Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue

Check for updates

Cite this: *Digital Discovery*, 2026, **5**, 254

Real-time cell sorting with scalable *in situ* FPGA-accelerated deep learning

Khayrul Islam, [a] Ryan F. Forelli,[b] Jianzhong Han,[c] Deven Bhadane,[d] Jian Huang,[cef] Joshua C. Agar, [g] Nhan Tran,[h] Seda Ogrenci[b] and Yaling Liu *[ijk]

Precise cell classification is essential in biomedical diagnostics and therapeutic monitoring, particularly for identifying diverse cell types involved in various diseases. Traditional cell classification methods, such as flow cytometry, depend on molecular labeling, which is often costly, time-intensive, and can alter cell integrity. Real-time microfluidic sorters also impose a sub-ms decision window that existing machine-learning pipelines cannot meet. To overcome these limitations, we present a label-free machine learning framework for cell classification, designed for real-time sorting applications using bright-field microscopy images. This approach leverages a teacher–student model architecture enhanced by knowledge distillation, achieving high efficiency and scalability across different cell types. Demonstrated through a use case of classifying lymphocyte subsets, our framework accurately classifies T4, T8, and B cell types with a dataset of 80 000 pre-processed images, released publicly as the LymphoMNIST package for reproducible benchmarking. Our teacher model attained 98% accuracy in differentiating T4 cells from B cells and 93% accuracy in zero-shot classification between T8 and B cells. Remarkably, our student model operates with only 5682 parameters (~0.02% of the teacher, a 5000-fold reduction), enabling field-programmable gate array (FPGA) deployment. Implemented directly on the frame-grabber FPGA as the first demonstration of *in situ* deep learning in this setting, the student model achieves an ultra-low inference latency of just 14.5 μs and a complete cell detection-to-sorting trigger time of 24.7 μs, delivering 12× and 40× improvements over the previous state of the art in inference and total latency, respectively, while preserving accuracy comparable to the teacher model. This framework establishes the first sub-25 μs ML benchmark for label-free cytometry and provides an open, cost-effective blueprint for upgrading existing imaging sorters.

Received 6th August 2025
Accepted 17th October 2025

DOI: 10.1039/d5dd00345h

rsc.li/digitaldiscovery

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM.
This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

1 Introduction

Accurate cell classification is critical for a wide range of biomedical applications, including disease diagnostics, immunological studies, and personalized therapies. Traditional methods for cell classification, such as molecular labeling through flow cytometry, rely on detecting specific surface markers.[1] While these techniques are accurate, they have notable limitations, including high costs, time-intensive protocols, and potential interference with the natural state of the cells.[2] Equally important, modern acoustofluidic sorters provide only a ~1 ms window between image acquisition and actuation, a latency budget that no published machine-learning (ML) pipeline has yet satisfied.[3] In response to these challenges, label-free classification methods have emerged as a promising alternative by leveraging intrinsic cell properties, such as morphology and biomechanics. Recent work has demonstrated the fundamental interconnection between biophysical cues and cellular morphology, with substrate geometry alone capable of reverting pluripotent stem cells to naivety through morphological changes.[4] This highlights that morphological features captured in bright-field images encode meaningful information about cellular state and phenotype. These approaches preserve the natural state of the cells, enabling downstream applications such as transplantation, functional studies, and real-time analysis or sorting.[5,6]

[a]*Department of Mechanical Engineering, Lehigh University, Bethlehem, PA 18015, USA*
[b]*Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208, USA*
[c]*Coriell Institute for Medical Research, Camden, NJ, USA*
[d]*Department of Computer Science, Lehigh University, Bethlehem, PA 18015, USA*
[e]*Cooper Medical School of Rowan University, Camden, NJ 08103, USA*
[f]*Center for Metabolic Disease Research, Temple University Lewis Katz School of Medicine, Philadelphia, PA 19122, USA*
[g]*Department of Mechanical Engineering and Mechanics, Drexel University, Philadelphia, PA 19104, USA*
[h]*Real-time Processing Systems Division, Fermi National Accelerator Laboratory, Batavia, IL 60510, USA*
[i]*Precision Medicine Translational Research Center, West China Hospital, Sichuan University, Chengdu, Sichuan, 610041, China. E-mail: yaling.liu@gmail.com*
[j]*Center for High Altitude Medicine, West China Hospital, Sichuan University, Chengdu, Sichuan, 610041, China*
[k]*Department of Bioengineering, Lehigh University, Bethlehem, PA 18015, USA*

© 2026 The Author(s). Published by the Royal Society of Chemistry

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM. This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

Recent advancements in ML have revolutionized cell classification by offering innovative solutions to circumvent the limitations of traditional methods. For instance, deep CNNs have been successfully applied to bright-field images for label-free identification of cell types, with feature fusion approaches integrating morphological patterns across multiple convolutional modules to achieve high accuracy.[7] While such specialized approaches show promise, many general ML models perform suboptimally on datasets other than those they were specifically trained on, revealing inadequate generalization and transfer-learning capabilities. Furthermore, training protocols optimized for general image datasets often fail to translate effectively to biological datasets.[8,9] Progress is also slowed by the scarcity of large, publicly available bright-field datasets with ground-truth phenotypes, making reproducible benchmarking difficult.

Addressing these challenges requires robust training methodologies tailored specifically for diverse biological image datasets. In this study, we focus on optimized training protocols that achieve high specificity and sensitivity in cell classification. Using lymphocyte classification as a use case, we demonstrate the adaptability and effectiveness of these training recipes, highlighting their potential to extend seamlessly to various cell types and enabling versatile applications across different biological contexts. Specialized expertise in lymphocyte classification remains limited even in well-resourced communities, leading to variability in diagnostic accuracy. This issue is exacerbated in underserved areas, where the lack of access to expert pathology services results in prolonged or erroneous diagnostic outcomes that critically impair patient management. Our ML framework leverages bright-field images to detect cellular morphological features for the cell classification process. By eliminating reliance on molecular labels, this approach reduces human subjectivity, ensures reproducibility, and offers consistent results across different settings. To facilitate community adoption, we release both our training code and the 80 000-image LymphoMNIST dataset as pip-installable packages.

Moreover, to meet the demands of real-time inference, we have implemented a field-programmable gate array (FPGA) version of our optimized student model, achieving ultra-low latency and high throughput. Previous studies have demonstrated cell classification ML inference performance on the order of milliseconds, primarily on GPU and CPU hardware.[10–12] The previous state-of-the-art (SOTA) in terms of inference latency implements a deep neural network (DNN) for standing surface acoustic wave cell sorting and achieves an inference latency of approximately 183 s^2 and a full cell detection-to-sorting trigger latency of <1 ms. Leveraging high-level-synthesis tools (hls4ml) and a knowledge-distilled student network with only 5682 parameters (about 0.02% of the 28 M-parameter teacher, a 5000-fold reduction), we achieve the first frame-grabber-resident deep-learning implementation that fits within this strict latency envelope.

By leveraging our ML framework in a use case involving the classification of T4, T8, and B cells, we have achieved remarkable accuracy improvements. Our teacher model demonstrates approximately 98% accuracy in classifying T4 cells from B cells and achieves about 93% accuracy in zero-shot classification of T8 *vs.* B cells. Employing knowledge-distillation (KD) techniques, our student 2 model attains sufficiently high accuracy relative to the teacher model with just about 0.02% of its parameters. The FPGA implementation of the student model further enhances processing speed, reducing inference latency to just 14.5 s. This improvement in processing speed facilitates the real-time analysis and accurate sorting of T and B cells, significantly advancing their rapid classification in clinical settings. With these insights and results in place, the core achievements and contributions of our study are summarized in the following research highlights:

(1) Dataset: we present a dataset of 80 000 images, which supports the training and validation of our models. The data are freely available *via* the pip-installable LymphoMNIST package for immediate benchmarking.

(2) Models: we publish detailed recipes for a high-capacity teacher and a KD-trained student with an in-house, light-weight architecture tuned for bright-field cell images, achieving 5000-fold parameter compression (5682 params, 0.02% of the teacher) while retaining F1 > 0.97.

(3) Transfer learning: we demonstrate the transfer-learning capability for T8 *versus* B cell classifications, indicating that the model can perform zero-shot inference and can be further tuned to detect other lymphocyte cell types.

(4) *In situ* FPGA Implementation: we deploy our student model directly on the frame-grabber FPGA, eliminating PCIe transfer overhead and reducing inference latency from the 183 s previous SOTA and 325 s on GPU to just 14.5 s, a 12× and 22× improvement, respectively. Thus, we institute a new SOTA real-time deep-learning benchmark and implementation for real-time cell sorting and rapid classification.

2 Results and discussion

2.1 Composition of training and validation sets

The LymphoMNIST dataset consists of 80 000 high-resolution lymphocyte images, each with a resolution of 64 × 64 pixels (Fig. 1(a)). These images are categorized into three primary classes: B cells, T4 cells, and T8 cells (Fig. 1(b and c)). To support the development and evaluation of machine learning models, the dataset is partitioned into training, validation, and testing sets in an 80-10-10 split, resulting in 64 000 images for training and 8000 images each for testing and validation (Fig. 1(e)). To enhance accessibility and usability, we have developed a pip-installable package that allows researchers to seamlessly download the dataset and incorporate it into their experimental workflows.[13] The images in the dataset were captured under diverse environmental conditions, including variations in lighting and camera settings, to introduce a realistic level of complexity for algorithm development. These conditions are designed to simulate the variability encountered in real-world scenarios, challenging models to generalize effectively. Furthermore, the dataset includes images from both young (65%) and aged (35%) mice to account for age-specific cellular variability, a factor that enhances the model's ability to

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM.
This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

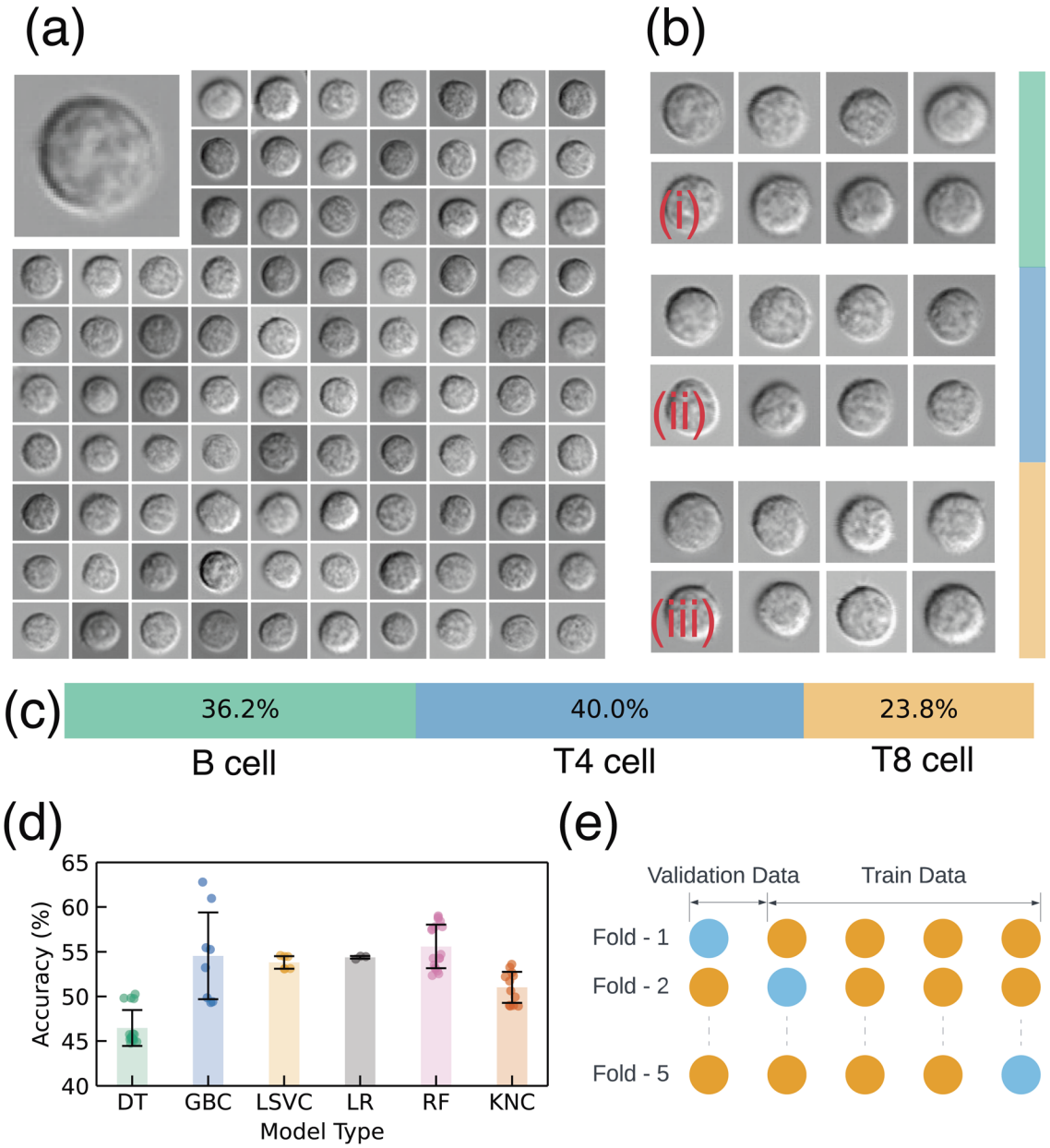

Fig. 1 Overview of LymphoMNIST dataset and ablation study. (a) Sample dataset images, (b) examples of B(I), T4(II), and(III) T8 cells, (c) cell type distribution, (d) classification method performance in ablation study, and (e) training-validation split visualization.

generalize across different biological conditions. The collection spanned 18 months across four seasons, ensuring that environmental fluctuations such as controlled humidity (±5%) and temperature (±2 °C) were captured, further contributing to dataset diversity. Performance benchmarks of various models, like Decision Tree (DT), Gradient Boosting Classifier (GBC), Linear Support Vector Classifier (LSVC), Logistic Regression (LR), Random Forest (RF), and K-Nearest Neighbors Classifier (KNC), applied to the dataset are detailed in the SI. Accuracy metrics for these models are presented in Fig. 1(d), providing insights into the dataset's applicability for machine learning tasks.

2.2 Detection of cell class by teacher

In this study, we utilized the ResNet50 architecture as our Teacher Network (TN) for the classification of B cells and T4 cells using bright-field microscopy images. ResNet50 is a deep convolutional neural network with residual connections, designed to alleviate the vanishing gradient problem and enable deeper feature extraction. Its capability to learn hierarchical representations makes it well-suited for complex image classification tasks such as distinguishing cell types.[14]

We observed that increasing the image size from the original 64 × 64 pixels in the LymphoMNIST dataset to 120 × 120 pixels improved both training and validation accuracy. This larger size allowed TN to capture more spatial information, enhancing

© 2026 The Author(s). Published by the Royal Society of Chemistry

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM. This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

feature extraction. The choice of image size is closely tied to the depth of the architecture, as deeper models like ResNet50 can leverage larger feature maps for improved performance, as noted in previous research.[15,16] However, further increasing the size led to overfitting due to the model's increased complexity. Thus, we standardized all images to 120 × 120 pixels to achieve an optimal balance between feature learning and generalization (Table 1).

To improve generalization and reduce overfitting, we employed a range of data augmentation techniques, including random flips, rotations, scaling, translations, shearing, contrast adjustments, hue and saturation adjustments, and Gaussian blur. The choice and intensity of these augmentations must be carefully balanced depending on both the complexity of the model and the amount of available data. For complex models like ResNet50, stronger augmentations can introduce sufficient variability, preventing the model from overfitting by helping it generalize better across the dataset.[17] However, when the dataset is limited, applying overly strong augmentations can introduce excessive noise, which may degrade performance, particularly in tasks with high-dimensional latent spaces (Fig. 2(a)) by causing the model to fit irrelevant or spurious patterns.[18] In such cases, it can be more effective to use a less complex model that is better suited to the smaller dataset, as it reduces the risk of overfitting to noise and irrelevant patterns in the training data.[19] The dataset exhibited a class imbalance between B cells and T4 cells. To address this, we employed a weighted random sampler during training to ensure that the underrepresented classes were adequately sampled. This approach allowed the model to learn distinguishing features for both classes effectively, preventing bias towards the majority class (Fig. 2(c)).

The TN model achieved a training accuracy of approximately 97%, and a validation accuracy of approximately 98% after 70 epochs (Fig. 2(a)). The close alignment between the training and validation accuracies indicates strong generalization without significant overfitting. Notably, the validation accuracy occasionally surpassed the training accuracy, likely due to the extensive augmentations applied to the training data, which were not applied to the validation set. Fig. 2(b) shows the Receiver Operating Characteristic (ROC) curve, which highlights the model's strong discriminatory capability between B cells and T4 cells, with a high Area Under the Curve (AUC) for both the training and validation datasets. The confusion matrix in Fig. 2(c) demonstrates high true positive rates and low false positive rates for both classes. Finally, the t-distributed Stochastic Neighbor Embedding (t-SNE) visualization (Fig. 2(d)) provides a visual representation of the separation between B cells and T4 cells in the latent feature space. The minimal overlap between clusters further confirms the model's ability to effectively capture distinguishing features between the two cell types, making it a reliable tool for cell classification in biomedical applications.

2.3 Detection of cell class by student

In this section, we investigate the effectiveness of KD in training student models by transferring knowledge from a pre-trained teacher model. Adopting the principles from Beyer *et al.*,[22] we employed a "consistent and patient" teaching strategy, emphasizing the importance of long training schedules and uniform input views between teacher and student. The distillation process allows the student model to leverage the richer representations of the teacher, improving its predictive capabilities. In this study, we trained two distinct student models, referred to as student 1 and student 2. student 1 utilizes ResNet-18, a moderately complex convolutional neural network (CNN) with approximately 11.2 million parameters and an input size of 64 × 64 pixels. We also developed a significantly compact model, student 2, which is a lightweight CNN optimized for resource-constrained devices with only 5682 parameters and a smaller input size of 48 × 48. Notably, student 2 achieved approximately 90% accuracy in the classification task, demonstrating high efficiency with just 0.02% of the parameters used by the teacher model, which achieved ∼98% accuracy.

Our experiments reconfirmed that data-mixing augmentation techniques, such as CutMix and MixUp, substantially enhance KD performance. Conversely, other image-based augmentations, including random flipping and shearing, degraded the accuracy of the distilled student model when applied inconsistently between teacher and student,[23] as demonstrated by Beyer *et al.*[22] Maintaining identical image crops and augmentation strategies for both teacher and student networks during training was crucial to ensure consistent learning and effective knowledge transfer without misalignment in data representation.[22]

We observed that the student 2 model attained significantly higher accuracy when trained using KD compared to training from scratch. This outcome aligns with prior research indicating that KD enables smaller models to focus on relevant information by utilizing outputs from a larger teacher model, including softened labels, as guidance.[24] Such guidance allows the student model to capture complex patterns by receiving nuanced data representations, which may be challenging to learn independently, especially in resource-constrained scenarios.[25] Furthermore, studies have demonstrated that KD

Table 1 Comparison of model performance with published studies

Study	Imaging technique	Model	Metric	Score
Turan *et al.*[20]	Fluorescence	AlexCAN	Accuracy	98%
Nassar *et al.*[21]	Bright-field	Gradient boosting	F1 score	78%
This study	Bright-field	Teacher	Accuracy	98%
			F1 score	97.05%

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM.
This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

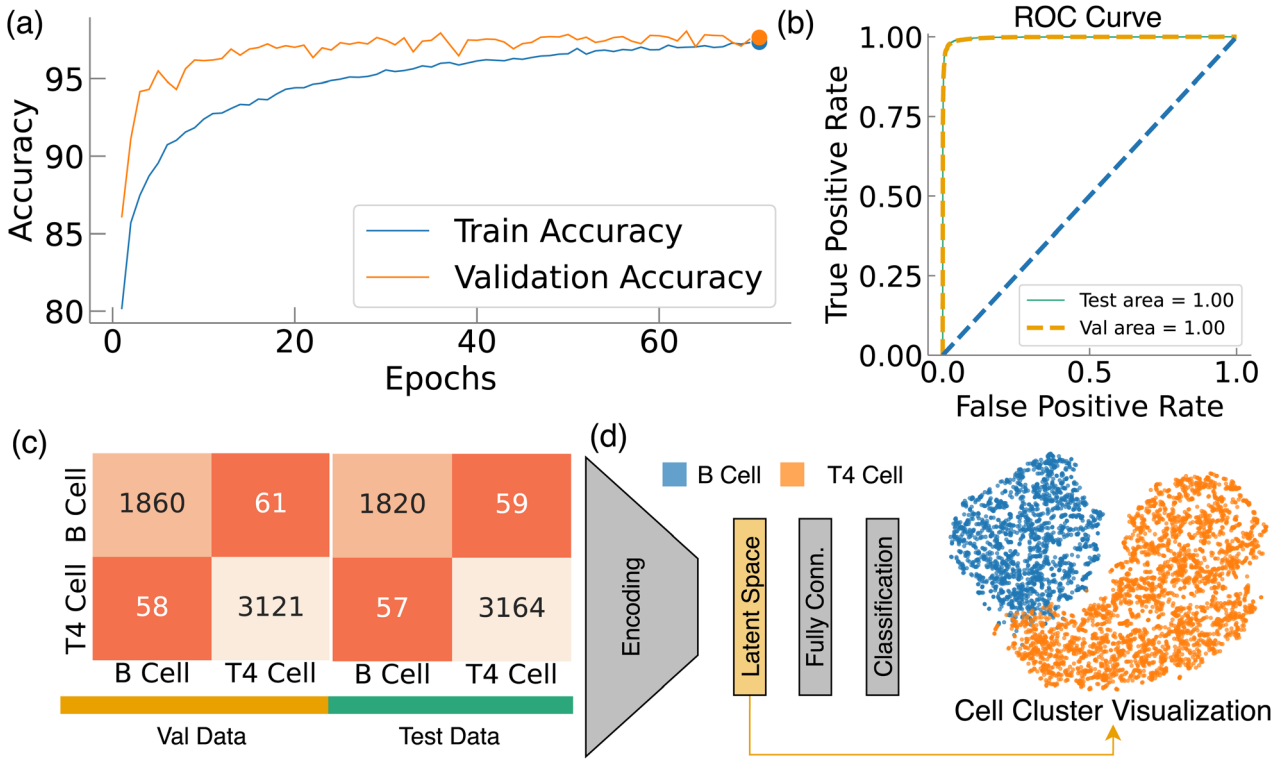

Fig. 2 Evaluation of the Teacher Network (TN). (a) Accuracy during training and testing phases, (b) ROC curve, (c) confusion matrix demonstrating model efficacy on training and validation datasets using the TN, (d) depiction of the TN framework and t-Distributed Stochastic Neighbor Embedding (t-SNE) visualization derived from the latent space.

improves the ability of student models to capture high-level abstractions that are difficult to learn without teacher supervision.[26] For instance, Hinton *et al.*[27] showed that soft targets enhance student model performance by conveying richer information about class relationships.

The performance evaluation of student networks, shown in Fig. 3, reveals their accuracy on training and validation datasets. Confusion matrices on Fig. 3(a) and (b) indicate that student 1 slightly outperforms student 2, although student 2 demonstrates strong generalization capabilities in more challenging classes, suggesting that KD effectively maintains robustness in smaller models.[28] Fig. 3(c) presents a t-SNE visualization for student 1, showing distinct clusters that signify successful feature extraction and class differentiation. ROC curves (Fig. 3(d)) for both models illustrate high discriminative performance, with AUC values of 98% for student 1 and 96% for student 2 respectively. Comparative analysis of model parameters and latency in Fig. 3(e) and (f) reveals that student 2 operates with only 0.02% of the teacher model's parameters, achieving a latency of $\sim 0.325 \pm 0.004$ ms. This is significantly lower than student 1 ($\sim 2.11 \pm 0.03$ ms) and the teacher model ($\sim 5.05 \pm 0.06$ ms), with the FPGA implementation further reducing latency to $\sim 0.0145 \pm 0.001$ ms.

2.4 Transfer learning for T4 and T8 cell classification

This section investigates the utilization of transfer learning to differentiate between T8 and B cells employing a pre-trained teacher model. Originally trained on the LymphoMNIST dataset, the teacher network exhibited substantial feature extraction capabilities, achieving ~98% accuracy on both validation and test datasets. In a zero-shot learning framework (Transfer0 in Fig. 4) for classifying T8 *versus* B cells, the model demonstrated an initial accuracy of ~93%. To improve classification performance, the teacher model underwent fine-tuning on a subset of the dataset specifically annotated for T8 and B cells. This fine-tuning involved modifying the training regimen to include only eight epochs, which facilitated model convergence without inducing overfitting. Post fine-tuning, the model reached an improved accuracy of ~97%, which surpassed its zero-shot learning performance.

To further assess the generalizability of the transfer learning approach beyond the specific T8 *vs.* B cell classification task, we evaluated our model on an external dataset,[29] which includes additional hematological cell types. Our results demonstrated a ~1% accuracy boost for T *vs.* Leukemia cell classification when using our pretrained teacher model as the starting point, compared to an ImageNet-pretrained ResNet50. This indicates that leveraging prior domain-specific knowledge enhances model adaptability across different cell types and pathological conditions, reinforcing the robustness of our transfer learning strategy.

Fig. 4 illustrates the model's performance through comparative assessments of accuracy, precision, recall, and F1 score across panels (a) to (d). The adaptability of the model to the new classification task, with minimal risk of overfitting and improved

© 2026 The Author(s). Published by the Royal Society of Chemistry

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM.
This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

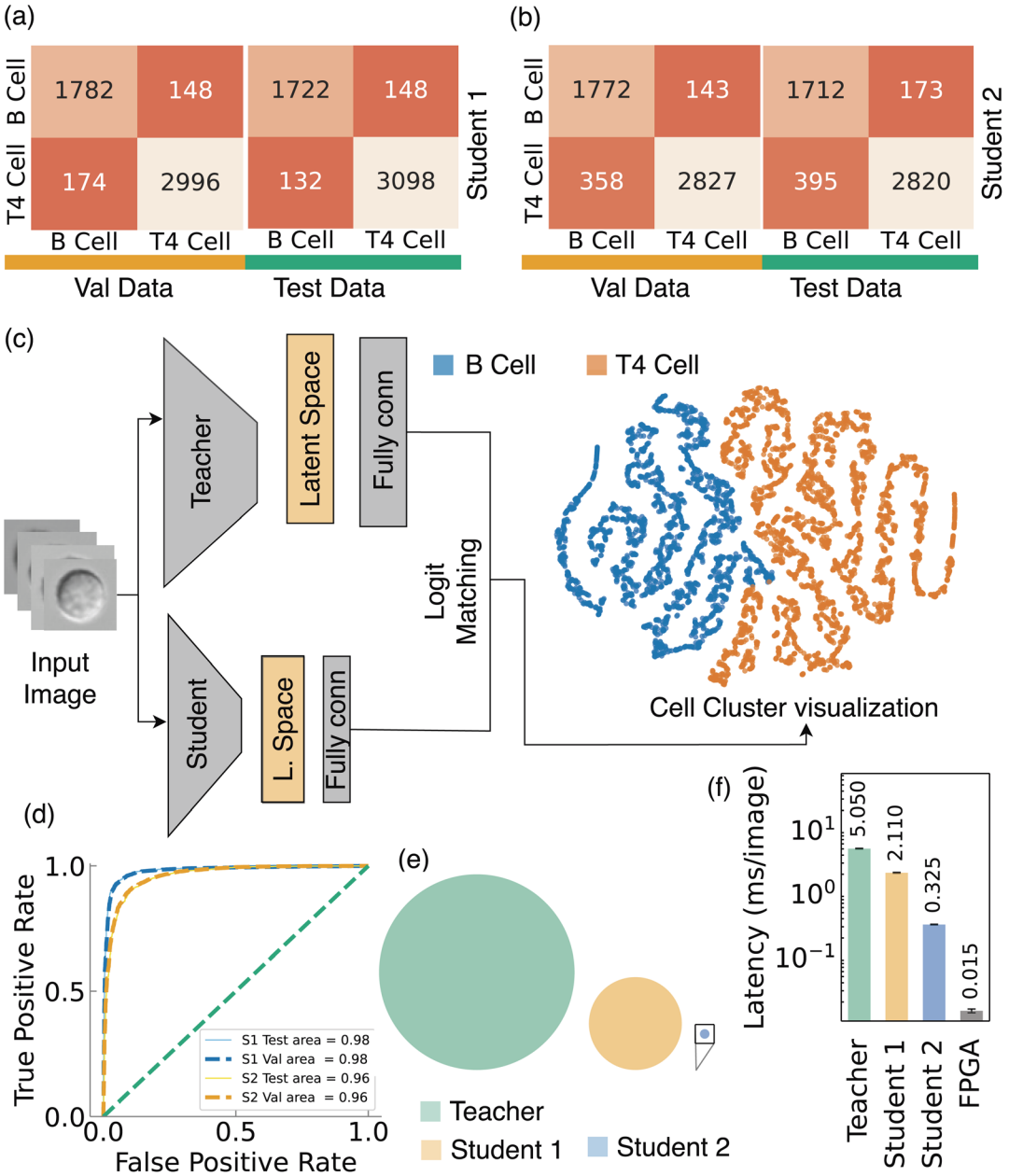

Fig. 3 Performance evaluation of the Student Network (SN). (a) Confusion matrix for student 1, (b) confusion matrix for student 2, (c) t-SNE visualization of the SN framework, (d) ROC curve, (e) comparative analysis of model parameters (student 2 magnified 200x), (f) latency comparison between teacher and student networks.

generalization capabilities, highlights the practical application of transfer learning in biomedical image analysis. Future research directions include extending these methodologies to other cell types or imaging modalities and combining them with continuous learning strategies or domain adaptation to enhance model performance under diverse imaging conditions.

2.5 FPGA implementation of the student model

For real-time cell sorting applications, latency is more critical than throughput because a decision must be made quickly within the short period that each cell spends passing under the camera's region of interest after detection and before passing through the acoustic sorting region. GPUs are specifically designed for high throughput processing as they have high-bandwidth memory and can handle massive data flow. However, they falter with latency-sensitive tasks as they are not optimized for single-threaded performance. In our testing, student 2 achieves an average inference latency of 0.325 ms and can reach a throughput of 3.1 kfps with a batch size of 1 on our NVIDIA A100 GPU.

To achieve the latencies required for real-time control in cell sorting, an alternative platform is required. FPGAs are devices characterized by their flexibility and parallelism and provide

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM.
This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

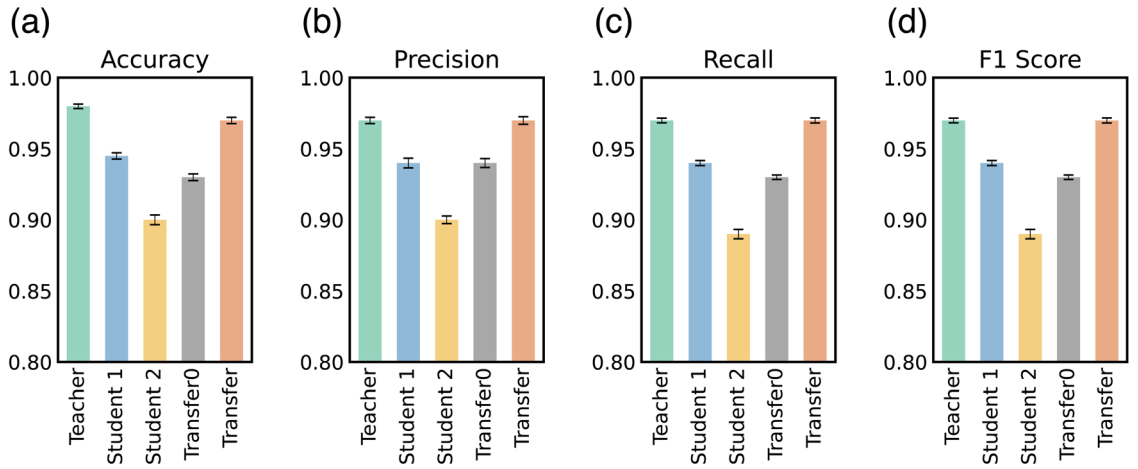

Fig. 4 Model performance evaluation. (a)–(d) present the comparative assessment of accuracy, precision, recall, and F1 score.

a suitable balance between throughput and latency for real-time applications. They primarily consist of an array of reconfigurable hardware blocks, such as multipliers, logic blocks, and memories that can be used to implement an algorithm directly as a circuit, thereby forgoing the stack of software and drivers required for a GPU or CPU implementation. Additionally, the emergence of HLS technologies, enabling the synthesis of standard C++ code to register-transfer level hardware descriptions, means that deploying algorithms to custom hardware is easier than ever.

Furthermore, tools like hls4ml facilitate the process of deploying neural networks to FPGA hardware and have been shown capable of achieving nanosecond-level latencies for machine learning inference.[32] hls4ml enables the translation of most neural network architectures written in a high-level deep learning framework such as PyTorch or Keras/Tensorflow to an HLS representation using dictionary configuration files and prewritten layer templates for all common HLS synthesis tools including Xilinx, Intel, and Siemens.[33–35]

hls4ml provides multiple avenues of optimization that empowers us to meet this project's latency constraints. First and foremost, previous work has demonstrated that neural network parameters can be quantized to a lower bit width with minimal impact on overall accuracy.[36] This finding is critical for enabling the deployment of neural networks on resource-constrained devices. In this implementation, we use hls4ml to quantize the student 2 network with layer-level granularity while still achieving 86% accuracy. We also leverage hls4ml's "reuse factor" hyperparameter to fine-tune the level of parallelization applied to each layer of the network. The value of this parameter indicates the maximum number of operations that can share a given physical instance of a resource. This feature allows us to achieve the ultra-low latencies required for this application while remaining within the resource constraints of the FPGA device. The effects of varying this hyperparameter can be illustrated as a Pareto frontier where a high reuse factor results in low resource usage but high latency, and a low reuse factor results in high resource usage but low latency.[37] In general, we find that implementing dense layers with a higher reuse factor of 25, and the two convolutional layers with lower reuse factors of 1 and 2, respectively, yields an optimal balance between latency and resource usage.

Apart from latency, another challenge to enabling real-time control presents itself in the substantial input/output (IO) overhead that we would incur when utilizing a CPU or any external PCIe GPU or FPGA accelerator. Therefore, we endeavored to place our student 2 model computation as close to the edge as possible in our experiment to minimize this overhead. Our experimental setup consists of a Phantom S710 high-speed streaming camera aimed at the microfluidic channel through the microscope camera port, paired with the Euresys frame grabber PCIe card. This frame grabber card is responsible for reading out and processing the raw camera sensor data before transmitting frames back to the host computer. Frame grabbers typically implement this processing on an onboard FPGA chip. Conveniently, Euresys offers a tool, CustomLogic, that enables users to deploy custom image processing to their frame grabber FPGA.[38] A separate framework, Machine Learning for Frame Grabbers (ml4fg) has also been developed specifically to bridge the gap between CustomLogic and hls4ml and enables seamless deployment of neural network models to Euresys frame grabbers.[39] Thus, we leverage all three of these existing tools to deploy student 2 directly *in situ* in the data readout path of the frame grabber, thereby circumnavigating the need for off-chip compute and completely eliminating all associated IO overhead while achieving ultra-low latency inference. Our full workflow from Python model to bitstream deployment is illustrated in Fig. 5.

We then empirically benchmark the latency of the FPGA implementation of student 2 by monitoring the internal communication protocol used by the neural network intellectual property (IP). We then utilize the frame grabber's TTL IO to output a square wave where the high time denotes inference latency which we measure with an oscilloscope. Fig. 6(a) exhibits the results of this latency test, showing a model inference latency of just 14.5 μs. Additionally, we observe that inference begins approximately 10.0 μs after the trigger edge.

© 2026 The Author(s). Published by the Royal Society of Chemistry

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM.
This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

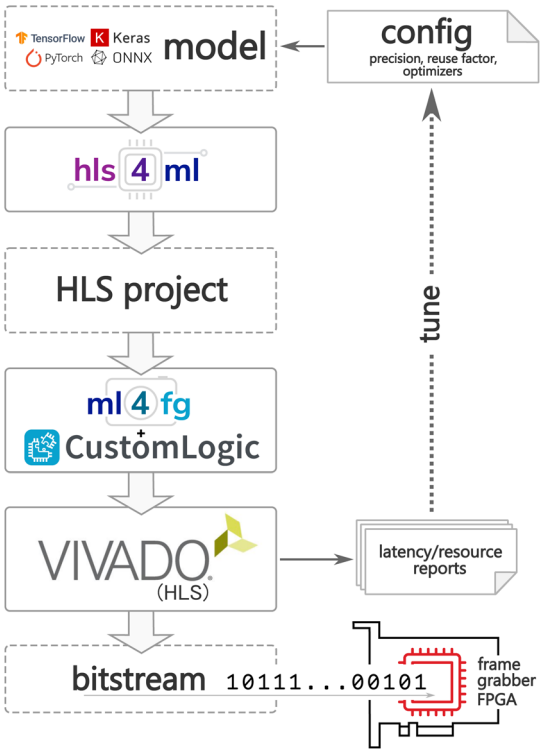

Fig. 5 Workflow from a high-level Python framework to HLS conversion, synthesis, frame grabber integration, and final bitstream generation of deep learning models with hls4ml for frame grabber deployment. The final generated bitstream contains the configuration information that the FPGA uses to implement our deep learning model.

Given a 2 μs exposure time, our model completes inferencing approximately 22.5 μs after image exposure is finished. The model output writeout procedure takes an additional 0.2 μs. The writeout consists of the model's two-bit output indicating the cell output class, and can be expanded or adapted for any cell classification task or communication protocol. Aggregating these constituent components yields a full cell detection-to-sorting trigger time of 24.7 μs. By reducing inference latency to under 25 μs, our pipeline shifts the limiting factor from computation to fluidics. This margin not only exceeds the ~1 ms actuation window of current acoustofluidic sorters,[3] but also opens the door to applications previously inaccessible to image-based ML—such as sorting extracellular vesicles or bacteria, where transit times are an order of magnitude shorter than for mammalian cells. As shown in Fig. 6(b), we pipeline neural network inference with the exposure and readout processes to accelerate the algorithm to a throughput of 81 kfps in our implementation. This benchmark far exceeds our GPU's best performance at a batch size of 1. Note that in Fig. 6(a) we capture at 50 kfps such that consecutive inference traces do not overlap for readability purposes.

As shown in Fig. 6(c), our implementation of student 2 consumes the majority of the FPGA resources. DSPs, the resource primarily used to implement neural network multiply accumulate operations, are most heavily utilized because we parallelized the network to the limit of the chip's resource capacity with hls4ml's reuse factor hyperparameter. The high-speed camera's communication protocol IP imposes an additional resource tax, totaling about 95% DSP usage for the full design. A more granular breakdown of the neural network resource consumption is shown in Fig. 6(d). Most notably, the second convolutional layer consumes far more resources than any other layer due to the higher number of input channels, which results in more multiply-accumulates. Both convolutional layers consume the most lookup tables as they require more complex control logic to manage the sliding kernel window and to direct data between buffers.

By optimizing our student 2 model and leveraging existing tools like hls4ml for deployment *in situ* on low-cost off-the-shelf frame grabber FPGAs, we are able to bypass data transfer overhead and accelerate our deep learning algorithm to achieve a new SOTA 14.5 μs inference latency and 24.7 μs full cell detection-to-sorting trigger time for cell classification in real-time sorting applications (see Table 2).

3 Methods

3.1 Animals

Evi1-IRES-GFP knock-in (Evi1^{GFP}) mice, kindly provided by Dr Mineo Kurokawa at the University of Tokyo, were used for this study. The mice were bred and housed under specific-pathogen-free (SPF) conditions within the animal facility at Cooper University Health Care. All animal handling and experimental protocols adhered strictly to NIH-mandated guidelines for laboratory animal welfare. Protocols were reviewed and approved by the Institutional Animal Care and Use Committee (IACUC) at Cooper University Health Care to ensure compliance with ethical standards for the care and use of laboratory animals.

3.2 Antibodies

For flow cytometric analysis and cell sorting, the following fluorochrome-conjugated antibodies were used: CD3ε-FITC (BioLegend, cat# 152304), CD4-BV421 (BioLegend, cat# 100543), CD8a-PE/Cy7 (BioLegend, cat# 100722), CD19-PE/Cy5 (eBioscience, cat# 15-0193-82), and B220 (eBioscience, cat# 56-0452-82). These antibodies were selected for their specificity in targeting key immune cell surface markers, enabling accurate discrimination of immune cell subpopulations through fluorescence-based gating strategies.

3.3 Flow cytometric analysis and cell sorting

Murine lymphocytes were isolated from spleen tissue. Spleens were carefully dissected and homogenized to produce single-cell suspensions, followed by red blood cell lysis to ensure a clear lymphocyte population. After washing with Dulbecco's Phosphate-Buffered Saline (DPBS), cells were stained with the selected fluorochrome-conjugated antibodies at 4 °C for 15–30 minutes to ensure optimal labeling conditions. Flow cytometric analysis and fluorescence-activated cell sorting (FACS) were performed using a Sony SH800Z automated cell sorter or a BD

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM.
This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

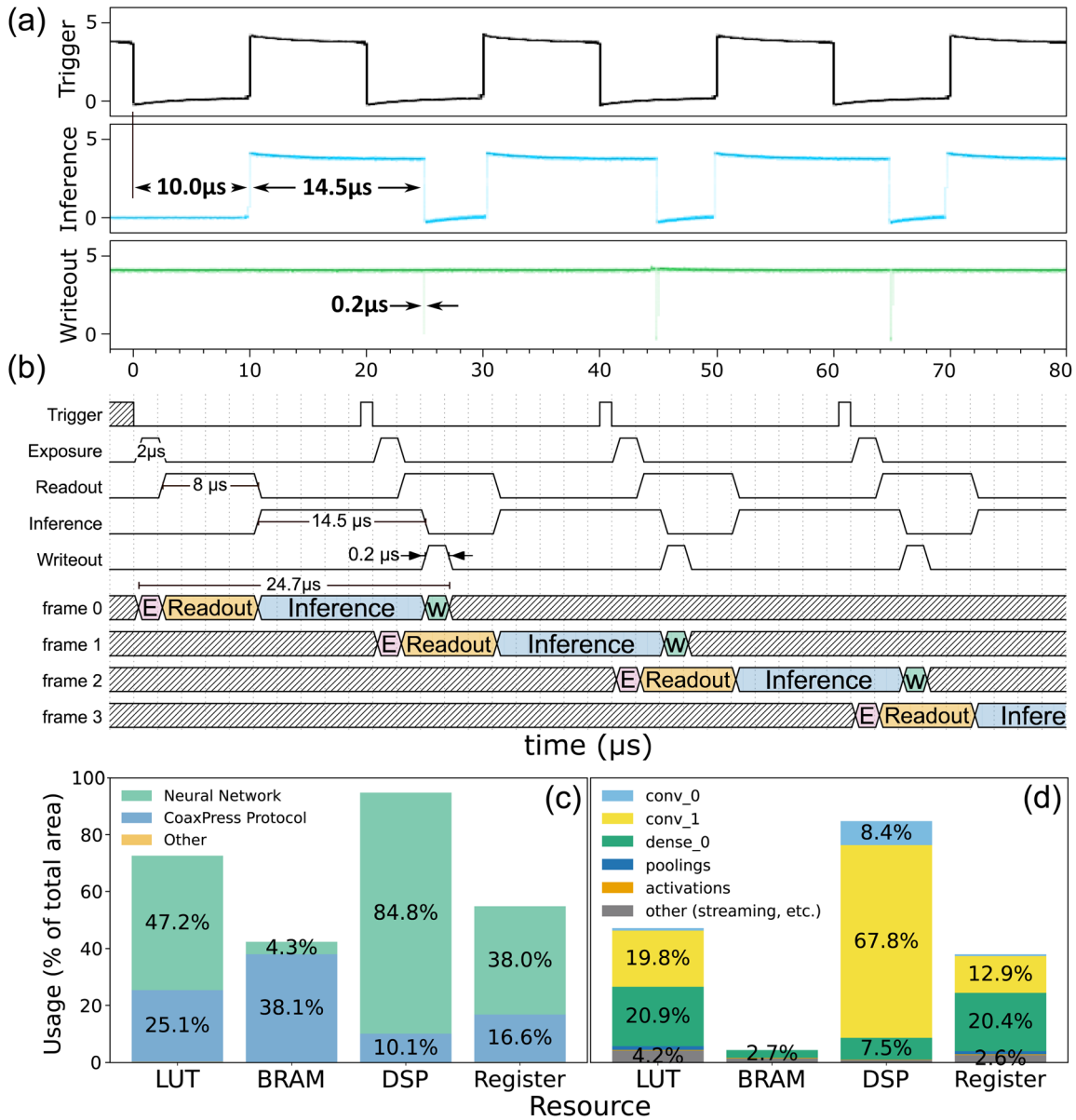

Fig. 6 Latency and resource performance of our FPGA implementation of student 2. (a) student 2 empirical oscilloscope benchmark of inference latency where "E" denotes camera exposure and "W" denotes the serial writeout, (b) student 2 frame grabber inference timing diagram illustrating pipelined model inference with exposure and readout, (c) overall resource consumption of the FPGA broken down by IP, (d) resource consumption of the neural network IP broken down by layer.

FACSAria™ III cell sorter. Negative controls were prepared with unstained cells to set appropriate gating thresholds. Data analysis was conducted using FlowJo software (v10) or the native software associated with the Sony cell sorter, employing stringent gating strategies to accurately identify and isolate specific immune cell subsets while excluding debris and non-viable cells.

3.4 DIC image acquisition

Following FACS, sorted cells were seeded into coverglass-bottomed chambers (Cellvis) and maintained in DPBS supplemented with 2% fetal bovine serum (FBS) to preserve cell viability throughout the imaging process. Differential interference contrast (DIC) imaging was performed on an Olympus FV3000 confocal microscope, with images captured at a resolution of 2048 × 2048 pixels. High-resolution DIC images allowed for precise morphological characterization of the cells. Additionally, simultaneous fluorescence imaging was conducted to verify the accuracy of the cell sorting. Consistent imaging conditions were maintained across sessions to facilitate comparability of the acquired images.

3.5 Data processing

In this study, automated cell detection and image processing were conducted using the YOLOv5 object detection framework

© 2026 The Author(s). Published by the Royal Society of Chemistry

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM.
This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

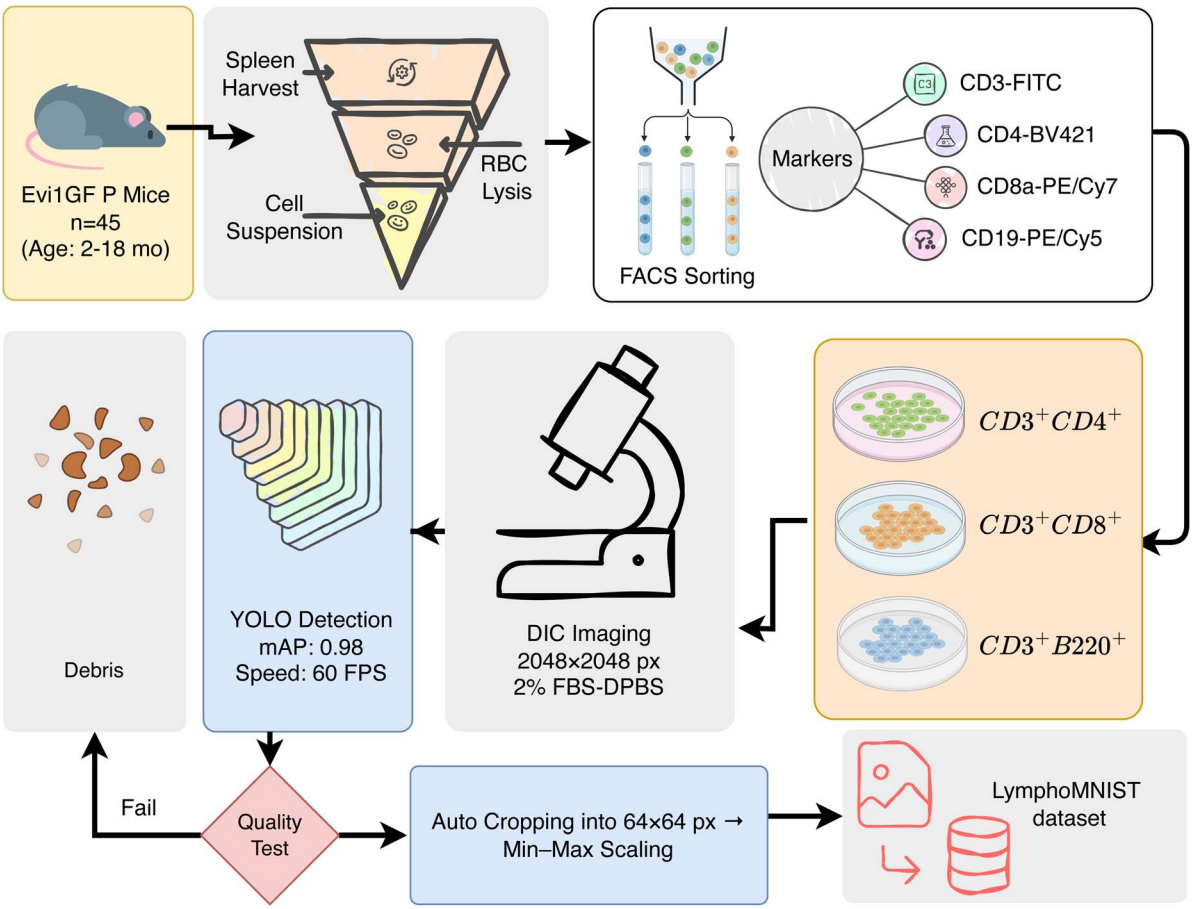

Fig. 7 Flowchart of LymphoMNIST data collection and preprocessing. Splenocytes were FACS-sorted into T4, T8, and B subsets, imaged by DIC microscopy, cropped with YOLOv5, pathologist-verified, and standardized to 64 × 64 pixels for dataset generation.

Table 2 Comparison of our method with other SOTAs

Method	Accuracy	Latency	Platform	App
Ours	86%	14.5 μs	FPGA	Cell sort
Ours	90%	325 μs	GPU	Cell sort
Prior SOTA[2]	—	183 μs	CPU	Cell sort
CellSighter[30]	88–93% (recall)	—	GPU	Cell class
FPGA DL[31]	89.5%	652 μs	FPGA	Obj. Class

(Fig. 7). Given the challenges posed by bright-field microscopy images, such as overlapping cells, debris, and lighting artifacts, YOLOv5 demonstrated exceptional accuracy and efficiency, achieving 98% detection accuracy on our validation subset compared to 82% for Watershed-based segmentation. This ensured a reliable dataset with minimal preprocessing errors that could impact downstream classification. YOLOv5 also automated the cropping process, reducing manual labor by over 300 hours, whereas traditional methods like thresholding and Watershed segmentation required manual correction for 30% of images in pilot tests, introducing variability and delays. YOLOv5 efficiently identified and cropped individual cells from DIC images, standardizing each to 64 × 64 pixels centered on the cell, minimizing variability for downstream machine learning tasks. Its feature extraction capabilities detected cells despite variations in size, shape, or orientation, enabling high-throughput processing. Manual inspection filtered misidentifications like debris or clusters, ensuring only correctly identified T4, T8, and B cells were retained. This workflow balanced efficiency and accuracy, yielding 80 000 images split into training, testing, and validation sets as described in Results.

4 Conclusion

We have developed a label-free machine learning framework for the classification of lymphocytes—specifically T4, T8, and B cells—using bright-field microscopy images. Utilizing a teacher-student model architecture with knowledge distillation, we achieved high accuracy while significantly reducing model complexity. In future work, we will extend the model by implementing additional FPGA hardware for the object detection component, as the current version only focuses on object classification. This hardware integration will enable real-time, high-throughput lymphocyte detection and sorting, enhancing its utility in clinical settings. Furthermore, expanding the model to classify rare lymphocyte subsets or those involved in specific diseases may increase its clinical relevance. This framework presents a significant advancement and new SOTA in

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM. This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

lymphocyte classification and general cell-sorting by offering a non-invasive, efficient, ultra-low latency, scalable solution. It provides a strong foundation for the development of automated, label-free cell sorting technologies for both research and clinical applications.

Conflicts of interest

There are no conflicts to declare.

Code availability

All models and experiments described in this work were implemented using Python with PyTorch as the core deep learning framework. Complete scripts, model training recipes, and instructions necessary to reproduce the experiments and results reported in this study are openly accessible[40] *via* **https://github.com/Khayrulbuet13/LymphoML**.

Data availability

The LymphoMNIST dataset described in this work is publicly available to promote reproducibility and facilitate further research.[13] Researchers can readily access and integrate the dataset into their workflows using the Python package LymphoMNIST. Comprehensive documentation, installation guidelines, dataset exploration methods, and implementation examples are provided in the project's GitHub repository (**https://github.com/Khayrulbuet13/LymphoMNIST**).

Supplementary information is available. See DOI: **https://doi.org/10.1039/d5dd00345h**.

Acknowledgements

This work was partially supported by the National Science Foundation (NSF) grant number 2215789 and the NSFPOSE Phase II Award 2303700. K. I. gratefully acknowledges Lawrence Livermore National Laboratory (LLNL) for supporting the continuation of this research during his postdoctoral tenure. R. F. gratefully acknowledges support from the Ryan Fellowship and the International Institute for Nanotechnology at Northwestern University.

References

1 C. M. MacLaughlin, *et al.*, Surface-enhanced Raman scattering dye-labeled Au nanoparticles for triplexed detection of leukemia and lymphoma cells and SERS flow cytometry, *Langmuir*, 2013, **29**, 1908–1919, DOI: **10.1021/la303931c**.

2 A. A. Nawaz, *et al.*, Intelligent image-based deformation-assisted cell sorting with molecular specificity, *Nat. Methods*, 2020, **17**, 595–599, DOI: **10.1038/s41592-020-0831-y**.

3 A. A. Nawaz, *et al.*, Image-based cell sorting using focused travelling surface acoustic waves, *Lab Chip*, 2023, **23**, 372–387, DOI: **10.1039/d2lc00636g**.

4 X. Xu, *et al.*, Substrates mimicking the blastocyst geometry revert pluripotent stem cell to naivety, *Nat. Mater.*, 2024, **23**, 1748–1758, DOI: **10.1038/s41563-024-01971-4**.

5 S. Wang, *et al.*, Label-free detection of rare circulating tumor cells by image analysis and machine learning, *Sci. Rep.*, 2020, **10**, 12226, DOI: **10.1038/s41598-020-69056-1**.

6 K. Islam, *et al.*, MIML: multiplex image machine learning for high precision cell classification *via* mechanical traits within microfluidic systems, *Microsyst. Nanoeng.*, 2025, **11**, 43.

7 I. Iqbal, I. Ullah, T. Peng, W. Wang and N. Ma, An end-to-end deep convolutional neural network-based data-driven fusion framework for identification of human induced pluripotent stem cell-derived endothelial cells in photomicrographs, *Eng. Appl. Artif. Intell.*, 2025, **139**, 109573, DOI: **10.1016/j.engappai.2024.109573**.

8 L. Zhang, *et al.*, Generalizing deep learning for medical image segmentation to unseen domains *via* deep stacked transformation, *IEEE Trans. Med. Imag.*, 2020, **39**, 2531–2540, DOI: **10.1109/TMI.2020.2973595**.

9 K. Sytwu, L. Rangel DaCosta, C. Groschner and M. C. Scott, Maximizing neural net generalizability and transfer learning success for transmission electron microscopy image analysis in the face of small experimental datasets, *Microsc. Microanal.*, 2022, **28**, 3124–3126, DOI: **10.1017/s1431927622011631**.

10 R. Tang, *et al.*, Low-latency label-free image-activated cell sorting using fast deep learning and AI inferencing, *Biosens. Bioelectron.*, 2023, **220**, 114865, DOI: **10.1016/j.bios.2022.114865**.

11 Y. Gu, *et al.*, Machine learning based real-time image-guided cell sorting and classification, *Cytom. J. Int. Soc. Anal. Cytol.*, 2019, **95**, 499–509, DOI: **10.1002/cyto.a.23764**.

12 Y. Li, *et al.*, Deep cytometry: Deep learning with real-time inference in cell sorting and flow cytometry, *Sci. Rep.*, 2019, **9**, 11088, DOI: **10.1038/s41598-019-47193-6**.

13 K. Islam, *et al.*, *LymphoMNIST Dataset*, 2025, DOI: **10.5281/zenodo.17352233**.

14 K. He, X. Zhang, S. Ren and J. Sun, Deep Residual Learning for Image Recognition, *arXiv*, 2015, preprint, arXiv:1512.03385, DOI: **10.48550/arXiv.1512.03385**.

15 M. Rahimzadeh, S. Parvin, A. Askari, E. Safi and M. R. Mohammadi, Wise-SrNet: a novel architecture for enhancing image classification by learning spatial resolution of feature maps, *Pattern Anal. Appl.*, 2024, **27**, 1–23, DOI: **10.1007/s10044-024-01211-0**.

16 S. Saponara and A. Elhanashi, *Impact of image resizing on deep learning detectors for training time and model performance, 10–17. Lecture notes in electrical engineering*, Springer International Publishing, 2022.

17 S. Ethiraj and B. K. Bolla, Augmentations, An insight into their effectiveness on convolution neural networks, *arXiv*, 2022, preprint.

18 S. Ethiraj and B. K. Bolla, Augmentations: An Insight into Their Effectiveness on Convolution Neural Networks, in *Advances in Computing and Data Sciences, ICACDS 2022*, ed. M. Singh, V. Tyagi, P. K. Gupta, J. Flusser, T. Ören, Communications in Computer and Information Science,

© 2026 The Author(s). Published by the Royal Society of Chemistry

Open Access Article. Published on 24 November 2025. Downloaded on 6/13/2026 9:45:07 PM. This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.

Springer, Cham, 2022, vol. 1613, DOI: 10.1007/978-3-031-12638-3_26.
19 F. Faghri, *et al.*, Reinforce Data, Multiply Impact: Improved Model Accuracy and Robustness with Dataset Reinforcement, *arXiv*, 2023, preprint, arXiv:2304.05895, DOI: 10.48550/arXiv.2304.05895.
20 B. Turan, *et al.*, High accuracy detection for T-cells and B-cells using deep convolutional neural networks, *ROBOMECH J.*, 2018, **5**, 1–9, DOI: 10.1186/s40648-018-0128-4.
21 M. Nassar, *et al.*, Label-Free Identification of White Blood Cells Using Machine Learning, *Cytom. J. Int. Soc. Anal. Cytol.*, 2019, **95**, 836–842, DOI: 10.1002/cyto.a.23794.
22 L. Beyer, *et al.*, Knowledge distillation: A good teacher is patient and consistent, *arXiv*, 2021, preprint, arXiv:2106.05237, DOI: 10.48550/arXiv.2106.05237.
23 H. Wang, S. Lohit, M. Jones and Y. Fu, What makes a "good" data augmentation in knowledge distillation – A statistical perspective, *arXiv*, 2020, preprint, arXiv:2012.02909, DOI: 10.48550/arXiv.2012.02909.
24 M. Ballout, U. Krumnack, G. Heidemann and K.-U. Kühnberger, Efficient knowledge distillation: Empowering small language models with teacher model insights, *arXiv*, 2024, preprint, arXiv:2409.12586, DOI: 10.48550/arXiv.2409.12586.
25 J. Ba and R. Caruana, Do Deep Nets Really Need to be Deep?, *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence and K. Q. Weinberger, Curran Associates, Inc., 2014, vol. 27.
26 A. Romero, *et al.*, FitNets: Hints for thin deep nets, *arXiv*, 2014, preprint, arXiv:1412.6550, DOI: 10.48550/arXiv.1412.6550.
27 G. Hinton, O. Vinyals and J. Dean, Distilling the knowledge in a neural network, *arXiv*, 2015, preprint, arXiv:1503.02531, DOI: 10.48550/arXiv.1503.02531.
28 J. Gou, B. Yu, S. J. Maybank and D. Tao, Knowledge distillation: A survey, *Int. J. Comput. Vis.*, 2021, **129**, 1789–1819, DOI: 10.1007/s11263-021-01453-z.
29 J. Jin, *et al.*, Robotic data acquisition with deep learning enables cell image-based prediction of transcriptomic phenotypes, *Proc. Natl. Acad. Sci. U. S. A.*, 2023, **120**, e2210283120, DOI: 10.1073/pnas.2210283120.
30 M. Pang, T. K. Roy, X. Wu and K. Tan, CelloType: a unified model for segmentation and classification of tissue images, *Nat. Methods*, 2014, 1–10, DOI: 10.1038/s41592-024-02513-1.
31 A. Mouri Zadeh Khaki and A. Choi, Optimizing deep learning acceleration on FPGA for real-time and resource-efficient image classification, *Appl. Sci.*, 2025, **15**, 422, DOI: 10.3390/app15010422.
32 T. Aarrestad, *et al.*, Fast convolutional neural networks on FPGAs with hls4ml, *Mach. learn.: sci. technol.*, 2021, **2**, 045015, DOI: 10.1088/2632-2153/ac0ea1.
33 A. Paszke, *et al.*, PyTorch: An imperative style, high-performance deep learning library, *arXiv*, 2019, preprint, arXiv:1912.01703, DOI: 10.48550/arXiv.1912.01703.
34 M. Abadi, *et al.*, TensorFlow: Large-scale machine learning on heterogeneous distributed systems, *arXiv*, 2016, preprint, arXiv:1603.04467, DOI: 10.48550/arXiv.1603.04467.
35 keras: Deep Learning for humans.
36 S. Hashemi, N. Anthony, H. Tann, R. I. Bahar and S. Reda, Understanding the impact of precision quantization on the accuracy and energy of neural networks, in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, 2017, DOI: 10.23919/date.2017.7927224.
37 Y. Wei, R. F. Forelli, H. Hansen, J. P. Levesque, N. Tran, J. C. Agar, G. Di Guglielmo, M. E. Mauel, G. A. Navratil, *et al.*, Low latency optical-based mode tracking with machine learning deployed on FPGAs on a tokamak, *Rev. Sci. Instrum.*, 2024, **95**(7), 073509, DOI: 10.1063/5.0190354.
38 *Euresys. CustomLogic*, 2021, https://www.euresys.com/en/CustomLogic, Software, accessed, 2024-10-22.
39 R. Forelli, *hls4ml-frame-grabbers*, Software, Fast Machine Learning Lab, 2024, https://github.com/fastmachinelearning/hls4ml-frame-grabbers, accessed: 2024-10-22.
40 K. Islam, *et al.*, LymphoML: Training and Deployment Code for Real-Time Cell Sorting, 2025, DOI: 10.5281/zenodo.17370318.