Processing Code-Multiplexed Coulter Signals via Deep Convolutional Neural Networks

Ningquan Wang,[a] Ruxiu Liu,[a] Norh Asmare,[a] Chia-Heng Chu,[a] A. Fatih Sarioglu[*abc]

[a] School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, United States
[b] Petit Institute for Bioengineering and Biosciences, Georgia Institute of Technology, Atlanta, GA 30332, United States
[c] Institute of Electronics and Nanotechnology, Georgia Institute of Technology, Atlanta, GA 30332, United States

Abstract

Beyond their conventional use of counting and sizing particles, Coulter sensors can be used to spatially track suspended particles, with multiple sensors distributed over a microfluidic chip. Code-multiplexing of Coulter sensors allows such integration to be implemented with simple hardware but requires advanced signal processing to extract multi-dimensional information from the output waveform. In this work, we couple deep-learning based signal analysis with microfluidic code-multiplexed Coulter sensor networks. Specifically, we train convolutional neural networks to analyze Coulter waveforms not only to recognize certain sensor waveform patterns but also to resolve interferences among them. Our technology predicts the size, speed, and location of each detected particle. We show that the algorithm yields a >90% pattern recognition accuracy for distinguishing non-correlated waveform patterns at a processing speed that can potentially enable real-time microfluidic assays. Furthermore, once trained, the algorithm can readily be applied for processing electrical data from other microfluidic devices integrated with the same Coulter sensor network.

Keyword

lab-on-a-chip; Coulter sensing; machine learning; deep learning; convolutional neural network; impedance cytometry.

1. Introduction

Coulter counters excel at rapid enumeration and sizing of suspended particles and therefore find widespread use in different applications such as hematology,[1,2] oncology,[3,4] microbiology,[5,6] pathology,[7,8] pharmacology,[9,10] industrial applications,[11,12] and environmental monitoring.[13,14] What makes Coulter counters practically attractive for those applications is their ability to transduce particle information directly into electrical signals that can readily be interpreted. In a Coulter counter, a pore-bearing membrane is placed between two electrolyte-filled chambers. When the particles of interest, initially suspended in one of the chambers, are driven across the membrane, the electrical impedance is modulated as particles pass through the pore. The number and size of particles can be determined from the number and the amplitude of the intermittent changes in the electrical current, respectively.[15,16]

Coulter counters can also be implemented in Lab-on-a-Chip (LoC) platforms to create integrated systems for the quantitative characterization of samples. In fact, microfluidic channels manufactured with the photolithographic resolution on LoC devices enable precise pore dimensions that can be tuned to maximize sensitivity and resolve particle coincidences.[17] Capitalizing on these benefits, Coulter counters have been used for a variety of applications including the assessment of cell deformability,[18] impedance cytometry,[19,20,21] single-cell monitoring,[22,23] nanoscale and molecular characterization,[24,25] DNA sequencing,[26] and protein analysis.[27,28,29]

While conventional Counter counters can only count and size suspended particles, we have recently shown that a network of Coulter counters, when distributed across a microfluidic chip, can be employed to track locations of those particles for microfluidic manipulation-based sample characterization.[30] Our technique, Microfluidic CODES, patterns Coulter sensor electrodes to form distinct electrode patterns at various nodes across a microfluidic device so that particles flowing by those nodes produce distinct waveforms.[31,32] Because the whole sensor network is essentially a single Coulter counter with micropatterned electrodes, information coming from different nodes on the device are code-multiplexed in a single output waveform. By decoding this waveform through signal processing, it is possible to measure the size, speed, and location of particles manipulated in a microfluidic device. Given microfluidics offers extensive manipulation capabilities[33] to fractionate cell populations under various force fields, an integrated spatiotemporal readout ,such as the Microfluidic CODES, therefore transforms a microfluidic device into a cytometer, capable of measuring the cell property, based on which, cells are differentially manipulated on the microfluidic device. In fact, using the Microfluidic CODES platform for tracking manipulated cells, we have developed electronic cytometers that can identify cell membrane antigens,[36] measure cell surface expression,[30,37] or mechanical properties.[38]

Microfluidic CODES - based cytometers have several advantages over conventional cytometers. First, compared to traditional impedance-based flow cytometers that only count and size cells, the Microfluidic CODES also tracks the location of manipulated cells, providing another dimension of information for cell analysis. Second, the Microfluidic CODES can measure any cell property, not necessarily measurable by a conventional cytometer, as long as the cell property can be used for differential microfluidic manipulation. Third, the use of electrical sensors instead of optical detection allows system integration and miniaturization to realize low-cost and portable systems that can perform as accurate as conventional systems.[30] Finally, compared to imaging-based cytometry, which can also provide spatial information on cell manipulation, the Microfluidic CODES (1) offers a nonrigid “field of view” that can be tuned to any microfluidic platform for cell manipulation, (2) has higher sub-millisecond temporal resolution, which can only be matched by specialized high-speed camera systems and (3) can efficiently compress spatial measurements on cells into an electrical waveform that could be processed more efficiently than a video footage.

How reliably and rapidly the code-multiplexed information from the Coulter sensor network can be processed determines the extent that the complexity of the hardware can be shifted towards

software. In a conventional code division multiple access (CDMA) network, codes assigned to individual sources are specifically designed to be mutually orthogonal so that information can be recovered, with a high signal-to-noise ratio, through correlation with a template library.[39,40] Likewise, Microfluidic CODES employed Gold sequences,[41,42] which were designed to remain mutually orthogonal under an asynchronous transmission. While successful in discriminating signals from different sensors even if they interfere due to coincident particles, reliance on specialized code sequences introduces challenges on both the physical and computational aspects of the system. On the physical side, the orthogonality constraint limits the number of Coulter sensors in the network and requires a complex sensor design for scaling.[43] On the computational side, the template matching and recursive approaches like the successive interference cancellation (SIC) are computationally expensive and preclude real-time implementations. Therefore, a more efficient signal processing algorithm enabling a straightforward coding scheme is required to improve the scalability, performance, and hence, the utility of the Microfluidic CODES.

Recently, machine learning (ML) has become a key research area in data analysis and signal processing. Unlike model-based signal processing, ML focuses on providing a machine with the ability to learn from experience without being explicitly programmed. More specifically, ML-based algorithms update and optimize their internal parameters by learning from an existing dataset (training data) and make predictions on a future unseen dataset (testing data). Currently, ML has been widely used in areas including computer vision[44] and healthcare.[45] Among various ML models, deep learning[46] is a popular learning model for complex pattern recognition tasks. Deep learning is a representation learning method, which allows a machine to automatically learn and discover the representations of input data needed for performing further pattern recognition. Like the vast network of neurons in a brain, a deep learning structure (deep neural network) is based on a multi-layer of artificial neurons, each of which is a computational node that is capable of performing a non-linear transformation on its input. In this way, a deep neural network combines the computational power of multiple artificial neurons, and solves highly non-linear problems, especially in time series processing.[47,48,49]

In this paper, we introduce deep learning enhanced microfluidic Coulter sensor networks, in which code-multiplexed Coulter signals are interpreted by a data-based pattern recognition algorithm. Specifically, we first design and fabricate a microfluidic system with a network of 10 code-multiplexed Coulter sensors, which are encoded to produce randomly-designed non-orthogonal waveforms. Then we build a signal processing algorithm based on the convolutional neural network (ConvNet),[50] a specific type of deep learning structure, to interpret sensor signals. We test the device with a cell suspension and use the recorded signals to train our algorithm not only to discriminate between different signature waveforms but also to resolve interfering sensor waveforms due to coincident events. We later employ the trained algorithm to analyze experimental data on cell suspensions and characterize its performance by benchmarking against independent measurements using high-speed optical microscopy.

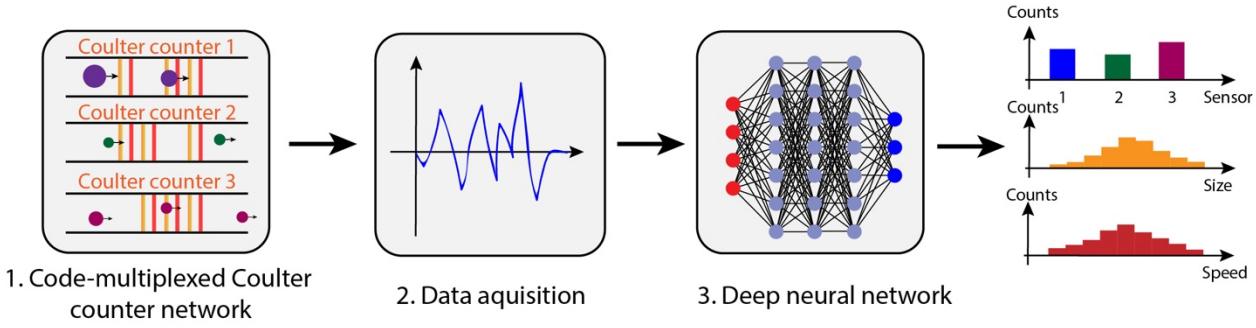

Figure 1: A schematic showing the process workflow. The data is generated by a microfluidic platform equipped with a code-multiplexed Coulter sensor network. A time waveform of the electrical current is acquired through a data-acquisition system. The waveform is processed by a trained neural network, which infers the sensor identity and particle parameters from the signal pattern. Results are classified and aggregated to provide particle statistics.

2. Material and Methods

2.1 System Overview

The workflow of the entire system developed in this work can be divided into three blocks (Figure 1). First, suspended microparticles were manipulated in a microfluidic device integrated with a code-multiplexed Coulter sensor network. Microparticles, sorted into different locations, were then intercepted by one of the coded Coulter sensors integrated on the chip. Each Coulter sensor in the network was designed with a unique electrode pattern and produced a distinct electrical signal (signature waveform) dictated by the underlying electrode pattern (Section 2.2). Second, a data acquisition system was built to drive the Coulter sensor network and measure the impedance changes due to flowing particles by recording intermittent changes in the total electrical current flow in the Coulter sensor network. This detection scheme combined signals from Coulter sensors, distributed on the chip, into a single, 1-dimensional time waveform. This waveform contained different signature waveforms of varying amplitudes and durations coming from individual sensors and also interfering sensors for times when multiple particles coincidently interacted with the sensor network (Section 2.3). Third, deep neural networks were designed and trained to interpret the output waveform. Trained neural networks provided the size, flow speed, and sensor identity for each particle detected on the microfluidic chip. The detailed implementation of the neural networks is illustrated in Section 2.4, and the training process of the neural networks is demonstrated in Section 3.2. Performance characterization of the trained neural network was done by processing experimental signals and comparing the results with independent measurements using high-speed optical microscopy (Section 3.4).

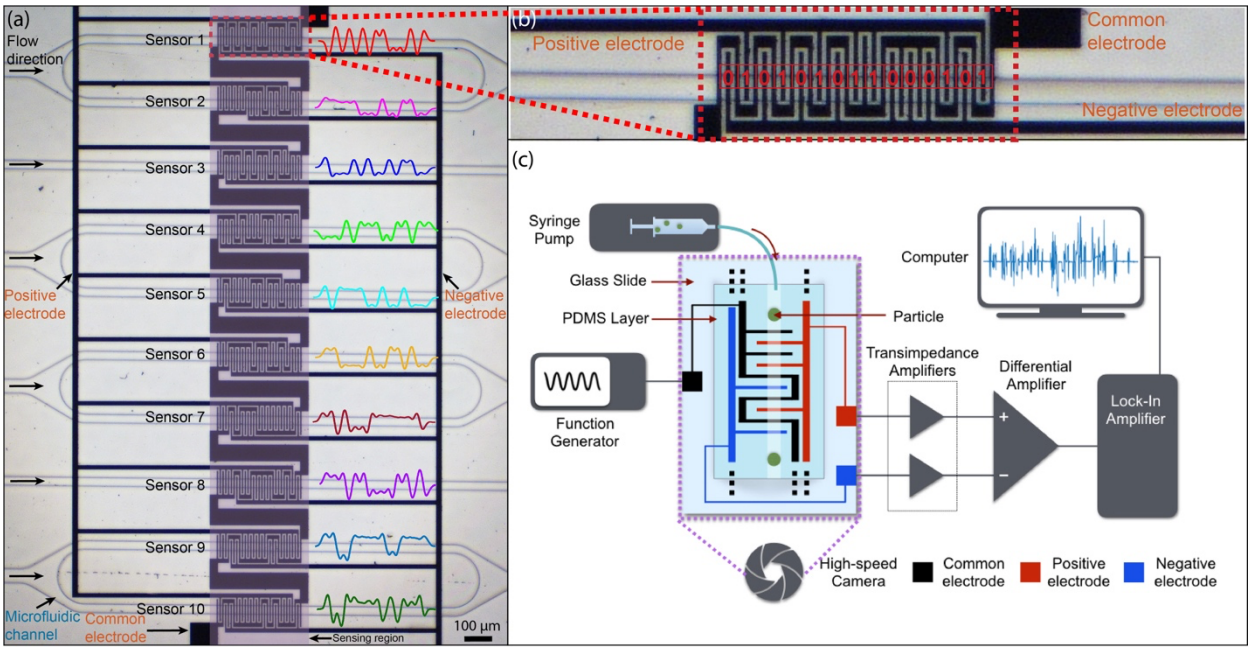

Figure 2: Microfluidic device design and the experimental set-up. (a) A microscope image of the code-multiplexed Coulter sensor platform. Au electrodes are micropatterned on a glass substrate to form 10 coded Coulter sensors with unique electrode patterns. Ten parallel PDMS microfluidic channels are aligned with sensors. (b) A close-up image of the first coded Coulter sensor with the assigned code sequence of 010101011000101. (c) The experimental setup for signal acquisition from the microfluidic device.

2.2 Microfluidic Device Design and Fabrication

As a test platform, we designed a code-multiplexed Coulter sensor network with 10 sensors. Each sensor was designed to produce a distinct but non-orthogonal waveform. To create the codeset, we generated 10 15-bit binary code sequences, where each bit was treated as a Bernoulli random variable with p = 0.5. Specifically, the generated code sequences were:

Sensor 1: 010101011000101;
Sensor 2: 111110001001100;
Sensor 3: 100010100101100;
Sensor 4: 000101110011011;
Sensor 5: 101111001001000;
Sensor 6: 110000100110100;
Sensor 7: 110100011111110;
Sensor 8: 111011000011010;
Sensor 9: 110011111001111;
Sensor 10: 100111110101110.

The sensor network was created on a glass substrate with micromachined electrodes coupled with a microfluidic layer. On the glass substrate, a thin gold layer was patterned to form the sensor network created by three coplanar electrodes: one common electrode to excite the sensor network, and two sensing electrodes, one positive and one negative, to acquire the output signal (Figure 2a, 2b). In the sensing region (Figure 2a, colored section), the electrodes were arranged as an interdigitated array with 5 µm-wide electrode fingers separated by 5 µm-wide gaps. For each Coulter sensor, the spatial arrangement of positive and negative sensing electrode fingers was determined by the assigned code sequence. The common electrode was then routed between the sensing electrodes to uniformly excite the sensor network.

The device was fabricated using a combination of surface micromachining and soft lithography. Specifically, the glass substrate with patterned electrodes was fabricated using a lift-off process. A 1.2 µm-thick negative photoresist (NR9-1500PY, Futurrex, Inc.) was patterned on a glass slide using a maskless photolithography system (MLA150, Heidelberg Instruments), followed by an e-beam evaporation of a 20/480 Cr/Au film stack. The glass substrate was then immersed in acetone to strip the non-patterned photoresist region and diced into individual chips. The microfluidic layer was fabricated out of polydimethylsiloxane (PDMS) using a soft lithography process. A 15 µm-thick SU-8 photoresist (MicroChem) was spun and patterned on a 4-inch silicon wafer to create the mold. The PDMS prepolymer (Sylgard 184, Dow Corning) was mixed with the crosslinker at a 10:1 ratio, and then poured on the mold, degassed, and baked at 65 °C for >4 hours. The cured PDMS was then peeled off from the mold and punched using a biopsy punch to create the fluidic inlet and outlet. The glass substrate and the PDMS layer were then activated in an oxygen plasma environment, aligned and bonded to form the final device. A more detailed explanation of the design and the fabrication process can be found elsewhere.[51]

2.3 Experimental Set-up

In this work, human ovarian (HeyA8), breast (MDA-MB-231) and prostate (PC3) cancer cell lines were used as a simulated biological sample to acquire experimental data for the training and characterization of the deep learning model. Cells were cultured in a culture medium (Mediatech; Cellgro, Herndon, VA) supplemented with 10% fetal bovine serum (FBS; Seradigm, Radnor, PA) and maintained in a cell culture incubator in 5% CO_2 atmosphere at 37 °C. Once the cells reached >80% confluence, they were harvested by treating with trypsin, pelleting by centrifugation, and spiking into phosphate buffered saline (PBS) with gentle pipetting.

The cell suspension was then driven through the microfluidic device at a constant flow rate of 500 µL/h using a syringe pump. A 460-kHz sine wave (2 Vpp) was applied to the common electrodes to excite the Coulter sensor network, and the output signal was acquired from the sensing electrodes and followed a signal path comprised of transimpedance amplifiers and a differential amplifier. A lock-in amplifier (HF2LI, Zurich Instruments) was used to demodulate the signal, and the demodulated signal was sampled into a computer with a sampling rate of 57 kHz for processing. Besides the electrical signal recorded by the described electronic setup, the interactions between the cells and the sensor network were also monitored and recorded simultaneously using an inverted optical microscope (Nikon Eclipse Ti-U, Nikon) equipped with a high-speed camera (Phantom v7.3, Vision Research). The recorded video footage was later used

for benchmarking the performance of our algorithm in interpreting the events inside the microfluidic chip.

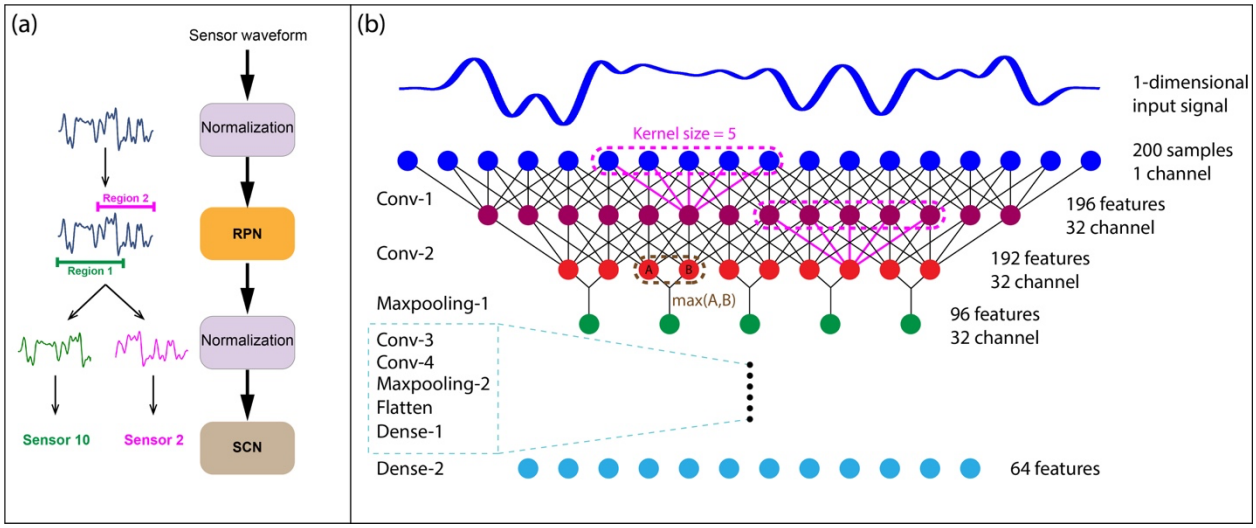

Figure 3: Implementation of the developed 2-stage deep-learning based algorithm. (a) A schematic showing the algorithm workflow. Given an input signal, the first stage ConvNet (RPN) searches for intervals that contain signature waveforms. The second stage ConvNet (SCN) predicts the sensor identity corresponding to each signature waveform extracted by the RPN. (b) A schematic of the designed ConvNet structure. Both the RPN and the SCN use the same ConvNet structure.

2.4 Algorithm Design

Our algorithm employed ConvNets, which were often used in image recognition because of their effectiveness in representing local saliences in an image. Here, we analogized the recognition of signature waveforms to the recognition of objects in a 1-dimensional space. Our ConvNet consisted of several specific artificial layers, including convolutional layers, rectified linear units (ReLU) layers, pooling layers, and dense (fully-connected) layers. The convolutional layer extracted features from the input feature map using multiple sliding feature detectors (small kernels with specific weights and bias). The ReLU layer introduced nonlinear properties to the system. The pooling layers performed downsampling operations to the input feature map, decreasing the number of trainable parameters.

To process the code-multiplexed Coulter sensor signal, we developed a two-stage ConvNet structure (Figure 3a). The first stage ConvNet was the region proposal network (RPN), which searched an input signal for regions (bounding boxes) that potentially contained signature waveforms. At the same time, the scale of each bounding box was used to estimate the amplitude and duration of the signature waveform providing information on the size and speed of the corresponding particle, respectively. The second stage ConvNet was the sensor classification network (SCN), which was trained to perform sensor-identity classification on signature waveforms extracted from the first stage. The SCN predicted the probability with which the input signature waveform belonged to each and every Coulter sensor in the network integrated on the microfluidic device.

The RPN and the SCN shared the same structure for feature extraction (Figure 3b). We adapted our ConvNet structure from a work[52] that aims pattern recognition in grayscale images. The structure was optimized using the Bayesian optimization algorithm.[53] We chose this structure due to several reasons: (1) We analogize classification of sensor waveforms in an electrical signal to object recognition in an image frame; (2) Greyscale images have only one channel, like our electrical waveform and therefore the ConvNet can be compact for faster processing. Both of our ConvNets contained 4 convolutional layers, each of which was activated by a ReLU layer. A max-pooling layer was placed after the second and the forth convolutional layers. Two dense layers were placed at last. The model had a total of 217056 trainable parameters. For reproducibility, detailed information on our ConvNet design parameters is presented in Table 1.

	C-size	C-Stride	C-pad	Act	P-size	P-stride	Params	O/P shape
Input	-	-	-	-	-	-	-	(200)
Conv-1	(5)	1	0	-	-	-	192	(32, 196)
Activation-1	-	-	-	ReLU	-	-	-	(32, 196)
Conv-2	(5)	1	0	-	-	-	5152	(32, 192)
Activation-2	-	-	-	ReLU	-	-	-	(32, 192)
Maxpooling-1	-	-	-	-	(2)	2	-	(32, 96)
Conv-3	(5)	1	0	-	-	-	10304	(64, 92)
Activation-3	-	-	-	ReLU	-	-	-	(64, 92)
Conv-4	(5)	1	0	-	-	-	20544	(64, 88)
Activation-4	-	-	-	ReLU	-	-	-	(64, 88)
Maxpooling-2	-	-	-	-	(2)	2	-	(64, 44)
Flatten	-	-	-	-	-	-	-	(2816)
Dense-1	-	-	-	-	-	-	180224	(64)
Activation-5	-	-	-	ReLU	-	-	-	(64)
Dense-2	-	-	-	-	-	-	640	(-)

*Table 1: ConvNet design parameters. (**C-size**: Kernel size of the convolutional layer. **C-stride**: Stride size of the convolutional layer. **C-pad**: Zero-padding of the convolutional layer. **Act**: Type of the activation function. **P-size**: Kernel size of the pooling layer. **P-stride**: Stride size of the pooling layer. **Params**: Number of trainable parameters of the layer. **O/P shape**: Output dimension of the layer.)*

3. Results and Discussion

3.1 Training Data Construction

We first processed recorded sensor waveforms to construct the training data for ConvNets. To extract representative sensor waveforms from the raw sensor output signal, we developed a signal-identification program (Figure 4a). With this program, sensor waveforms were discovered by computing the signal variance within a sliding window as the window traversed the entire raw sensor output signal. The sole purpose of this process was to identify and mark the regions of

interest in the raw sensor output signal with potential sensor activity to be used in subsequent operations.

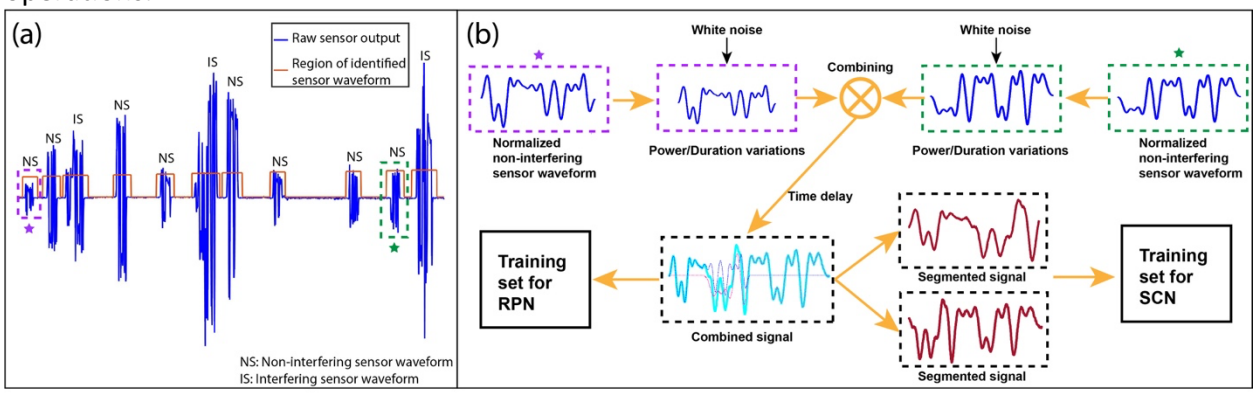

Figure 4: Construction of the training data. (a) A search algorithm is implemented to detect waveforms of sensor activity in the raw sensor output signal. A correlation-based algorithm is used to classify each detected sensor signal as a non-interfering sensor waveform or an interfering sensor waveform. (b) A workflow schematic for the digital data augmentation process employed to increase the size of the training dataset.

To automatically label each identified sensor waveform with the corresponding sensor identity, we implemented a correlation-based algorithm. By computing the cross-correlation between each extracted sensor waveform with a template library containing all code sequences from Section 2.2, the algorithm obtained two vital pieces of information about each waveform. First, it determined if the waveform was a non-interfering sensor waveform (i.e., contained only one signature waveform), or an interference sensor waveform, which contained multiple signature waveforms interfering with each other. This differentiation was achieved by comparing the amplitude of the primary correlation peak to that of the secondary correlation peak. Second, for each non-interfering sensor waveform, the algorithm identified and labeled its corresponding sensor identity based on the code template that produced the primary correlation peak. At the same time, the power and duration of each labeled non-interfering sensor waveform were also calculated. Labeled non-interfering sensor waveforms were first manually checked for an accuracy assertion, then normalized, and used to construct the training data.

To increase the number of waveforms available for constructing the training data, and thereby improve the performance of our ConvNets, we employed a data augmentation process[54] on the labeled non-interfering sensor waveforms. We first randomly picked waveforms from the dataset and then scaled their power and duration in the digital domain to simulate signals for cells that have different sizes and speeds, respectively. In this process, the power and duration of a waveform were treated as random variables, whose distributions were ensured to match those of the original dataset. Additive white Gaussian noise (SNR = 30 dB, to mimic experimental noise level) was then added to each augmented waveform to introduce variation in the training data set against potential overfitting. We repeated this pick-and-vary process and created a waveform database of 1,000,000 augmented non-interfering sensor waveforms to be used for training data construction.

Besides the non-interfering sensor waveforms, we also generated a database of interfering sensor waveforms to train our algorithm for resolving data from coincident cells. We created interfering sensor waveforms in the digital domain by adding two non-interfering sensor waveforms of known amplitudes and durations with a certain time delay. We repeated this process by randomly drawing different pairs of non-interfering sensor waveforms from the waveform database and adding them with a randomly-determined time delay to create a large database that covers different coincidence scenarios. For this work, we randomly picked 150,000 signals from the waveform database to construct a database of non-interfering sensor waveforms and used the remaining 850,000 to construct a database of interfering sensor waveforms.

Using the constructed non-interfering and interfering sensor waveform database, we created different training data for the RPN and the SCN because of the specific role each ConvNet played in the algorithm. For the RPN, the training data consisted of non-interfering and interfering sensor waveforms directly from the database along with labels on waveforms' amplitudes and durations. For the SCN, the interfering sensor waveforms needed to be pre-conditioned in the digital domain as if they were already processed by a "perfect" RPN because the RPN output was fed into the SCN in our algorithm. Specifically, this process involved extracting sections of an interfering sensor waveform such that the extracted section includes one of the signature waveforms in full along with parts of the contaminating waveform. The extracted section was then labeled with the sensor identity corresponding to the full signature waveform, and it was used to train the SCN to identify the sensor in the presence of interference (Figure 4b).

	Loss function	Optimizer	Regularization	Learning rate	Momentum	Batch size	Epochs
RPN	MSE	SGD	N/A	0.001	0.9	500	50
SCN	Cross-Entropy	SGD	L2	0.001	0.9	500	50

Table 2: Hyper-parameters for ConvNet training.

3.2 ConvNet Training

We trained both ConvNets with a batch size of 500 (batch size: the number of training signals processed before the model is updated) and an epoch number of 50 (epoch number: the number of times the learning algorithm works through the entire training data). In each iteration (iteration number: the number of batches needed to complete one epoch), we updated parameters by employing a stochastic gradient descent (SGD) optimizer. We used the grid search to determine the optimal combination of the learning rate and the momentum. In this process, the learning rate and the momentum was chosen from two different lists ([0.1, 0.01, 0.001, 0.0001, 0.00001] and [0.5, 0.9, 0.99], respectively), which were assembled based on typical values used in practice.[55] For the PRN, we used the mean square error (MSE) to compute the error between actual bounding boxes and predicted bounding boxes. For the SCN, we used the cross-entropy to calculate the classification error. Furthermore, we employed an L2

regularization in training the SCN to prevent overfitting. Hyper-parameters for training the networks are shown in Table 2.

To interpret the trained ConvNets, we also visualized the learned parameters of kernels in each convolutional layer (Figure S1). The kernels in the first two convolutional layers (Figure S1a, S1b) learned first-order features in a coded sensor waveform such as orientations and amplitudes of individual pulses. In deeper convolutional layers, the patterns of kernels became more complex, indicating that the last two layers represented more abstract information, including slopes and transitions between two adjacent pulses (Figure S1c, S1d). This observed hierarchical representation matches with the fact that a ConvNet interprets an input as a hierarchy of features with increasing abstraction.[46] In our ConvNet, a few kernels in deeper layers showed noisy patterns, indicating these kernels were not activated given the specific training data.

3.3 ConvNet Querying

Trained ConvNets were then used to process experimental signals. For non-interfering sensor waveforms (Figure 5a, I), as the input contained only one signature waveform, the RPN only produced one valid bounding box (Figure 5a, II). Then the input signal was clipped according to the bounding box, and the extracted waveform was normalized in power (Figure 5a, III). The normalized waveform was fed into the SCN for sensor identity classification. The sensor identity was determined by the index of the output node with the highest probability value (Figure 5a, IV). For interfering sensor waveforms (Figure 5c, I), multiple bounding boxes were identified (Figure 5c, II). The predicted bounding boxes had different lengths and heights, according to different durations and amplitudes of the detected signature waveforms, respectively. Similarly, the waveform in each bounding box was then extracted, normalized, and processed by the SCN (Figure 5c, III). The SCN then determined the identities of the two sensors that detected the cells and provided the confidence levels for its prediction (Figure 5c, IV). The algorithm predictions were validated using a simultaneously recorded high-speed video of the cell flowing in the device (Figure 5b and 5d).

3.4 ConvNets Testing

3.4.1 Testing of Waveform Boundary Estimation

To test the ConvNets, we first constructed two testing datasets, one for single cells and another for coincident cells. Each of these sets contained signature waveforms from 900 cells. Each ConvNet was then tested separately with these two testing datasets for non-interfering and interfering sensor waveforms. For the RPN, the bounding box regression accuracy on non-interfering sensor waveforms was higher than that on interfering ones (Figure 6a). This difference was expected as the bounding box for a non-interfering sensor waveform was the entirety of the input sensor waveform with only one signature waveform present. In contrast, for an interfering sensor waveform, the interference between signature waveforms resulted in less predictable boundaries effectively leading to lower accuracy. We observed that the accuracy of the RPN for both non-interfering and interfering sensor waveforms increased with the training epoch number and remained stable after 45 epochs (Figure 6a). We achieved a final testing accuracy of 97% on non-interfering sensor waveforms, and 92% on interfering sensor waveforms at epoch 50.

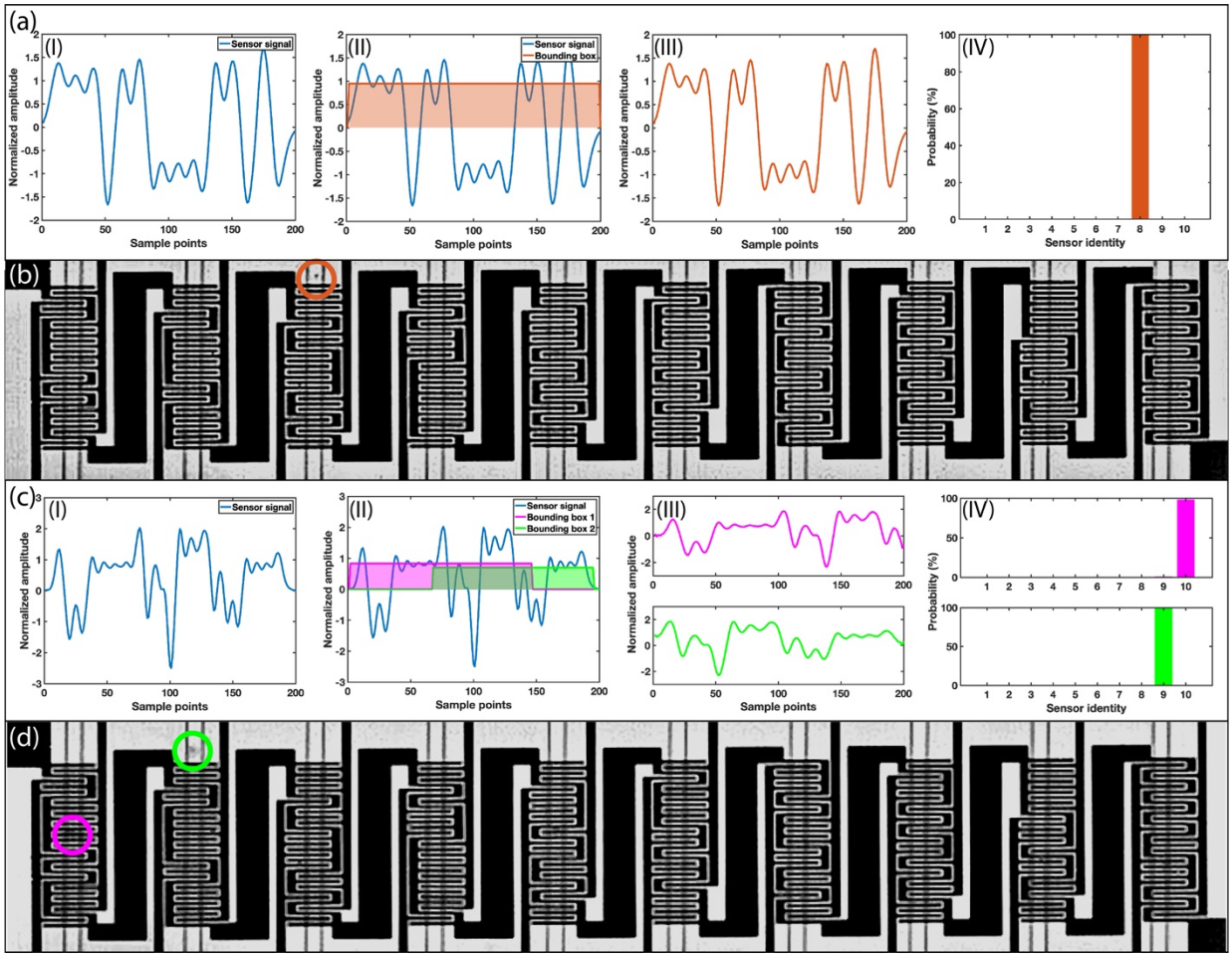

Figure 5: ConvNet decoding process steps. (a) For a non-interfering sensor waveform, the RPN produces one bounding box that contains the signature waveform. The detected signature waveform is then extracted, normalized, and fed into the SCN. The SCN predicts that this signature waveform is generated by sensor 8 with a probability of 99.5%. (b) Simultaneously recorded high-speed camera image confirms a cell flowing over the sensor 8. (c) For an interfering sensor waveform, the RPN produces two bounding boxes for two signature waveforms. The detected signature waveforms are then extracted, normalized, and fed into the SCN. The SCN predicts that these two signature waveforms are generated by sensor 10, with a probability of 97%, and sensor 9, with a probability of 99%, respectively. (d) Simultaneously recorded high-speed camera image confirms two cells concurrently flowing over the sensor 10 and the sensor 9, respectively.

3.4.2 Testing of Cell Size Estimation

Heights of the predicted bounding boxes were used to estimate sizes of detected cells. Because the height of each predicted bounding box corresponded to the amplitude of the identified signature waveform, it could be used to determine the cell volume, according to the Coulter principle.[56] Following the calibration of signal amplitude for cell size using microscope images, we compared algorithm predictions with the actual size data directly calculated from the testing data.

To observe potential effects of interference-induced error in size estimation, we analyzed non-interfering (Figure 6b, I) and interfering (Figure 6b, II) sensor waveforms separately and observed that resulting size distributions closely matched with each other. Furthermore, the size measurements from our algorithm also agreed well with the size distribution directly calculated from the testing data (Figure 6b, III).

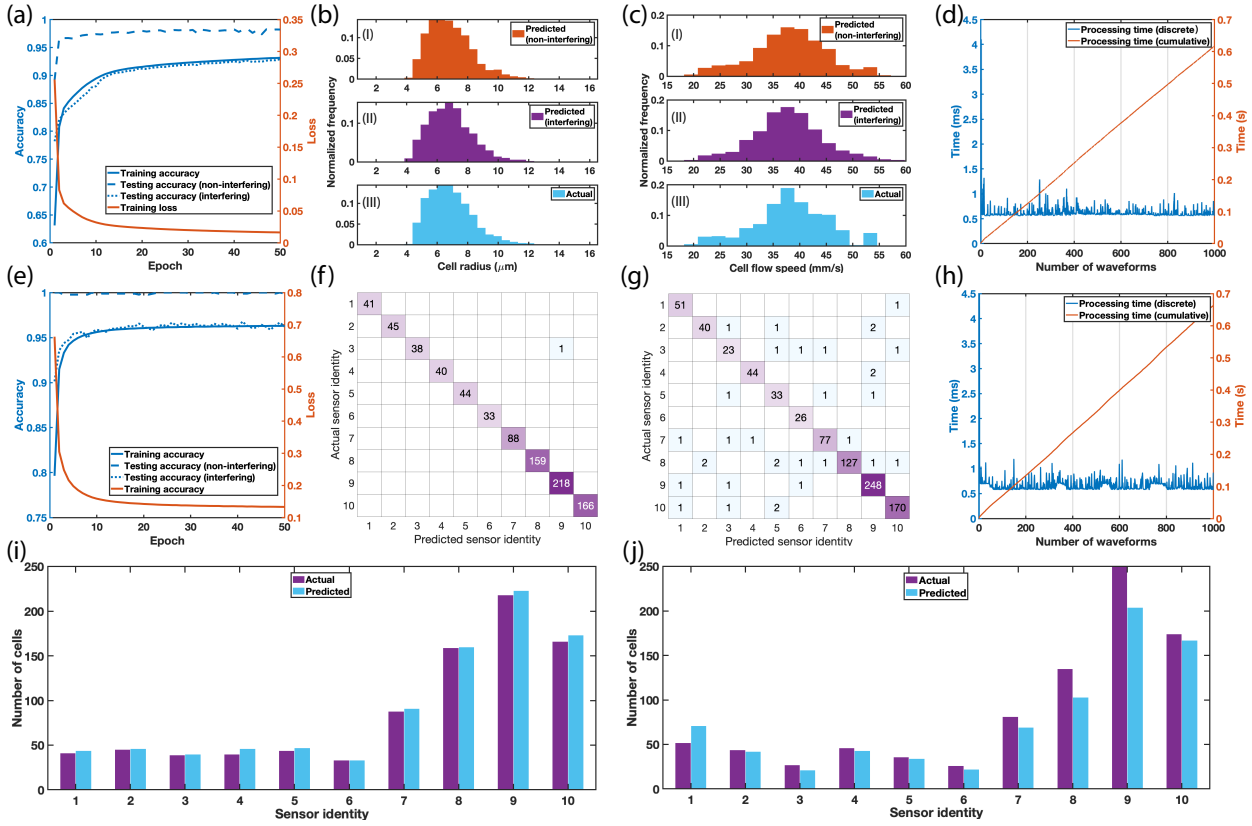

Figure 6: ConvNets performance characterization. (a) Training and testing results for the RPN bounding box regression accuracy. (b) Testing of cell size estimation accuracy (c) Testing of cell speed estimation accuracy. (d) Computation speed test results for the RPN. (e) Training and testing results for the SCN sensor identity classification accuracy. SCN confusion matrices for the (f) the non-interfering sensor waveforms and (g) the interfering sensor waveforms. (h) Computation speed test results for the SCN. Test results for sensor identity estimation accuracy of the cascaded ConvNets for (i) non-interfering sensor waveforms and (j) interfering sensor waveforms.

3.4.3 Testing of Cell Speed Estimation

For the speed estimation, we used the length of each predicted bounding box, which corresponded to the duration of the identified signature waveform. Because the duration of the waveform provided the residence time of a flowing cell in the sensing region, by combining the waveform duration with the physical length of the coded sensor, we could calculate the speed of each cell. Using our algorithm, we calculated the flow speed for single (Figure 6c, I) and coincident (Figure 6c, II) cells separately. The calculated speed distributions for both tests matched, demonstrating the negligible effect of sensor interference on cell speed estimations. The results

were also in close agreement with the speed data (Figure 6c, III) directly calculated from the testing data.

3.4.4 Testing of Sensor Identity Classification

We first tested SCN alone to test its accuracy in sensor identity classification for non-interfering and interfering sensor waveforms. The classification accuracy for non-interfering sensor waveforms was found to be higher than that of interfering ones (Figure 6e). This difference was expected because a non-interfering sensor waveform faithfully followed the pattern of the assigned code sequence. While deviations could result from differences in shape, size, and vertical position of a cell, those were often not at a level to negate the underlying signature waveform. However, for an interfering sensor waveform, part of a signature waveform was by definition distorted by contaminating signature waveforms. The partial deviation could be significant enough, especially if the interfering cell was larger, to dominate the signature waveform pattern and lower the classification accuracy. Nevertheless, we achieved a testing accuracy of 99% for non-interfering sensor waveforms, and 95% for interfering sensor waveforms. Furthermore, confusion matrices for the tests of non-interfering (Figure 6f) and interfering sensor waveforms (Figure 6e) did not present a misclassification bias for any specific sensor combination.

3.4.5 Testing of the Complete Algorithm

We completed the testing of our algorithm by cascading the RPN and the SCN. In this setting, each testing signal was first processed by the RPN, and the extracted signature waveforms were then classified by the SCN. The accuracy was calculated by comparing the total number of cells detected by each code-multiplexed Coulter sensor (sensor identity distribution) with the known number of each signature waveform in the testing data (Figure 6i, 6j). We achieved an accuracy of 97% for single cells and 85% for coincident cells. The overall testing accuracy for the cascaded ConvNets (i.e., the complete algorithm) was less than the calculated accuracy for a single ConvNet due to the propagation of the error. Specifically, the bounding-box estimation errors that occurred in the first stage (RPN), including occasional missing of low-power signature waveforms in interfering sensor waveforms, propagated to the second stage (SCN), resulting in reduced classification accuracy.

3.5 Computation Speed Test

The processing speed is an essential factor when evaluating an algorithm. To estimate the processing speed, we used each ConvNet to process 1000 input waveforms and recorded the unit processing time for each input (Figure 6d, 6h). We also calculated the cumulative time elapsed as each ConvNet processed those 1000 waveforms. As the RPN and the SCN shared the same structure (same number of parameters), they had similar processing speeds. On average, the RPN required ~610 ms, and the SCN required ~670 ms, to process 1000 input waveforms. Based on these metrics, the two-stage ConvNet structure could process 780 cells per second (2.7 GHz Intel Core i7, Intel). Processing speeds of this order can potentially make real-time analysis possible for a variety of sample types.

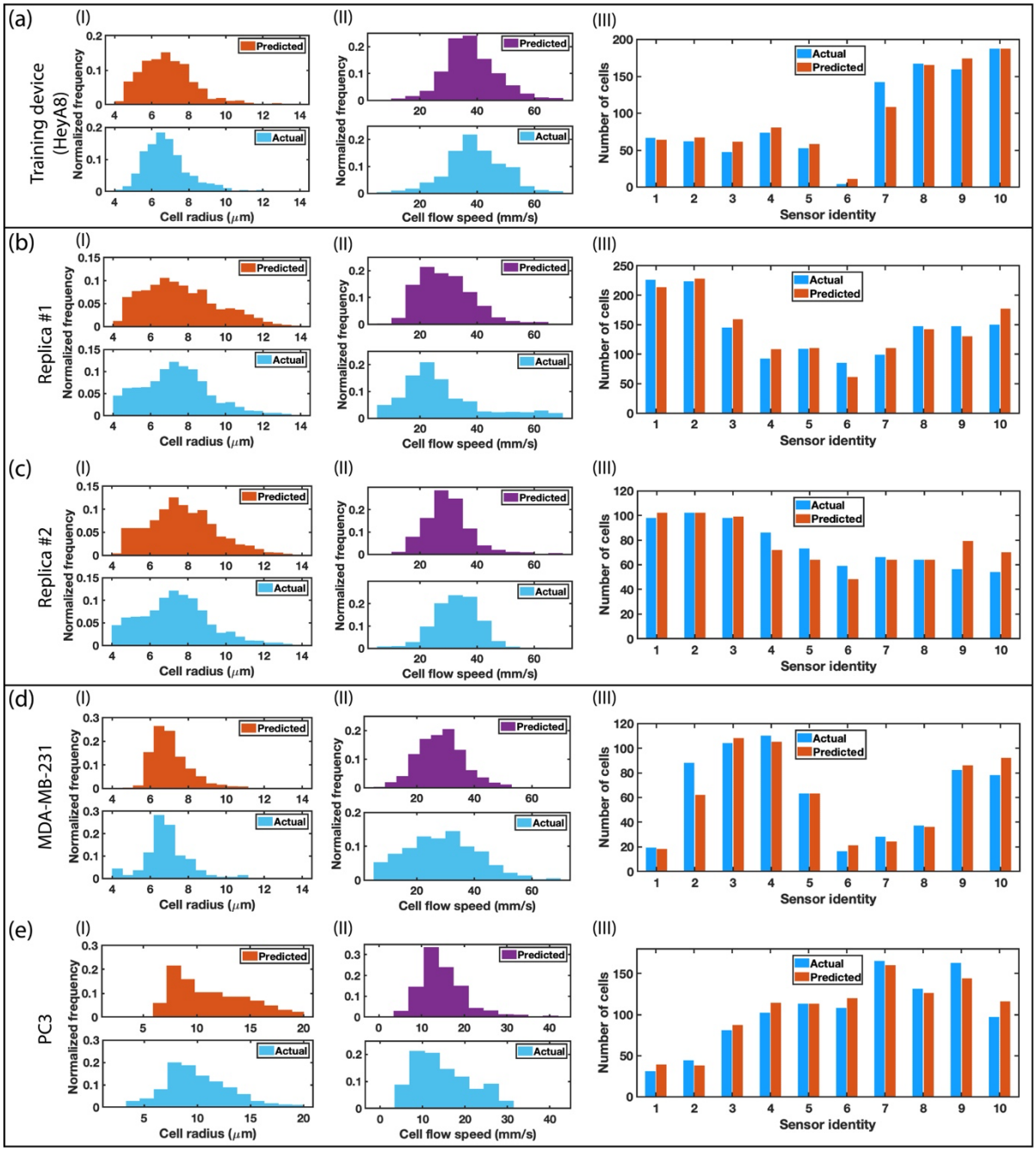

Figure 7: Cross-platform and cross-cell type benchmarking of the algorithm against optical imaging. (a) Testing of the algorithm accuracy on the training microfluidic device with HeyA8 cells. (i) Cell size and (ii) cell flow speed measurements by the algorithm (top) and microscopy (bottom). (iii) Sensor identity classification results shown in a histogram comparing algorithm and microscopy data for the number of cells received by each sensor. (b) Results from the same test (with HeyA8 cells) performed by processing the signals from another but identical microfluidic device (Replica #1) using the already-trained algorithm for cross-platform validation. (c) Cross-platform validation test results with HeyA8 cells from another device (Replica #2). (d) Test results from processing human breast cancer cells (MDA-MB-231) on a non-training microfluidic device for cross-cell type validation of our algorithm. (e) Results from the same test repeated using human prostate cancer cells (PC3) on another non-training microfluidic device.

3.6 Validation via Optical Imaging

To independently validate the performance of our algorithm, we compared the algorithm results with a simultaneously recorded high-speed (1000 fps) microscopy video footage of human cancer cells flowing through the microfluidic device. The video was recorded by placing all the sensors within the same field of view so that the whole sensor network activity can be visually acquired. By processing the recorded video of ~1000 cells by a custom-built image-processing program, speed and the sensor identity for each cell were automatically determined. Cell size distribution was obtained in a separate experiment by imaging cells of the same type and processing the recorded images with the ImageJ software. Microscope-measured cell size (Figure 7a) and cell speed (Figure 7b) histograms closely matched with the prediction. Besides algorithm-induced errors, the differences from optical measurements of cell properties are expected to be due to several factors: (1) the cells used for imaging might have had a different size distribution from the cells detected by the device even though they were sampled from the same tissue culture; (2) accuracy in cell size measurements might have suffered from calibration errors as well as the sensor-proximity effects in the microfluidic channel; (3) optical cell speed measurements with the high-speed camera are prone to errors from low spatial and temporal resolution. In terms of the sensor identity prediction, our algorithm was able to identify the correct sensor with an overall accuracy of 90.3% (Figure 7c). These results validated the ability of our algorithm to accurately capture the microfluidic activity of the cells and their characteristics.

3.7 Cross-platform Validation

To be of practical utility, trained ConvNets should be directly applicable to signals from other LoC devices with identical sensor designs. Furthermore, using the same device to generate both the training and testing signals might artificially enhance the measured accuracy of our algorithm. Therefore, we tested the cross-platform operability by training our algorithm on data from one device and testing its performance on other devices. For this purpose, we fabricated two microfluidic devices, which were replicas of the original device (the training device) we used in this study. Even though all the three devices had the same electrode design, their signature waveforms for each Coulter sensor were expected to show observable differences due to variations from the fabrication processes and the electrical contacts.

We processed ~1000 human ovarian cancer cells sampled from the same PBS suspension with each replica microfluidic device. High-speed microscopy videos were recorded as a benchmark to determine the cross-platform accuracy of our algorithm. Similar to Section 3.6, the videos were processed, and microscopy measurements were compared with the algorithm predictions for the cell size, cell flow speed, and the sensor identity. For both replica devices, the microscope-measured cell size (Figure 7d, 7g) and flow speed (Figure 7e, 7h) distributions matched closely with algorithm results, yielding similar mean and variance. As for sensor identities, we achieved 90.65% (Figure 7f), and 89.42% (Figure 7i) accuracy on Replica 1 and Replica 2, respectively. These accuracies were virtually the same with accuracy we achieved with the training microfluidic device in Section 3.6. Taken together, these results demonstrated the robustness of our trained ConvNets against cross-platform waveform variations, leading us to the conclusion that a pre-trained network could directly be used to interpret sensor signals from different microfluidic designs, as long as the same set of code sequences was used in the sensor network.

3.8 Cross-cell type Validation

To be used in a variety of applications, trained ConvNets should be directly applicable to signals generated by any cell types. Therefore, we tested the cross-cell type compatibility of our technique by applying our ConvNet, trained with human ovarian cancer cells (HeyA8) to interpret signals from processing of human breast (MDA-MB-231) and prostate (PC3) cancer cell lines. For these measurements, we fabricated two identical microfluidic devices (replicas of the training device) and separately processed the two cell lines on these devices. Simultaneously-recorded high-speed microscopy videos were treated as the ground truth to calculate the cross-cell type accuracy. For both cell lines, the microscope-measured cell size (Figure 7j, 7m) and flow speed (Figure 7k, 7n) distributions matched closely with algorithm results, yielding similar mean and variance. As for sensor identities, we achieved 89.76% (Figure 7l), and 91.11% (Figure 7o) accuracy on MDA-MB-231 and PC3, respectively. These results demonstrated the compatibility of trained ConvNets with different sample types and the potential of our technique for general-purpose cytometry applications.

4. Conclusion

Besides their conventional use for sizing and counting suspended particles, Coulter counters can be patterned to produce location-specific electrical waveforms and can therefore serve as sensor networks for tracking those particles. This additional layer of spatial information can successfully be extracted by processing the output signal via a deep learning-based algorithm that employs ConvNets. ConvNets are well suited for pattern recognition problems and can discriminate between non-correlated sensor waveforms with high accuracy. Moreover, ConvNets can be trained to recognize interference patterns of Coulter sensor waveforms to resolve data from coincident particles. Computationally, the pattern recognition process is efficient and can potentially enable real-time microfluidic assays for quantitative measurements on particle suspensions. Finally, an algorithm, trained on an instance of a Coulter sensor network, can perform equally well on different microfluidic devices equipped with an identical sensor network demonstrating that the presented approach can readily be employed for biomedical applications.

5. Acknowledgment

This work was supported by National Science Foundation, USA (Award no. ECCS 1752170), and the Arnold and Mabel Beckman Foundation, USA (Beckman Young Investigator Award to A.F.S.).

References

1 B. S. Bull, M. A. Schneiderman and G. Brecher, *Am. J. Cli. Pathol.*, 1965, **44**, 678-688.
2 G. T. Roberts and S. B. El Badawi, *Am. J. Cli. Pathol.*, 1985, **83**, 222-226.
3 E. Ginsburg and B. K. Vonderhaar, *Cancer Res.*, 1995, **55**, 2591-2595.
4 Saleem, M. Husheem, P. Härkönen and K. Pihlaja, *J. Ethnopharmacol.*, 2002, **81**, 327-336.
5 R. Duckworth, W. M. Brück, K. E. Janda, T. P. Pitts and P. J. McCarthy, *Mar. Bio. Res.*, 2006, **2**, 243-248.
6 H. E. Kubitschek, *Nature*, 1958, **182**, 234-235.
7 Poutrel and C. Lerondelle, *J. Dairy Sci.*, 1983, **66**, 2575-2579.

8 J. Wang, Y. Sun, X. Y. Meng, L. F. Li, Y. Li, Y. Luo, W. Wang, S. Yu, C. Yin, S. Li and H. J. Qiu, *Virus Res.*, 2018, **255**, 68-76.
9 W. Bult, S. G. C. Kroeze, M. Elschot, P. R. Seevinck, F. J. Beekman, H. W. A. M. de Jong, D. R. A. Uges, J. G. K. Kosterink, P. R. Luijten, W. E. Hennink, A. D. van het Schip, J. L. H. Ruud Bosch, J. F. W. Nijsen and J. J. M. Jans, *PLoS ONE*, 2013, **8**, e52178.
10 Nyström, J. Mazur, M. I. Barnett and M. Glazer, *J. Pharm. Pharmacol.*, 1985, *37*, 217-221.
11 S. M. Bezrukov, I. Vodyanoy and V. A. Parsegian, *Nature*, 1994, **370**, 279-281.
12 S. Lambert and M. Wagner, *Chemosphere*, 2016, **161**, 510-517.
13 M. M. Arimi, *Environ. Technol. Rev.*, 2018, **7**, 274-290.
14 C. Grulke, N. A. Marsh and B. A. Hills, *Br. J. Exp. Pathol.*, 1973, **54**, 684-691.
15 W. H. Coulter, *Proc. Natl. Electron. Conf.*, 1956, **12**, 1034-1040.
16 H. Bayley and C. R. Martin, *Chem. Rev.*, 2000, **100**, 2575-2594.
17 M. R. Kellman, F. R. Rivest, A. Pechacek, L. L. Sohn and M. Lustig, *IEEE Sens. J.*, 2018, **18**, 3068-3079.
18 Adamo, A. Sharei, L. Adamo, B. Lee, S. Mao and K. F. Jensen, *Anal. Chem.*, 2012, **84**, 6438-6443.
19 S. C. Bürgel, C. Escobedo, N. Haandbæk and A. Hierlemann, *Sens. Actuators, B*, 2015, **210**, 82-90.
20 De Ninno, V. Errico, F. R. Bertani, L. Businaro, P. Bisegna and F. Caselli, *Lab Chip*, 2017, **17**, 1158-1166.
21 T. E. Winkler, H. Ben-Yoav and R. Ghodssi, *Microfluid. Nanofluid.*, 2016, **20**, 134.
22 H. Song, J. M. Rosano, Y. Wang, C. J. Garson, B. Prabhakarpandian, K. Pant, G. J. Klarmann, A. Perantoni, L. M. Alvarez and E. Lai, *Anal. Methods*, 2016, **8**, 7437-7444.
23 J. Sun, J. Yang, Y. Gao, D. Xu and D. Li, *Microfluid. Nanofluid.*, 2017, **21**, 33.
24 O. A. Saleh and L. L. Sohn, *Nano Lett.*, 2003, **3**, 37-38.
25 T. Ito, L. Sun and R. M. Crooks, *Anal. Chem.*, 2003, **75**, 2399-2406.
26 Branton, D. W. Deamer, A. Marziali, H. Bayley, S. A. Benner, T. Butler, M. Di Ventra, S. Garaj, A. Hibbs, X. Huang, S. B. Jovanovich, P. S. Krstic, S. Lindsay, X. S. Ling, C. H. Mastrangelo, A. Meller, J. S. Oliver, Y. V. Pershin, J. M. Ramsey, R. Riehn, G. V. Soni, V. Tabard-Cossa, M. Wanunu, M. Wiggin and J. A. Schloss, *Nat. Biotechnol.*, 2008, **26**, 1146-1153.
27 W. Li, N. A. W. Bell, S. Hernández-Ainsa, V. V. Thacker, A. M. Thackray, R. Bujdoso and U. F. Keyser, *ACS Nano*, 2013, **7**, 4129-4134.
28 F. Liu, L. Ni and J. Zhe, *Biomicrofluidics*, 2018, **12**, 021501.
29 L. Zou, S. Li, Y. Kang, J. Liu, L. He, S. Sun, D. Gao, B. Qiu and W. Ding, *Biomed. Microdevices*, 2017, **19**, 30.
30 Civelekoglu, N. Wang, M. Boya, T. Ozkaya-Ahmadov, R. Liu and A. F. Sarioglu, *Lab Chip*, 2019, **19**, 2444-2455.
31 R. Liu, N. Wang, F. Kamili and A. F. Sarioglu, *Lab Chip*, 2016, **16**, 1350-1357.
32 R. Liu, W. Waheed, N. Wang, O. Civelekoglu, M. Boya, C. H. Chu and A. F. Sarioglu, *Lab Chip*, 2017, **17**, 2650–2666.
33 C. Wyatt Shields Iv, C. D. Reyes and G. P. López, *Lab Chip*, 2015, **15**, 1230-1249.
34 A. Adamo, A. Sharei, L. Adamo, B. Lee, S. Mao and K. F. Jensen, *Anal. Chem*., 2012, **84**, 6438- 6443.

35 K. R. Balakrishnan, J. C. Whang, R. Hwang, J. H. Hack, L. A. Godley and L. L. Sohn, *Anal. Chem*., 2015, **87**, 2988-2995.
36 R. Liu, C. H. Chu, N. Wang and A. F. Sarioglu. *Proceedings of uTAS (The 22nd International Conference on Miniaturized Systems for Chemistry and Life Sciences), November 11-15*, 2018, Kaohsiung, Taiwan, 1248-1250.
37 Civelekoglu, R. Liu, M. Boya, C. H. Chu, N. Wang and A. F. Sarioglu, *Proceedings of 19th International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS), June 18-22*, 2017, Kaohsiung, Taiwan, 480-483.
38 N. Asmare, AKM Arifuzzman, M. Boya, N. Wang, R. Liu, C. H. Chu and A. F. Sarioglu, *Proceedings of uTAS (The 22nd International Conference on Miniaturized Systems for Chemistry and Life Sciences), November 11-15*, 2018, Kaohsiung, Taiwan, 1308-1310.
39 M. A. Abu-Rgheff, *Introduction to CDMA Wireless Communications*, Academic Press, Oxford, 2007.
40 R. Rao and S. Dianat, *Basics of Code Division Multiple Access (CDMA)*, SPIE, Bellingham, WA, 2005.
41 R. Gold, *IEEE Trans. Inf. Theory*, 1967, **13**, 619-621.
42 R. Gold, *IEEE Trans. Inf. Theory*, 1968, **14**, 154-156.
43 R. Liu, N. Wang, N. Asmare and A. F. Sarioglu, *Biosens. Bioelectron.*, 2018, **120**, 30-39.
44 W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu and A. C. Berg, *European Conference on Computer Vision*, 2016, 21-37.
45 M. Chen, Y. Hao, K. Hwang, L. Wang and L. Wang, *IEEE Access*, 2017, **5**, 8869-8879.
46 Y. LeCun, Y. Bengio and G. Hinton, *Nature*, 2015, **521**, 436-444.
47 Ignatov, *Appl. Soft Comput.*, 2018, **62**, 915-922.
48 S. Kiranyaz, T. Ince and M. Gabbouj, *IEEE Trans. Biomed. Eng.*, 2015, **63**, 664-675.
49 Y. Zhang, W. Chan and N. Jaitly, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, 4845-4849.
50 Krizhevsky, I. Sutskever and G. E. Hinton, *Commun. ACM*, 2017, **60**, 84-90.
51 N. Wang, R. Liu and A. F. Sarioglu, *J. Visualized Exp.*, 2017, **121**, e55311.
52 P. Murugan, arXiv: 1801.01397, 2018.
53 D. R. Jones, M. Schonlau and W. J. Welch, *J. Global Optim.*, 1998, **13**, 455-492.
54 D. A. van Dyk and X. L. Meng, *J. Comput. Graph. Stat.*, 2001, **10**, 1-50.
55 I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*, MIT Press, Cambridge, MA, USA, 2016.
56 R. W. DeBlois and C. P. Bean, *Rev. Sci. Instrum.*, 1970, **41**, 909-916.