Proceedings of the Estonian Academy of Sciences
2025, **74**, 2S, 302–311

https://doi.org/10.3176/proc.2025.2S.05

www.eap.ee/proceedings
Estonian Academy Publishers

EMBEDDED IMAGING FLOW CYTOMETRY

RESEARCH ARTICLE

Received 16 December 2024
Accepted 2 April 2025
Available online 9 June 2025

Keywords:
droplet, droplet classification, neural networks, resource constrained platform, single board computer

Corresponding author:
Fariha Afrin
fariha.afrin@taltech.ee

Citation:
Afrin, A., Le Moullec, Y., Pardy, T. and Rang, T. 2025. Lightweight CNN-based microfluidic droplet classification for portable imaging flow cytometry. *Proceedings of the Estonian Academy of Sciences*, **74**(2S), 302–311. https://doi.org/10.3176/proc.2025.2S.05

Lightweight CNN-based microfluidic droplet classification for portable imaging flow cytometry

Fariha Afrin, Yannick Le Moullec, Tamas Pardy and Toomas Rang

Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology (TalTech), Ehitajate tee 5, 19086 Tallinn, Estonia

ABSTRACT

Classifying microfluidic droplets is an essential step in imaging flow cytometry. While deep learning algorithms can detect and classify such droplets in benchtop laboratory settings, their deployment on portable devices remains challenging because the computational requirements often exceed the capabilities of compact, resource-limited devices. This hinders the transition from stationary lab setups to field-deployable instruments. To tackle this issue, we introduce a customized YoloV4-tiny model deployed on a Raspberry Pi-5 (RPi5) single-board computer. Our neural network is trained using 878 images from a custom dataset of 975 images, derived from two videos captured with real-life microfluidic experimental setup. We evaluate performance based on inference time and mean average precision. Our system successfully classifies three distinct droplet types (no cell, one cell, multiple cells) within 13 ms, achieving a 99.95% mean average precision at an intersection over union threshold of 0.5 (mAP@0.5). We also compare the classification performance metrics of our customized YoloV4-tiny model against seven other combinations of machine learning models and platforms, including a recent low-cost, highly compact edge device with tensor processing unit capabilities, specifically, the MaixCam board with LicheeRV Nano module (SOPHGO SG2002) running a YoloV5-s model. Compared to this proposed customized YoloV4-tiny on the RPi5, the YoloV5-s on MaixCam achieves a significantly shorter classification time (5.34 ms) owing to its onboard tensor processing unit but suffers from a lower mAP@0.5 of 55.09% due to quantization. Our work shows that carefully designed systems can achieve a balance between speed and accuracy, enabling robust performance even on resource-limited devices and paving the way for microfluidic droplet classification in portable imaging flow cytometry.

Introduction

Microfluidic droplet classification offers remarkable potential for biomedical research and applications [1,2]. In parallel, recent progress in machine learning (ML) and deep learning has enabled (near) real-time, automated, reliable, and precise object classification. In particular, convolutional neural networks (CNNs), a subset of deep learning models, have led to advances in object detection and classification in various fields, including biomedical imaging [3,4]. Given their ability to extract complex features and patterns from image datasets, CNN models show promise in automating and enhancing the droplet classification phase in imaging flow cytometry (IFC).

Nevertheless, current ML-based droplet classification methods are predominantly designed for benchtop (laboratory-grade) applications, typically relying on high-performance graphics processing units (GPUs). Implementing ML-based droplet classification on embedded/edge devices remains challenging due to: (i) resource constraints (limited computational power, memory, and energy compared to laboratory-grade systems), (ii) the need to customize, adapt, and optimize large models and complex algorithms to run efficiently on resource-constrained devices, (iii) requirements for possible (near) real-time processing and speed vs accuracy trade-off on low-power devices; and (iv) the potential availability and exploitation of specialized hardware, such as low-power tensor processing units (TPUs) and neural processing units (NPUs). This paper is an extended version of [5], providing additional references, more details about model customization and deployment, as well as additional deployment comparisons.

In [6], researchers developed a CNN-based algorithm called the weakly supervised cell counting network (WSCNet) to classify cell-encapsulated droplets. The proposed method significantly improved the accuracy of traditional image

classification (approx. 89%) and exhibited robustness under different lighting conditions; however, WSCNet relied on a desktop computer for operation.

Another research effort aimed to improve the accuracy and efficiency of detecting microfluidic droplet contents in liquid biopsy workflows using CNN [7]. This CNN-based automatic classification system achieved 96% precision in droplet classification but was not designed for resource-constrained platforms.

Transfer learning with deep CNNs has been used to classify cellular morphological changes, achieving high accuracy between 95% and 97% [8]. Although this approach reduces the need for extensive dataset collection and labeling, which is a significant bottleneck in the development of deep learning models for specific microfluidic applications, it is not suitable for resource-constrained platforms.

In [9], researchers developed an optimized image-activated cell sorter based on a deep learning model specifically tailored to classify and sort polystyrene beads and cells in real time. However, it was deployed using the TensorRT29 framework on an NVIDIA GeForce GTX 1080 TI GPU.

The authors of [10] developed a ML-based computer vision solution for real-time detection of droplets and bubbles. Their approach used the Yolov5 framework with custom pre- and post-processing techniques, trained on a dataset of 5115 images. Their results demonstrated real-time speed and high accuracy in detecting and differentiating between droplets and unwanted bubbles, but the method required a relatively high-end PC.

In [11], researchers explored the application of computer vision and deep learning techniques to automate the analysis of yeast cell replicative lifespans. They compared Yolo and Mask R-CNN in terms of their efficacy in detecting and analyzing yeast cells from microfluidic images. They found that Yolo demonstrated superior sensitivity in cell detection, while Mask R-CNN provided more detailed information on cell sizes. However, their work was implemented on a desktop GPU (RTX 2080 Ti).

The authors of [12] developed a deep learning pipeline for high-throughput, label-free cell classification, using CNNs to process raw measurement signals and enabling low-latency inference suitable for real-time cell sorting applications. While their method demonstrated over 95% accuracy in label-free classification of specific types of white blood cells and epithelial cancer cells, their system was deployed on a desktop GPU (Tesla K80).

In [13], researchers proposed a label-free chemical IFC that combines pulse pair-resolved wavelength-switchable Stokes laser, multicolor stimulated Raman scattering (SRS) microscopy, and a 3D acoustic focusing microfluidic chip, supported by deep learning algorithms. They achieved a throughput of approximately 140 cells/s; however, they did not indicate the platform used (we assume that it was a high-end PC).

The research presented in [14] applied CNNs for processing large-scale datasets of label-free cell images for high-throughput cell classification. The authors compared CNN performance against k-nearest neighbors (kNN) and support vector machine (SVM) methods. Their CNN-based approach yielded over 99% accuracy in identifying multiple cell types based on label-free bright-field images. However, they used a desktop GPU (Tesla K40c).

The work presented in [15] introduced a rapid and label-free antimicrobial susceptibility testing method for colistin, combining deep learning with droplet microfluidics. The DropDeepL AST method used a deep learning-powered approach for sensitive detection of bacterial growth in droplets, achieving 100% categorical agreement with the reference broth microdilution method for colistin susceptibility profiles. However, the paper did not specify the type of platform on which the model was trained or deployed; we assume that it was not a resource-constrained platform.

A deep learning-augmented T-junction droplet generation system was presented in [16]. The study used finite element analysis to simulate droplet production and its dynamics, followed by ML algorithms to estimate droplet characteristics based on input parameters. This approach enabled preselecting designs with comparable microfluidic configurations within the studied range. Nevertheless, the specific platform used for training and deploying the model was not disclosed; however, our assumption is that it was not a resource-constrained platform.

Finally, [17] developed a droplet-based microfluidic platform to detect peptides that are self-secreted by yeast. They used ML-based image processing techniques to analyze fluorescence emitted by single yeast cells in droplets, yielding high-throughput analysis and characterization of agonistic peptides. However, the paper did not specify the training or deployment platform; we also assume it was not a resource-constrained platform.

Despite the notable progress in microfluidic droplet object analysis, a research gap remains in the scientific literature related to model optimization for portable IFC devices. Many research efforts have focused on maximizing classification accuracy, often at the cost of increased computational demands, making them impractical for portable platforms. Our research contributes to bridging this gap by refining a deep learning-based droplet classification system for use in portable devices, demonstrating its viability even with resource limitations.

To address this issue, we have built a portable droplet classification system that leverages the YoloV4-tiny model. This model was chosen for its balance between efficiency and accuracy; YoloV4 remains well suited for embedded systems due to its compact architecture, which allows for rapid inference without compromising detection capabilities. We specifically selected YoloV4-tiny because it offers a significant reduction in computational requirements compared to its full-size counterpart, making it ideal for deployment on resource-constrained devices such as the Raspberry Pi-5 (RPi5) [18]. Furthermore, we identified the need to customize this model for our specific use case. Customization is necessary to enhance the model's performance in classifying droplets within our unique microfluidic setup. By fine-tuning the model on our custom dataset of microfluidic droplets, we ensure that it can accurately distinguish between different droplet types (no cell, one cell, multiple cells). This customization process

allows us to optimize the model's architecture and parameters, resulting in improved accuracy and faster inference times specifically for our droplet classification task. This also aligns with prior work on optimizing ML models for energy-efficient applications, where ML has been used in low-power applications [19]; in a similar vein, our approach explores the feasibility of deploying deep learning models on resource-limited embedded systems by balancing accuracy, inference speed, and computational complexity.

We evaluate performance in terms of inference time and mean average precision. On the RPi5, our system successfully classifies three distinct droplet types (no cell, one cell, multiple cells) in 13 ms, while maintaining over 98% accuracy. We compare the classification performance metrics of our customized YoloV4-tiny model against seven other models/platforms, including a recent, low-cost and highly compact edge device with TPU capabilities. The main steps and contributions of this research are as follows:

- We generate a new custom dataset of 975 microfluidic droplets images. The dataset is created from two videos recorded in microfluidic experimental setup.
- We introduce a droplet classification method (based on the YoloV4-tiny model) that offers high accuracy and inference speed, while remaining lightweight for resource-constrained implementation. High accuracy is achieved through data augmentation, and high inference speed is achieved by reducing the number of filters in the convolutional layers by 20% and applying batch processing (six images per batch).
- Our neural network is trained using 878 images extracted from the custom dataset of 975 images. We first implement and deploy our proposed approach on two resource-constrained single-board computers (SBCs): initially on a Raspberry Pi-4B (RPi4) with a BCM2711 chip and 8 GB RAM, and then on an RPi5 with a BCM2712 SoC and 8 GB RAM, achieving high accuracy (98%) and fast inference (13 ms). We then compare our results with those obtained on other platforms, including the compact and low-cost MaixCam board [20] with an SG2002 SoC featuring a TPU[1] and 256 MB RAM.

We describe our customized model, built on deep learning principles, and how it attains both high accuracy and rapid performance, allowing for effective analysis of microfluidic droplets containing cells. Furthermore, we evaluate our system using a new and previously unexamined test dataset, highlighting its robustness. By integrating deep learning methods into a resource-limited platform, our work supports the development of resource-efficient droplet classification, thereby advancing the field of portable microfluidic technology.

The remainder of this paper is organized as follows. Section 2 describes the materials and method used in this work, particularly focusing on model implementation for resource-constrained devices. Section 3 presents the results, including a comparative performance analysis. Section 4 concludes the paper and suggests potential future research.

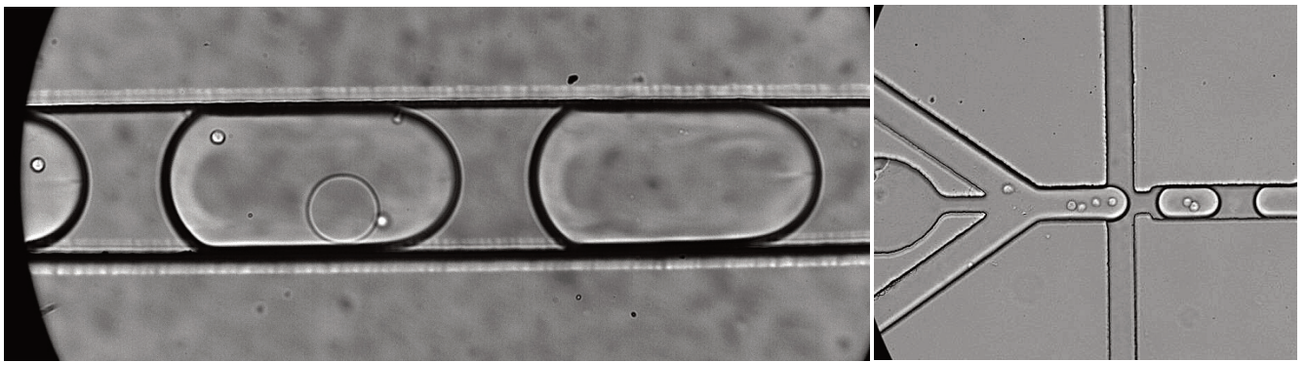
2. Materials and method

2.1. Dataset

The acquisition of training data is a major bottleneck in advancing microfluidic object detection and classification due to the scarcity of available datasets. In our work, a camera positioned directly above the microfluidic chip records the experimental video stream, capturing droplets formed at the T-junction of two channels. To extract images for model training, we used two distinct experimental videos [21] recorded under varying conditions (illustrative still images extracted from these videos are presented in Fig. 1). The images were retrieved by capturing video frames at a rate of 15 frames per second (fps), rather than the standard 1 fps, and were stored in JPG format. Each image contains either one or two microfluidic droplets, and each droplet may contain either no cell, a single cell, or multiple cells.

Next, to detect and classify such droplets in images, we consider an ML-based approach based on CNNs. However,

(a) (b)



An empty droplet (right) and a droplet containing multiple cells (left)

A droplet containing two cells (right) and a droplet under formation (left)

Fig. 1. Images of droplets flowing inside the microfluidic channel. Original resolutions are 1024 × 416 pixels for (a) and 560 × 512 pixels for (b); both are resized to 416 × 416 pixels during the training phase of the proposed customized YoloV4-tiny model.

[1] Similar to an NPU for low-precision operations (INT8) used in edge AI inference.

while CNNs are powerful tools for image classification, their performance can be negatively impacted by a lack of diversity in the training dataset. Without sufficient variety, the model may struggle to generalize well to new, unseen data. This may lead to overfitting, where the model performs well on training data but poorly on new examples. To address this issue, the two above-mentioned video recordings offer some diversity by capturing variations in illumination, contrast, motion blur, and object positioning, thereby helping to reduce overfitting.

However, to further combat the risk of overfitting, we implemented a data augmentation strategy to synthetically expand the size and diversity of the training dataset by creating modified versions of existing images.

We applied the following augmentation techniques:

- Contrast modification: the image contrast was adjusted within a range of 0.4 to 1.6, simulating variations in lighting conditions and image quality that may occur in real-world scenarios.
- Gaussian blur: two ranges of Gaussian blur were applied: 0.3 to 0.9 and 1.1 to 2.5. This technique mimics different levels of focus or image clarity, helping the model become more robust to variations in image sharpness.
- Vertical and horizontal rotations: these transformations enable the model to recognize objects regardless of their orientation within the image.

The augmentation process significantly expanded the dataset to a total of 975 images. This expanded dataset was then split into two parts, i.e., training and validation set, and testing set, as follows:

- Training and validation set: 878 images (approx. 90% of the total 975 images), with 791 images (approx. 90% of the 878 images) used for training and 87 images (approx. 10% of the 878 images) for validation[2]. This validation set plays a vital role in assessing the model's generalization capabilities and helps prevent overfitting during the training process.
- Testing set: 97 images (approx. 10% of the total 975 images).

This split ensures a substantial amount of diverse data for training and validation while reserving a separate set for final testing to evaluate the model's performance on unseen data.

2.2. Image annotation

The process of image annotation is a crucial step in preparing datasets for object detection tasks, such as identifying droplets in IFC systems. This procedure requires recognizing and precisely marking the locations of all target objects within each image.

To conduct the annotation, we leveraged the Make Sense AI tool [22], which provides an intuitive interface for manual labeling and allows for meticulous, high-precision annotation of each image. We opted for rectangular bounding boxes as our annotation method, because they effectively capture the spatial extent of droplets within the images. The annotation process consisted of the following steps:

- Uploading images to the makesense.ai platform;
- Carefully examining each image for droplets;
- Drawing rectangular bounding boxes around each identified droplet;
- Verifying the accuracy of annotations through multiple reviews;
- Exporting the annotation data in the required format (.txt files).

The annotation format we adopted aligns with the YoloV4-tiny architecture requirements, using the [x, y, w, h] convention. In this format, (x, y) represents the center point of the bounding box, providing the focal point of the detected object; w denotes the width of the bounding box, capturing the horizontal extent of the droplet; and h stands for the height of the bounding box, representing the vertical extent of the droplet.

2.3. Proposed customized CNN model and deployment

In response to the requirements set forth by a bioanalytical specialist, we categorized the droplets into three groups: (i) empty droplets, (ii) droplets containing a single cell, and (iii) droplets containing multiple cells. Only droplets that are entirely visible in an image are factored into the classification process.

Our approach leverages the YoloV4-tiny model as a foundation. We implemented a refinement process to optimize the model for our specific use case of microfluidic droplet detection, with a focus on inference speed without significantly compromising accuracy, as outlined in what follows.

The original YoloV4-tiny architecture consists of 21 convolutional layers, organized into a series of cross-stage partial network (CSPSNet) blocks. A key modification in our customization process involved a careful reduction in the number of filters in these convolutional layers, as shown in Fig. 2. Through a series of ablation studies, we empirically determined that a 20% reduction in filter count provides a suitable trade-off between model complexity and accuracy, as explained below. Each cross-stage partial (CSP) module consists of a convolutional layer followed by batch normalization and the Leaky-RELU activation function, collectively referred to as CL. The module also features skip connections, which help achieve an optimal balance between detection efficiency and accuracy. Three CSP modules in the backbone progressively extract the image features. For the detection head, two heads are used for detecting larger and smaller objects. The detection section consists of one CL block and a convolutional layer, followed by the detection layer. The feature maps are upsampled and combined with residual connections from the same feature map resolution for the detection head at the second scale. The first scale, with a feature map size of 13 × 13, is used for larger objects, while the second scale, with a feature map size of 26 × 26, targets smaller objects.

To ensure that the accuracy remained within acceptable bounds, we used an iterative process of filter reduction and performance evaluation. Through a series of ablation studies, we empirically tested different filter reduction levels (10%, 20%, and 30%). We used a validation set to monitor the

[2] A 10% validation split is commonly used for microfluidic applications with relatively small datasets, such as in [4], which used 786 images.

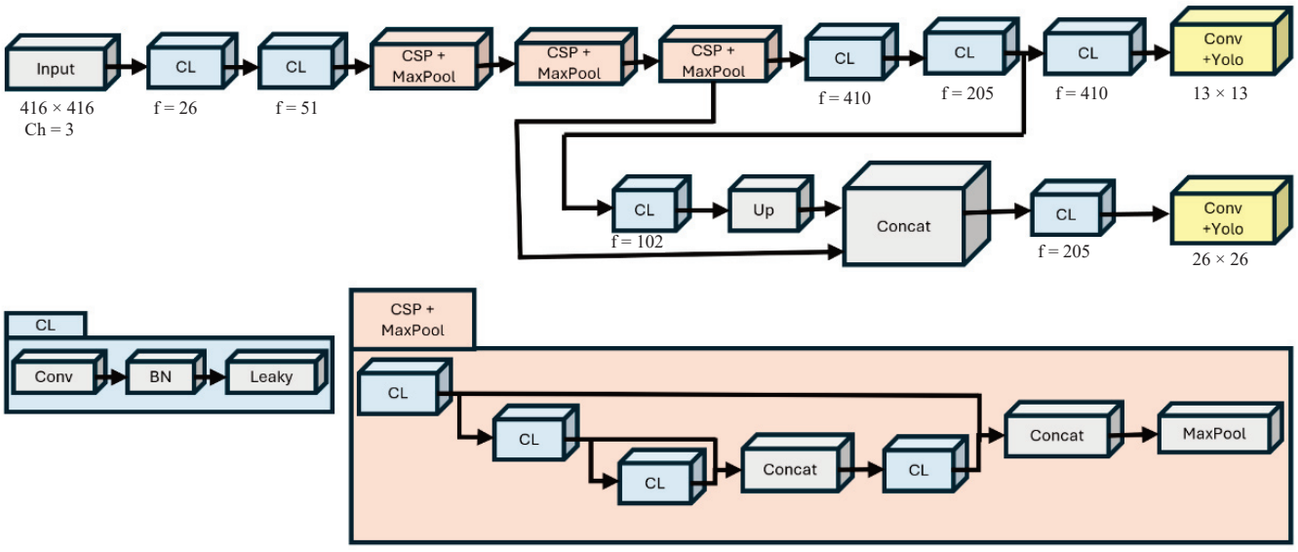


Fig. 2. Proposed customized YoloV4-tiny architecture. The network accepts a three-channel input (ch = 3), and its backbone comprises a series of convolutional layers (CL) and cross-stage partial (CSP) blocks combined with MaxPooling. Skip connections and upsampling (Up) are used for multi-scale feature fusion via concatenation. The number of filters is indicated by f, and the final stage includes two detection heads. Abbreviations: Conv – convolutional, BN – batch normalization.

model's performance after each round of filter reduction, ensuring that any accuracy drop remained minimal. Based on previous studies and our experimental results, a 20% filter reduction typically results in a 1–5% decrease in mean average precision (mAP), while batch processing may introduce an additional 0.5–2% loss due to batch normalization effects. In our case, empirical evaluation showed that the accuracy drop remained within this expected range, making the trade-off acceptable for our specific application. The 20% filter reduction resulted in a significant 36.16% decrease in the weight file size, from 22.4 MB to 14.3 MB.

In practice, this filter reduction strategy serves four purposes, listed below.

First, by reducing the number of filters, we significantly decrease the total number of parameters in the model. This compression result in a smaller model size, which is crucial for deployment on resource-constrained edge devices used in portable microfluidic systems.

Second, fewer filters result in fewer computations during the forward pass of the network. This directly translates to faster inference times, which are critical for real-time droplet detection and classification in flow cytometry applications.

Third, reducing the model's complexity through filter reduction can help mitigate overfitting, especially when working with limited datasets, which are common in specialized scientific applications, such as microfluidic droplet analysis.

Fourth, the reduced model size requires less memory during both training and inference, making it more suitable for deployment on devices with limited RAM.

Then, we fine-tuned the hyperparameters to optimize the performance of our customized YoloV4-tiny model. After extensive experimentation, we settled on a final learning rate of 0.00261, which was determined through a cyclical learning rate test to find an optimal balance between convergence speed and stability. This learning rate was coupled with a weight decay rate of 0.0005 to prevent overfitting and improve generalization.

Finally, we set a momentum of 0.9 in the stochastic gradient descent optimizer, which helped accelerate convergence and mitigate oscillations during training. The size of the batch of images was set to 64 to balance memory constraints and the need for stable gradient estimates. This batch size was further divided into eight subdivisions and the training process was configured to run for a maximum of 6000 batches, following the default settings recommended for YoloV4-tiny.

For training, we used the Darknet framework [23] on Google Colaboratory Pro, with a T4 GPU configured with a software environment including Python 3.10.12, CUDA 12.2, cuDNN 8.9.6, and OpenCV 4.8.0. The trained model processes each input image to assign a confidence score to each detected droplet. We assessed performance by evaluating accuracy and mean average precision on the validation dataset. As depicted in Fig. 3, the model achieved optimal performance at 6000 iterations, reaching a training loss value of 0.09.

3. Deployment results

This section first outlines the outcomes of our proposed system for classifying microfluidic droplets, designed particularly for integration into portable devices. Our analysis emphasizes three primary aspects: classification accuracy, inference processing time, and resource consumption on a resource-constrained device. The section then presents a comparison with similar studies.

3.1. Droplet classification performance evaluation

We evaluated the performance of droplet classification using an input size of 416 × 416 pixels and a non-maximum-suppression threshold set to 0.7 for our test datasets. Figure 4 illustrates examples of droplet detection outcomes on images from our test dataset.

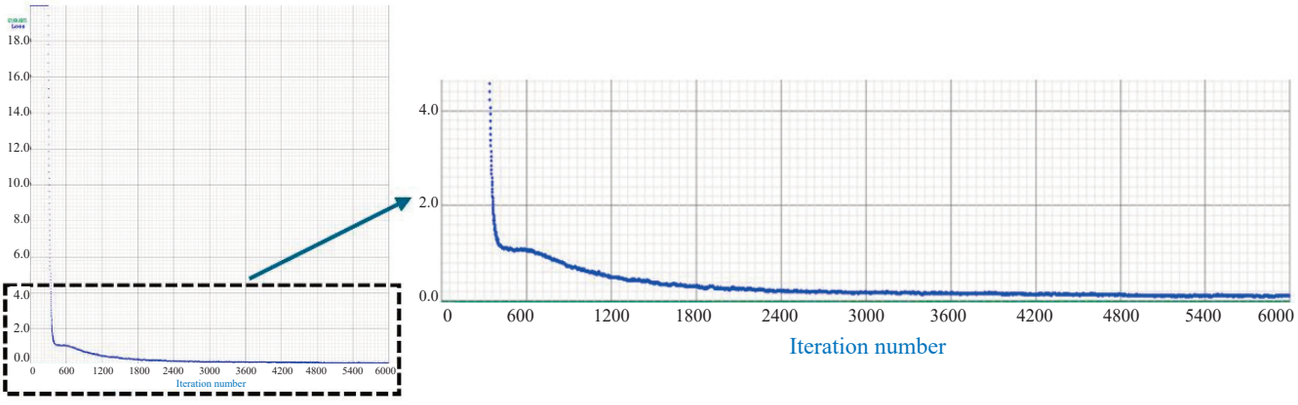
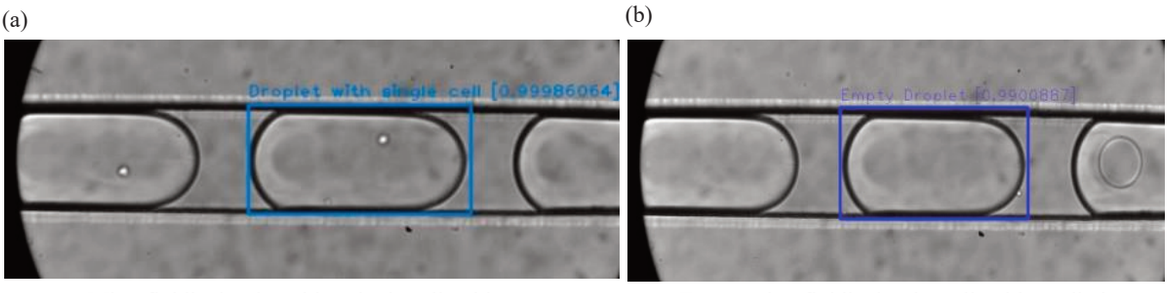


Fig. 3. Left: overview of the training loss function for YoloV4-tiny with compressed filters, trained for up to 6000 iterations, at which point the average loss reaches 0.09; right: zoomed-in view showing the convergence of the training loss function.



Microfluidic droplet with a single cell, with prediction probability above 99%

Empty microfluidic droplet cell, with prediction probability above 99%

Fig. 4. Examples of microfluidic droplet classification results after 6000 training iterations, with prediction probability above 99%.

3.2. Custom model performance

Table 1 presents the performance metrics of our customized YoloV4-tiny model, which are derived using Eqs (1) and (2). In these equations, *TP* represents true positives, *FP* signifies false positives, and *FN* stands for false negatives:

$$Precision = TP/(TP + FP), \tag{1}$$

$$Recall = TP/(TP + FN). \tag{2}$$

The model demonstrates substantial reliability and effectiveness in accurately distinguishing various droplet types, as shown by its high precision of 0.95 and perfect recall of 1, resulting in minimal misclassification. The mAP at an intersection over union (IoU) threshold of 0.5, denoted as mAP@0.5, is a reflection of the accuracy, where IoU measures the overlap between predicted and actual bounding boxes. The robustness of the model is significant, with a notable mAP of 99.95%.

Table 1. Classification performance metrics of the proposed customized YoloV4-tiny model (test dataset = 97 images of droplets, each containing zero, one, or multiple cells)

Metric	Performance value (%)
TP	112
FP	6
FN	0
Precision	0.95
Recall	1
mAP@0.5	99.95

Our implementation was evaluated on both RPi4 and RPi5 SBCs, each equipped with 8 GB of RAM and running on a 64-bit operating system. High-speed image processing is crucial for real-time applications. To assess processing efficiency, we conducted tests with the model using the appropriate test dataset on the specified devices. The test results showed that the processing duration on the RPi5 was 20 times faster (13 ms) compared to RPi4 (265 ms).[3] On the RPi5, the model achieved a classification inference time of 13 ms, equating to 76 fps; this speed, obtained with a six-image batch, suffices for practical applications. Given these superior results with the RPi5, the RPi4 was not considered for further analysis.

As noted above, these results were obtained using batches of six images rather than a single image. We evaluated inference speed across batch sizes ranging from 1 to 16 images and observed that processing time improved up to a batch size of six, beyond which no further improvements were noted. Subsequently, all results were averaged, and inference time was calculated for a single input image.

Beside inference processing time, resource utilization was an important part of our assessment due to the limitations of portable devices. We tracked CPU and memory consumption during classification tasks to guarantee the system's performance efficiency. As shown in Table 2, the mean CPU usage

[3] We also transformed our customized model into TensorFlow and then TensorFlow Lite (TFLite) to try to reduce the inference processing time. However, the inference processing time for a single image exceeded 1 s on RPi4.

Table 2. Resource utilization of the proposed customized YoloV4-tiny model on RPi5

Resource	Average usage
CPU usage	70%
Memory usage	60%

and memory consumption on the RPi5 were 70% and 60%, respectively. These values suggest that the proposed approach is adequately lightweight, making it applicable for portable use, since CPU and memory consumption remain within practical limits.

3.3. Comparison with similar studies

In this section, we juxtapose the classification performance metrics of our optimized YoloV4-tiny model results against seven other models/platforms. It should be noted that direct one-to-one comparisons are not feasible because (i) previous works do not focus on the same applications as ours, and (ii) many of these studies were implemented on high-performance desktop PCs and/or GPUs. Nevertheless, such a comparison helps get a better understanding of the trade-offs between accuracy and inference processing time.

To balance inference processing time and accuracy, we leveraged model optimizations, hardware-aware deployment strategies, and empirical comparisons. Notably, we deployed our customized YoloV4-tiny on an RPi5, and also retrained YoloV5-s using our dataset for deployment on the MaixCam board. This additional training and deployment were conducted as part of this work and are briefly described below, prior to presenting the overall comparison table.

3.4. Training and deployment of the YoloV5-s model onto the MaixCam board

The MaixCam board [20] is a prime example of a recent (July 2024), low-cost (approx. 34 EUR), and highly compact (22.86 × 35.56 mm; see Fig. 5) edge device with an embedded neural processing unit. The MaixCam board is based on the LicheeRV Nano module [24], built around a SOPHGO SG2002 system on chip (Soc) [25]; it features a primary 700 MHz RISC-V C906 core and a secondary (boot-selectable) 1 GHz RISC-V C906 core or 1 GHz ARM A53 core, along with 256 MB of on-chip RAM. Notably, the SG2002 features a TPU capable of up to 1 tera operations per second (TOPS) @INT8, which should help reduce the inference processing time.

We used the same dataset described earlier in the paper. We converted the labeled dataset from a Darknet-compatible format (.jpg and .txt files) to a VOC-compatible format (.jpg and .xml), and uploaded the data to the MaixHub environment [26] for training the model and generating a format compatible with the SG2002 chip on the MaixHub board (.mud and .cvi files).

We also experimented with several hyperparameters, such as batch size and learning rate. The best validation accuracy results that we obtained empirically were with a batch size of four and a learning rate of 0.0001, which yielded a validation accuracy of 0.966 at epoch #120.

3.5. Contrasting combinations of ML models and their deployments

As mentioned earlier, a direct comparison of the different combinations of ML models and their deployments on various hardware targets is not feasible; however, some insights can be derived from Table 3.

Firstly, when compared to desktop deployments, it can be clearly seen that our customized YoloV4-tiny model deployed on the RPi5 delivers a smaller inference processing time of 13 ms, compared to the 74.3 ms reported in [10] for single-class detection using YoloV5 on an Intel Core i7, along with comparable mAP accuracy values (99.95%@0.5 vs 99.3%@0.5, respectively). It is also worth noting that studies using high-end PCs or GPUs ([12–14]) achieved both high mAP and low processing times (where such metrics were reported); however, the GPUs used – such as the NVIDIA Tesla K80 in [12] and the Nvidia Tesla K40c in [14] – are dual-slot PCIe cards (267 mm × 111 mm), with a computational power of approximately 8.74 TFLOPS@FP32 / 2.91 TFLOPS@FP64 and 4.29 TFLOPS@FP32 / 1.43 TFLOPS@FP64, respectively. These boards have power requirements of up to 375 W and 245 W, respectively. Such specifications clearly position these solutions for non-portable applications due to the physical size and power requirements of their processing units.

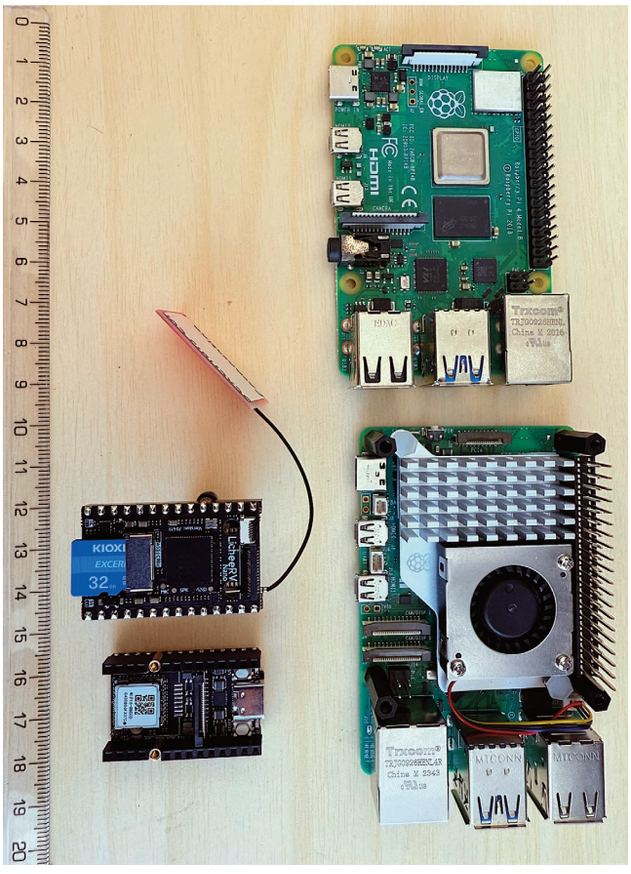


Fig. 5. Boards used in this work. Clockwise from the top: RPi4 SBC (used only up to Section 3.2 in this paper), RPi5 SBC with heatsink/fan, and MaixCam edge devices (top side with SG2002 SoC, bottom side with WiFi chip), along with a metric ruler. As can be seen, the MaixCam is a highly compact board owing to its minimal connectivity options (USB and WiFi), yet featuring a SoC with a 1 TOPS@INT8 TPU.

Table 3. Combinations of machine learning models and their deployments

Reference	Model	Platform	Object	mAP, %	Average inference processing time per image
[12]	CNN	NVIDIA Tesla K80 GPU	Cells	95.7	2.2 ms
[13]	CNN	High-end PC (assumed)	Cells	93–99	Not reported in the paper
[14]	CNN	Nvidia Tesla K40c GPU	Cells	99	Not reported in the paper
[11]	Mask-RCNN	Nvidia RTX 2080 Ti GPU	Cells	73	Not reported in the paper
[10]	YoloV5	Intel Core i7-12650H	Droplet	99.3@0.5	74.3 ms
This work	**Customized YoloV4-tiny**	**RPi5**	**Droplet**	**99.95@0.5**	**13 ms**
This work	YoloV5-s	RPi5	Droplet	92.10@0.5	208.5 ms
This work	YoloV5-s	MaixCam	Droplet	55.09@0.5	5.34 ms

In contrast, the RPi5 measures only 85.6 mm × 56.5 mm and has a power requirement of up to 25 W only, providing approximately up to 12 GFLOPS@FP32/CPU, 20 GFLOPS@FP32/GPU, and 750 MFLOPS@FP64/CPU. (Note: The RPi4's Cortex-A72 CPU lacks efficient FP64 acceleration and native GPU acceleration, so Yolo is executed as FP32/CPU.) The MaixCam features even more compact specifications: 22.86 × 35.56 mm in size, up to 2.5 W power consumption, approximately 2.8–4 GFLOPS@FP32, 250 MFLOPS@FP64, and 1 TOPS@INT8/TPU (Yolo is accelerated on this TPU).

Secondly, when comparing the resource-constrained deployments, it can be seen that the MaixCam running the YoloV5-s model achieves a smaller inference processing time (5.34 ms) than the customized YoloV4-tiny on the RPi5 (13 ms); however, this comes at the cost of a much lower mAP of 55.09%. This lower score stems from the more aggressive quantization required to map the model onto the INT8 TPU of the SG2002 SoC on the MaixCam board.

We also trained and deployed the YoloV5-s model for the RPi5; while it achieved a mAP@0.5 of 92.10%, the classification time was high at 208.5 ms, indicating that this combination is not favorable.

Due to a yet unsolved issue with the toolchain, we were unable to convert our customized YoloV4-tiny model for the MaixCam. Instead, we provide estimated performance numbers, as explained below. Using the YoloV5-s results, we can derive an approximate performance scaling factor between the MaixCam and RPi5 platforms:

- Speedup factor (MaixCam vs RPi5 for YoloV5-s): 208.5 ms (RPi5) / 5.34 ms (MaixCam) ≈ 39.1×. This reflects that the MaixCam's TPU accelerates inference significantly compared to the RPi5's CPU-based inference.
- Accuracy drop (YoloV5-s on MaixCam vs. RPi5): from 92.10@0.5 to 55.09@0.5 ≈ 40.2%. This reflects that the MaixCam's TPU has precision limitations due to lower-bit computation, i.e., only INT8 (MaixCam) instead of FP32 (RPi5).
- Since YoloV4-tiny is structurally similar to YoloV5-s, we assume the same performance ratio applies. Using the 39.1× speedup factor yields 13 ms / 39.1 ≈ 0.33 ms. This suggests that the MaixCam TPU could theoretically process our customized YoloV4-tiny model in under 1 ms; however, real-world constraints (e.g., memory access, TPU overhead) would likely increase inference time to around 1–2 ms. On the other hand, we expect the mAP as 99.95 × (100 – 40.2) ≈ 59.8%; i.e., running our customized YoloV4-tiny model on the MaixCam TPU might reduce accuracy to around 60%, which is not favorable despite the small processing time.

Besides the above accuracy and processing time performance results, it should also be noted that the RPi5 is an SBC with power requirements of up to 25 W (5 V, 5 A). This is not necessarily a major concern; for example, our portable setup [27] uses the RPi as a common platform, as this allows implementing additional functionalities on a single board. On the other hand, for applications where power and/or energy requirements are more stringent, a board such as the MaixCam – with an approximately 10× lower power requirement of 2.5 W (5 V, 500 mA) – would be more suitable, if the cost of lower accuracy is acceptable.

To sum up, the above results illustrate that our customized YoloV4-tiny on the RPi5 offers a good trade-off among the different tested combinations, offering significant accuracy with competitive processing time on a resource-limited platform, which makes it suitable for portable IFC devices.

4. Conclusion

This work demonstrated the feasibility of effective droplet classification in IFC on a resource-constrained device using a customized YoloV4-tiny model. A new dataset of droplet images was created from videos recorded on our existing setup, with improved accuracy and robustness achieved through data augmentation. The model's inference processing time was reduced by cutting the convolutional layer filters by 20% and using batches of six images. Our proposed system can accurately classify droplets in 13 ms, achieving an accuracy surpassing 99% when running on an RPi5 SBC. Additional experiments with YoloV5-s on the compact MaixCam board, featuring an SG2002 SoC with a TPU, illustrated how a significantly smaller classification time (5.34 ms) must be traded off for accuracy (mAP@0.5) due to quantization.

Future work could focus on decreasing inference processing time on resource-constrained platforms by applying more sophisticated pruning and quantization techniques, aiming for real-time or near-real-time classification while preserving accuracy. Moreover, we could explore other Yolo versions; while larger models (e.g., YoloV4, YoloV5-m/l/x) might offer higher accuracy, they would increase computational load, limiting real-time feasibility. Conversely, YoloV7-

tiny and YoloV8-nano may enhance accuracy while remaining efficient for edge deployment, though their optimization for our target hardware requires further investigation.

While we experimented with three different platforms, it should be noted that the landscape of edge AI hardware is rapidly evolving. For example, the recent Hailo-8 HAT for the RPi5 promises up to 26 TOPS (INT8). These developments offer promising avenues for further decreasing processing time while balancing accuracy; however, it is also essential to consider the maturity of these new solutions. Their integration (including the necessary software tools) into specialized scientific applications, such as IFC devices, will require additional careful evaluation and testing. We will investigate these issues in our future work.

Data availability statement

All data and code supporting this work are available at https://github.com/taltechloc/sw-open-droplet-analysis.

Acknowledgments

This work was supported by the Estonian Science Agency ETAg project "Cognitronic Lab-on-a-Chip System for Highly-Automated Flow Cytometry" (CogniFlow-Cyte, grant No. PRG620). The authors also acknowledge support from the Estonian Education and Youth Board project "Artificial Intelligence, Edge Computing and IoT Solutions in Distributed Systems" (AIoT*5G, grant No. ÕÜF11), and the Estonian IT Academy project "Sustainable Artificial Internet of Things" (SAIoT, grant No. TEMTA138). The authors thank Assistant Prof. Tomasz Kaminski, University of Warsaw, Poland, for providing the droplet image dataset for training the droplet classification model. The publication costs of this article were partially covered by the Estonian Academy of Sciences.

References

1. Trinh, T. N. D., Do, H. D. K., Nam, N. N., Dan, T. T., Trinh, K. T. L. and Lee, N. Y. Droplet-based microfluidics: applications in pharmaceuticals. *Pharmaceuticals*, 2023, **16**(7), 937. https://doi.org/10.3390/ph16070937
2. Chen, Z., Kheiri, S., Young, E. W. K. and Kumacheva, E. Trends in droplet microfluidics: from droplet generation to biomedical applications. *Langmuir*, 2022, **38**(20), 6233–6248. https://doi.org/10.1021/acs.langmuir.2c00491
3. Liu, Z., Jin, L., Chen, J., Fang, Q., Ablameyko, S., Yin, Z. et al. A survey on applications of deep learning in microscopy image analysis. *Comput. Biol. Med.*, 2021, **134**, 104523. https://doi.org/10.1016/j.compbiomed.2021.104523
4. Rutkowski, G. P., Azizov, I., Unmann, E., Dudek, M. and Grimes, B. A. Microfluidic droplet detection via region-based and single-pass convolutional neural networks with comparison to conventional image analysis methodologies. *Mach. Learn. Appl.*, 2022, **7**, 100222. https://doi.org/10.1016/j.mlwa.2021.100222
5. Afrin, F., Le Moullec, Y. and Pardy, T. Microfluidic droplet classification through tuned convolutional neural network on a resource constrained platform. In *2024 19th Biennial Baltic Electronics Conference (BEC)*, *Tallinn, Estonia, 2–4 October 2024*. IEEE, 2024, 1–4.
6. Zhou, X., Mao, Y., Gu, M. and Cheng, Z. WSCNet: biomedical image recognition for cell encapsulated microfluidic droplets. *Biosensors*, 2023, **13**(8), 821. https://doi.org/10.3390/bios1308 0821
7. Soldati, G., Del Ben, F., Brisotto, G., Biscontin, E., Bulfoni, M., Piruska, A. et al. Microfluidic droplets content classification and analysis through convolutional neural networks in a liquid biopsy workflow. *Am. J. Transl. Res.*, 2018, **10**(12), 4004–4016.
8. Kensert, A., Harrison, P. J. and Spjuth, O. Transfer learning with deep convolutional neural networks for classifying cellular morphological changes. *SLAS Discov.*, 2019, **24**(4), 466–475. https://doi.org/10.1177/2472555218818756
9. Lee, K., Kim, S.-E., Doh, J., Kim, K. and Chung, W. K. User-friendly image-activated microfluidic cell sorting technique using an optimized, fast deep learning algorithm. *Lab Chip*, 2021, **21**(9), 1798–1810. https://doi.org/10.1039/D0LC00747A
10. Xu, J., Fan, W., Madsen, J., Tanev, G. P. and Pezzarossa, L. AI-based detection of droplets and bubbles in digital microfluidic biochips. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium, 17–19 April 2023*. IEEE, 2023, 1–6. https://doi.org/10.23919/DATE56975.2023.10136887
11. Ghafari, M., Mailman, D., Hatami, P., Peyton, T., Yang, L. and Dang, W. A comparison of YOLO and Mask-RCNN for detecting cells from microfluidic images. In *2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Jeju Island, Korea, 21–24 February 2022*. IEEE, 2022, 204–209.
12. Li, Y., Mahjoubfar, A., Chen, C. L., Niazi, K. R., Pei, L. and Jalali, B. Deep cytometry: deep learning with real-time inference in cell sorting and flow cytometry. *Sci. Rep.*, 2019, **9**, 11088. https://doi.org/10.1038/s41598-019-47193-6
13. Suzuki, Y., Kobayashi, K., Wakisaka, Y. and Ozeki, Y. Label-free chemical imaging flow cytometry by high-speed multicolor stimulated Raman scattering. *Proc. Natl. Acad. Sci. U.S.A.*, 2019, **116**(32), 15842–15848. https://doi.org/10.1073/pnas.1902322116
14. Meng, N., Lam, E. Y., Tsia, K. K. and So, H. K.-H. Large-scale multi-class image-based cell classification with deep learning. *IEEE J. Biomed. Health Inform.*, 2019, **23**(5), 2091–2098. https://doi.org/10.1109/jbhi.2018.2878878
15. Riti, J., Sutra, G., Naas, T., Volland, H., Simon, S. and Perez-Toralla, K. Combining deep learning and droplet microfluidics for rapid and label-free antimicrobial susceptibility testing of colistin. *Biosens. Bioelectron.*, 2024, **257**, 116301. https://doi.org/10.1016/j.bios.2024.116301
16. Ahmadpour, A., Shojaeian, M. and Tasoglu, S. Deep learning-augmented T-junction droplet generation. *iScience*, 2024, **27**(4), 109326. https://doi.org/10.1016/j.isci.2024.109326
17. Asama, R., Liu, C. J. S., Tominaga, M., Cheng, Y.-R., Nakamura, Y., Kondo, A. et al. Droplet-based microfluidic platform for detecting agonistic peptides that are self-secreted by yeast expressing a G-protein-coupled receptor. *Microb. Cell Factories*, 2024, **23**, 104. http://dx.doi.org/10.1186/s12934-024-02379-0
18. Raspberry. *Raspberry Pi Hardware*. https://www.raspberrypi.com/documentation/computers/raspberry-pi.html (accessed 2024-12-13).
19. Khan, S. Z., Le Moullec, Y. and Alam, M. M. An NB-IoT-based edge-of-things framework for energy-efficient image transfer. *Sensors*, 2021, **21**(17), 5929. https://doi.org/10.3390/s21175929
20. Sipeed. *MaixCam – Fast Development for AI Vision and Audio Projects*. https://wiki.sipeed.com/hardware/en/maixcam/index.html (accessed 2024-12-13).
21. De Jonghe, J., Kaminski, T. S., Morse, D. B., Tabaka, M., Ellermann, A. L., Kohler, T. N. et al. spinDrop: a droplet microfluidic platform to maximise single-cell sequencing information content. *Nat. Commun.*, 2023, **14**, 4788. http://dx.doi.org/10.1101/2023.01.12.523500
22. Skalski, P. *makesense.ai*. https://github.com/SkalskiP/make-sense (accessed 2024-12-14).
23. Redmon, J. *Darknet Open Source Neural Network Framework*. https://github.com/pjreddie/darknet (accessed 2024-12-14).

24. Sipeed. *LicheeRV Nano*. https://wiki.sipeed.com/hardware/en/lichee/RV_Nano/1_intro.html (accessed 2024-12-13).
25. SOPHGO. *SG200X Hardware*. https://github.com/sophgo/sophgo-hardware/tree/master/SG200X (accessed 2024-12-13).
26. Sipeed. *MaixHub*. https://maixhub.com/ (accessed 2024-12-14).
27. Jõemaa, R., Gyimah, N., Ashraf, K., Pärnamets, K., Zaft, A. and Scheler, O. CogniFlow-Drop: integrated modular system for automated generation of droplets in microfluidic applications. *IEEE Access*, 2023, **11**, 104905–104929.

Konvolutsioonilistel närvivõrkudel (CNN) põhinev mikrofluidsete tilkade klassifitseerimine portatiivsetes voolutsütomeetrites

Fariha Afrin, Yannick Le Moullec, Tamas Pardy ja Toomas Rang

Tilkade klassifitseerimine on oluline aspekt pilditöötlust sisaldavate voolutsütomeetrite arendamisel. Süvaõppe algoritmid on võimelised avastama ja klassifitseerima tilku suuremõõtmelistes laboriseadmetes, kuid sarnase tehnoloogia rakendamine portatiivsetes seadmetes kujutab endast suurt väljakutset, kuna nende arvutusvõimsus ei vasta kompaktsete seadmete arvutusvõimekusele. See on oluline takistus üleminekul statsionaarsetelt laboriseadmetelt välioludes kasutatavatele portatiivsetele lahendustele. Takistuse ületamiseks tutvustame artiklis kohandatud YoloV4-tiny mudelit, mida on rakendatud Raspberry Pi-5 (RPi5) platvormil.

Närvivõrgupõhist lahendust treeniti 878 erineva kujutise abil, mis pärinesid 975 kujutisega kohandatud andmehulgast. See andmehulk koguti meie loodud reaalse eksperimentaalse mikrofluidikaseadmega. Tulemusi hindasime interferentsiaja ja keskmise täpsuse (mAP – *mean average precision*) alusel. Loodud lahendus suutis edukalt klassifitseerida kolme selgelt eristatavat olukorda (tilk puudub, üks tilk, mitu tilka) 13 ms jooksul, saavutades keskmise täpsuse 99,95% lävendiga 0.5 (mAP@0,5). Samuti võrdlesime kohandatud YoloV4-tiny mudelit seitsme masinõppemudeli (ML) ja platvormi kombinatsiooniga, sealhulgas uusima, kompaktse ja soodsa tensoritöötlusega tippseadmega (MaixCam plaat koos LicheeRV Nano mooduli / SOPHGO SG2002-ga), mis kasutab YoloV5 algoritmi. YOLOv4-tiny RPi5 lahendust võrdlesime YOLOv5-s mudeli ja MaixCam platvormi kombinatsiooniga. Tänu lisatud tensoritöötluse algoritmile lühenes klassifitseerimise aeg 5,34 ms-ni, saavutades keskmise täpsuse 55,09% juhtudest lävendiga 0,5 (mAP@0,5). Täpsuse protsentuaalne vähenemine on tingitud kvantimisest. Uuring näitab, et süsteemi täpse disainimise abil on võimalik saavutada tasakaal täpsuse ja kiiruse vahel, võimaldades mikrofluidsete tilkade usaldusväärset klassifitseerimist ka piiratud arvutusvõimekusega portatiivsetes voolutsütomeetrites.