.

(b) (Step 2) PB checks the tilt between hips and shoulders, as displayed by the lines.

Fig. 2: Visualization examples of pose evaluation, using silhouettes only for preserving privacy.

D. Evaluation of the Posture Buddy in a Field Study

Prerequisites for participating in the program included informing their employers, to provide them detailed information about the program, covering its objectives, content, and the roles of health visitors, parents, and students. Participating students were provided detailed information and consent forms (to their parents), who were also verbally informed about the details of the examination, its purpose, data protection measures, and the anonymous handling of data. The trial itself took place in official premises of the institution, involving students who volunteered and had consent forms, outside of regular class hours.

Overall, it can be said that the health visitors conducting the tests provided a realistic picture and assessment of the technical and professional usability of the pose estimation device. Their thorough and detailed technical evaluation was supported by quantified data. Their observations included that it was difficult to set up the internet on the device (for Eduroam), and it would be beneficial if the height of the device could be adjusted like a camera tripod.

They agreed, that after having some features corrected and adding further minor refinements, PB should be a useful tool for school health visitor work. Due to its digital nature, health visitors recommend that besides uploading the resulting examination data set to health visitor programs (such as Stefánia), it should also be uploaded to the EESZT (Electronic Health Service Space) for further use by general practitioners, pediatricians, school physicians, and orthopedic specialists. They found it suitable to attach the findings of the basic screenings, supported by measured data, to the health visitor reports, and to substantiate and justify the referrals for further medical and specialist examinations.

Regarding the operation of the device, one of the health visitors made a significant observation: under poor lighting conditions, the device provides a somewhat inaccurate predictions when viewed from an angle. This health visitor worked in two schools, and in one, their room was long, with poor lighting, and the power cable was too short to provide a good view over the examined student (just from the side). This led to a notably higher error rate of the pose estimation model.

Considering all their comments, a second round of testing is currently run at the time of the writing of this paper. This second round of testing is country wide, involving more schools and health visitors. A mobile application version of the PB device is also developed.

IV. Machine Learning Operations

The former and similar situations where malfunctions have occurred could have been detected in advance with the appropriate validation tools. Even if the devices were already deployed, health visitors' attention could have been drawn to this issue. This is where MLOps, an emerging field can help. MLOps tools cover a wide range, including tools related to failure detection of models, transfer learning and distribution of new model versions, among many others.

A. Our MLOps Infrastructure and Updates

One of the main objectives of the project was to create an MLOps infrastructure that is integrated with the PB.

1. Pipeline. Machine Learning Operations is a comprehensive approach to systematically and efficiently manage the lifecycle of machine learning models [14]. This lifecycle includes the continuous training, evaluation, deployment and monitoring of models. In the context of our project this meant that an MLOps system was created to continuously train, monitor and improve the keypoint detection model built in the PB software.

The infrastructure was mainly developed in Python, which is suitable for data science and machine learning tasks. The chosen deep learning framework was Pytorch [15] because of its popularity and ease of use. The chosen base keypoint detection model was TRT_Pose [4] as previously explained. A pivotal component of the infrastructure is the model training pipeline. It serves the purpose of training keypoint detection models with different hyperparameters and settings.

These settings include:

- the datasets the model is trained and evaluated on
- the model architecture
- the starting weights the model uses
- the ratio between the training and testing sets
- the shapes of the model input and output
- the transformations that are performed on the input images
- type of the optimizer
- the learning rate
- the number of epochs the training lasts

The whole pipeline can be controlled by a configuration file, which contains the values of the pipeline parameters.

The pipeline is structured into five sequential phases. Ingest is responsible for loading and preprocessing datasets, consisting of images and annotations, into a usable format. The next phase is split, where the dataset is sub-divided into distinct training and testing subsets. This is followed by transform, where random transformations to the training data were applied, which may include adjustments in rotation, scale, translation, and color modifications. In the training phase, the initialized model was trained using the specified parameters on the training dataset. Finally, in the evaluation phase a customized evaluation to measure the model's accuracy in keypoint detection was performed. The specifics of this evaluation are explained in a later paragraph. The pipeline is executable both as a standalone Python script and within an interactive Jupyter Notebook environment, offering flexibility in parameter adjustments and modular execution of steps.

2. Evaluation. To accurately assess the efficacy of the keypoint detection models trained via the outlined pipeline, the cocoapi [5] and coco-analyze [16] libraries were used. In the evaluation step the trained model is used to make predictions on the validation images and these predictions are compared to and analyzed with the ground truth values. Through this analysis the Average Precision (AP) and Average Recall (AR) concerning keypoints was calculated.

The coco-analyze library defines a number of error types and calculates the potential improvement in the average precision and average recall metrics if these errors were corrected. Some of these build on the term Object Keypoint Similarity (OKS), which is a metric used to evaluate the accuracy of detected keypoints. It calculates the similarity between the predicted keypoints and the ground truth, considering the distance between them, the standard deviation of the keypoints and the scale of the object. The defined errors are the following:

- **Miss:** the miss score means that the detected keypoint is not close to any body part.
- **Swap:** in this case, the detection close to a body part of a different person.
- **Inversion:** the keypoint is matched to another body part of the same person (e.g., mismatching keypoints of legs or hands).
- **Jitter:** correct keypoint identification, location slightly differs.
- **Score:** close to a ground truth annotation, when two detections are identified, it is the detection with the highest level of confidence that ends up having the lower OKS score.
- **Background false positive:** detections without a ground truth annotation match.
- **False negative:** missed detections.

These metrics are graphically represented, and a detailed report is generated. This facilitates a straightforward comparison of model performances across various validation datasets highlighting the model's strengths and areas for improvement.

3. Mlflow. Mlflow [9] is a popular library that can be used to streamline machine learning development, including tracking experiments (both parameters and results), packaging code into reproducible runs, and sharing and deploying models. The developed model training pipeline is integrated with Mlflow library to utilize its components. These allow the straightforward comparison of various pipeline executions in the Mlflow user interface. The Model Registry component of Mlflow provides a repository for storing the models that have been trained in the pipeline, presenting them in an easily deployable format.

4. Deployment. In the Mlflow user interface all of the available models can be easily compared based on their parameters and metrics. If the model with the best results is chosen, it can be released with running a script that uploads the model to an easily accessible cloud storage service, which in our case is a Github Release page. The trained models are stored and deployed in TorchScript format, in which the models can be packaged without the need to define their original architecture in the production environment. Also, these can be used in non-Python environments and can be optimized for use on edge devices. PB checks if a new model version is available on every start up, and runs an update script if there is any.

5. Monitoring. PB users, at the end of the examination process, can choose to send the original and the corrected keypoint detections to the server of Vitarex along with the anonymized images (faces removed). This way the collected data can be used to evaluate the performance of the model in production as well as to create new datasets to improve the model.

V. Model Validation with VALICY

A. How VALICY Works

VALICY is an AI-based black box testing environment, which allows the virtual validation of multi-dimensional AI based classification systems and complex software, developed by Spicetech GmbH since 2017 [17]. It creates test proposals for the black box application under test which in turn are evaluated and feed back to VALICY.

Through evolution and a competing AI swarm, awareness of the problem's nature increases as more evaluation points provide additional training data. This process helps identify safe or unsafe operational areas. Testing ceases once a predetermined number of runs conclude and either a residual uncertainty is quantified or the desired certainty level is achieved, signaling the end of test proposals.

VALICY's components are:

- a Python framework within a Docker container,
- a server with a set of CPUs,
- a frontend in the browser to display all job results along with the possibility to do cluster analysis of the results and get corresponding characteristics,
- a REST-API to exchange data securely [20],
- a Grafana dashboard to monitor operation during runs, and
- a database.

The Python framework operates across three layers: a Docker layer hosting multiple job instances on a single server, a job instance layer managing workload and enabling the creation of new AI instances (i.e., pre-configured models), and an AI instance worker layer. Depending on the nature of the response of the application under test (fast vs. slow), a different number of AI models run in parallel to generate proposals. The default value of competing AI models is three.

In this framework, grid points define the range of input parameters, serving as the initial dataset for AI to generate decision-making proposals. These proposals seek to pinpoint the boundary where outcomes shift from True to False, with their precision improving through iterative evaluations and feedback. AI models log each run's outcomes and settings, leveraging this history to enhance future predictions and decision-making processes.

Testing a ”black box” application requires only the input parameters—name, range, type (continuous or discrete)—and target parameters—name, threshold, direction (upper, lower), and desired certainty. The process begins by sampling these boundaries and submitting them to the black box via API, with responses stored in the VALICY database and used to inform AI models.

To avoid overlooking key areas, the system intersperses AI-driven evaluations with randomly generated points. Following initial sampling, AI analyses feedback from test points to refine its models, subsequently proposing and assessing potential evaluations based on their likelihood of success. This iterative refinement, informed by direct comparison between predicted outcomes and actual black box feedback, progressively enhances the AI model abilities to accurately predict near the decision boundaries, thus improving its effectiveness in identifying viable configurations for future tests.

To evaluate the coverage and global certainty of the test space for VALICY's stopping criterion, a ”geometrical” instance identifies and fills the largest unsampled volumes by placing test points at positions furthest from previously sampled points.

VALICY halts a job based on two criteria: achieving the predetermined certainty level consistently after a set number of runs, or when all configured test points have been used. Throughout the job, results are continuously sent to the application. Comprehensive analyses including performance comparisons of AI models, clustering of results for True and False values, identification of points closest to cluster centers as representative, and outlier detection through various methods (e.g., neighbor distance, cluster center distance) are available for export via the frontend. Plans are underway to define a ”safety envelope”, encompassing volumes of True values at a certain distance from the decision boundary, to ensure reliable operation within specified parameter ranges under the defined certainty level.

B. Model Evaluation with Blender

Blender [23] is an open-source 3D computer graphics software used for creating animated films, visual effects, 3D-printed models, motion graphics. Blender was used to create a basic scene, which tries to simulate the real-world use of PB. It consists of a realistic 3D human model, a pale wall as background, a camera and a light source to mimic real-world conditions. The camera is positioned to capture the human model, this way images of the human can be generated.

To run virtual validation with VALICY, Vitarex modified the publically accessible gitlab repository [13] and adjusted the provided validation sample Python code to account for the input parameter variations. The 4 input parameters of the validation process are lighting, radius (distance), phi (angle in the horizontal plane, with 0/360° being a frontal), theta (angle with respect to the horizontal plane).

During the testing phase every time a set of parameter values is received from VALICY, a Blender Python script is executed, which sets the position of the camera and the intensity of the light source. After that an image is captured of the human model. The image is fed to the keypoint detection model. Then resulting detections are evaluated primarily by counting the number of detected keypoints. The prediction is considered correct if at least 80% of the visible keypoints are detected. The prediction outcome is relayed back to VALICY, prompting

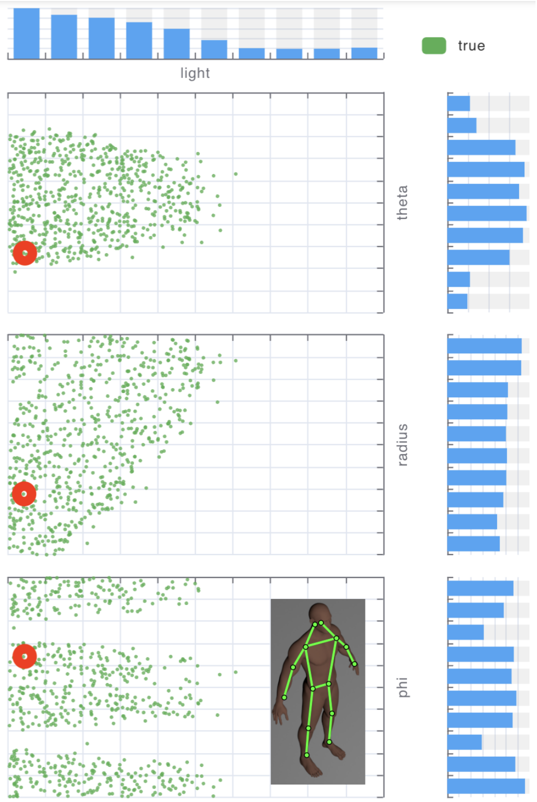


Fig. 3: Distribution of True results for the virtual validation runs of the key point detector plotted over the input parameter combinations (red circle is corresponding to the example).

the proposal of new parameter sets. For subsequent validation efforts, this threshold may be raised to enhance robustness.

After the completion of the testing, its results are displayed on the VALICY dashboard. See evaluation details in Figure 3. Based on the results, it can be concluded that the stronger the light source is, the bigger the distance between the camera and the person should be. It can also be clearly seen that the model cannot detect correctly when the phi parameter is between 77-110 degrees and 252-292 degrees, meaning when camera faces the side of the person. It is the same situation when the theta parameter is above 150 or below 33 degrees, which means the camera should not be placed too much above or below the person. In conclusion, one can say that, with the help of VALICY validation, one could determine the exact limits of the keypoint detection model, even more precisely than with the health visitors' findings.

VI. Conclusion

The development and field-testing of the PB device for scoliosis screening within the Hungarian health visitor system showcases the essential role of Machine Learning Operations (MLOps) in the successful deployment of AI technologies in healthcare. In this paper, the healthcare ecosystem in Hungary was shown, which is responsible for the screening of kids in schools. The development of an edgeML device (PB) for scoliosis screening was presented, and its potential was discussed that was tried out in a field study. It turned out that PB had some flaws that were rooted in the misbehaviour of the used ML model. These flaws could have been identified before its first real life deployment if the validation would have been done at an earlier stage (the validation was an international collaboration in the IML4E project [6], for which the device served as a testing ground).

Acknowledgment

The authors would like to acknowledge the contribution of their colleagues and partners to the work described in this paper. Our thanks go to former members of the development team at Vitarex, in particular to Dániel Kuknyó, Tamás Csarnó and Bence Várhidi, who all contributed to the development of PB. The authors would like to express their gratitude to Marianna Várfalvi for her professional advice (Health Visitor, chair of MVSZSZ, Professional Association of Hungarian Health Visitors).

Project no. 2019-2.1.1-EUREKA-2020-00016 has been implemented with the support provided by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, financed under the 2019-2.1.1-EUREKA funding scheme.

References

[1] A. Aszmann et al., "Health Protection in Public Education". Antikvárium Kiadó (Antiquarian Publisher), 2005.

[2] PoseNet. https://blog.tensorflow.org/2018/05/real-time-human-pose-estimation-in.html, accessed on 2024-04-26.

[3] Openpose. https://github.com/CMU-Perceptual-Computing-Lab/openpose, accessed on 2024-04-26.

[4] TRT_Pose. https://github.com/NVIDIA-AI-IOT/trt_pose, accessed on 2024-04-26.

[5] Coco API. https://github.com/cocodataset/cocoapi, accessed on 2024-04-26.

[6] The IML4E project. https://iml4e.org, accessed on 2024-04-26.

[7] NVIDIA Jetson Nano. https://developer.nvidia.com/embedded/jetson-nano-developer-kit, accessed on 2024-04-26.

[8] Raspberry Pi 4 model B. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/, accessed on 2024-04-26.

[9] Mlflow: A machine learning lifecycle platform. https://github.com/mlflow/mlflow, accessed on 2024-04-26.

[10] Flask Web Framework. https://flask.palletsprojects.com/en/3.0.x/, accessed on 2024-04-26.

[11] OpenCV. https://opencv.org, accessed on 2024-04-26.

[12] Stefánia Registration System. https://vitarex.hu/Stefania, accessed on 2024-04-26.

[13] VALICY repository. https://github.com/SpicetechGmbH/Valicy-Interface-Example, accessed on 2024-04-26.

[14] Google Cloud Architecture Center. Mlops: Continuous delivery and automation pipelines in machine learning. https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning, accessed on 2024-05-22.

[15] A. Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024-8035, Curran Associates, Inc., 2019.

[16] M.-R. Ronchi and P. Perona. Benchmarking and error diagnosis in multi-instance pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[17] Fortissimo908-success-story. https://www.fortissimo-project.eu/en/success-stories/908/massively-parallel-virtual-testing-of-safetyrelevant-driving-systems, accessed on 2024-04-26.

[18] J. Lapins et al., Massively Parallel Virtual Testing of Safety-Relevant Driving Systems. *Proceedings of 7th AutoTest Technical Conference*, Stuttgart, 2018.

[19] K. Gao, S. Hekeler, and M. Kütemeyer, Combination of virtual and real live tests of safety relevant driving functions (Original title: Kombination von virtuellen und realen Tests sicherheitskritischer Fahrfunktionen). ATZ Elektron 15, pp. 66–70 (2020). https://doi.org/10.1007/s35658-020-0268-1, accessed on 2024-04-26.

[20] VALICY REST-API description. https://api.valicy.de/docs, accessed on 2024-04-26.

[21] B. Pukánszky and A. Németh (1996): Neveléstörténet (Education history), Nemzeti Tankönyvkiadó Rt (National Textbook Publishing Co).

[22] M. Kachlichné Dr. Simon and M. Várfalvi (2020): Health visitors' web history museum. http://mvszsz.hu/index.php/hu/webmuzeum, accessed on 2024-05-22.

[23] Blender website. https://www.blender.org/, accessed on 2024-05-22.