Supplementary Material to:

Deciphering impedance cytometry signals with neural networks

Federica Caselli,*[a] Riccardo Reale,[b] Adele De Ninno,[c] Daniel Spencer,[d] Hywel Morgan [d] and Paolo Bisegna [a]

[a] *Department of Civil Engineering and Computer Science, University University of Rome Tor Vergata, Rome, Italy. E-mail: caselli@ing.uniroma2.it*

[b] *Center for Life Nano Science@Sapienza, Italian Institute of Technology (IIT), Rome, Italy.*

[c] *Italian National Research Council - Institute for Photonics and Nanotechnologies (CNR - IFN), Rome, Italy.*

[d] *School of Electronics and Computing Science, and Institute for Life Sciences, University of Southampton, Highfield, Southampton, UK.*

Lab Chip, 2022, DOI: 10.1039/D2LC00028H

Section S1: Basics of single-cell dielectric spectroscopy

The frequency-dependent impedance of a single cell conveys information on cell size and cell dielectric properties. However, it also includes effects from the measurement hardware (non-linear behaviour of the electronics and chip parasitics), which need to be removed. As detailed in Ref.[1], the cell impedance is normalised against the bead impedance according to the following relationship:

$$\tilde{f}_{CM}^{cell}\left(\frac{r^{cell}}{r^{bead}}\right)^3 = \tilde{f}_{CM}^{bead}\,\frac{S^{cell}}{S^{bead}} \qquad (1)$$

where $\tilde{f}_{CM}^{cell}$and $\tilde{f}_{CM}^{bead}$ denote the complex, frequency-dependent Clausius-Mossotti factor of cell and bead, respectively; r^{cell} and r^{bead} denote cell and bead radius, respectively; S^{cell} and S^{bead} denote the complex, frequency-dependent peak amplitude of the measured differential impedance signals of cell and bead. Here $\tilde{f}_{CM}^{bead}$ and r^{bead} are regarded as known quantities, whereas $\tilde{f}_{CM}^{cell}$ and r^{cell} are unknowns to be determined. The Clausius-Mossotti factor of the cell is given by:

$$\tilde{f}_{CM}^{cell} = \frac{\tilde{\varepsilon}_{cell}-\tilde{\varepsilon}_{med}}{\tilde{\varepsilon}_{cell}+2\tilde{\varepsilon}_{med}} \qquad (2)$$

where $\tilde{\varepsilon}_{med}$ is the complex permittivity of the medium and $\tilde{\varepsilon}_{cell}$ is the complex permittivity of the cell, which incorporates all the key information regarding the cell dielectric properties. Assuming a single-shell model, $\tilde{\varepsilon}_{cell}$ can be derived from[2]:

$$\tilde{\varepsilon}_{cell} = \tilde{\varepsilon}_{mem}\,\frac{\gamma^3+2\left(\frac{\tilde{\varepsilon}_{cyt}-\tilde{\varepsilon}_{mem}}{\tilde{\varepsilon}_{cyt}+2\tilde{\varepsilon}_{mem}}\right)}{\gamma^3-\left(\frac{\tilde{\varepsilon}_{cyt}-\tilde{\varepsilon}_{mem}}{\tilde{\varepsilon}_{cyt}+2\tilde{\varepsilon}_{mem}}\right)}\ , \quad \gamma = \frac{r}{r-d_{mem}} \qquad (3)$$

where $\tilde{\varepsilon}_{mem}$ and $\tilde{\varepsilon}_{cyt}$ are the complex permittivities of the membrane and the cytoplasm, respectively, d_{mem} is the membrane thickness, and r is the cell radius. Under the assumption $d_{mem} \ll r$ it simplifies to[3]:

$$\tilde{\varepsilon}_{cell} \cong \tilde{\varepsilon}_{cyt}\,\frac{\lambda}{1+\lambda}\ , \qquad \lambda = \frac{\tilde{\varepsilon}_{mem}/d_{mem}}{\tilde{\varepsilon}_{cyt}/r} \qquad (4)$$

The complex permittivities of the membrane and the cytoplasm can, in turn, be calculated using:

$$\tilde{\varepsilon}_{mem} = \varepsilon_{mem} + \sigma_{mem}/j\omega,\ \ \tilde{\varepsilon}_{cyt} = \varepsilon_{cyt} + \sigma_{cyt}/j\omega \qquad (5)$$

where ε and σ are permittivities and conductivities for each component, respectively, and ω is the angular frequency. The specific membrane capacitance is given by $C_{mem} = \varepsilon_{mem}/d_{mem}$. Fig. S1 illustrates how the Clausius-Mossotti factor, and hence the impedance spectrum, changes following small perturbations in cell electrical properties (shaded). Because the cell membrane has a very high electrical resistance, at low AC frequencies (<1MHz), cells behave as insulating particles ($\tilde{f}_{CM}^{cell} \cong \tilde{f}_{CM}^{bead}$). Therefore low-frequency impedance measurements provide a direct measurement of (electrical) cell volume.

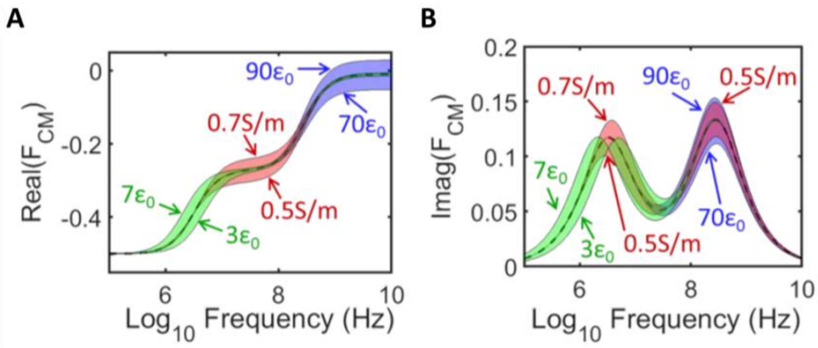

Fig. S1 Calculated examples of the (A) Real and (B) Imaginary parts of the Clausius-Mossotti factor of a spherical red blood cell (radius=2.75 µm, membrane thickness 4.5 nm, cytoplasm conductivity = 0.6 S/m, membrane permittivity = 4.5 ε$_0$, cytoplasm permittivity = 80 ε$_0$) suspended in phosphate-buffered saline (conductivity = 1.6 S/m). The shaded regions correspond to three frequency windows where differences in the cell electrical properties can be measured, corresponding to: membrane capacitance (green, 3-7 ε$_0$), cytoplasm conductivity (red, 0.5-0.7 S/m) and cytoplasm permittivity (blue, 70-90 ε$_0$). Reprinted with permission from Ref.[1]

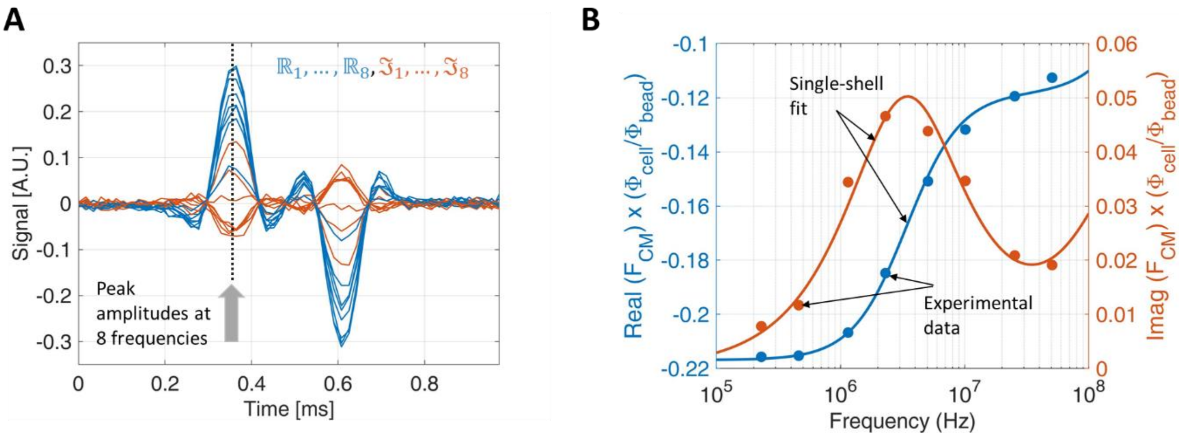

Fig. S2 Standard approach to single-cell dielectric characterization. (A) The complex peak amplitudes at eight frequencies are extracted from the measured impedance traces (e.g., by peak-finding or correlation with a bipolar Gaussian template). (B) According to the right-hand side of eq.(1), the measured complex peak amplitudes (S^{cell}) normalised with respect to the bead population (S^{bead}) and multiplied by $\tilde{f}_{CM}^{bead} = -1/2$ provide the experimental impedance spectrum (i.e. the real and imaginary parts of the Clausius-Mossotti factor $\tilde{f}_{CM}^{cell}$ multiplied by the ratio between cell and bead volume fractions, $\frac{\phi^{cell}}{\phi^{bead}} = \left(\frac{r^{cell}}{r^{bead}}\right)^3$, cf. left-hand side of eq. (1)). The single-shell model (eqs. (2)-(5)) is used to represent the Clausius-Mossotti factor $\tilde{f}_{CM}^{cell}$. Accordingly, least-square fitting of the experimental impedance spectrum yields the cell radius (r) and the cell intrinsic dielectric properties, namely, membrane capacitance C_{mem}, cytoplasm conductivity σ_{cyt} and cytoplasm permittivity ε_{cyt}.

Section S2: Data augmentation to mimic velocity changes

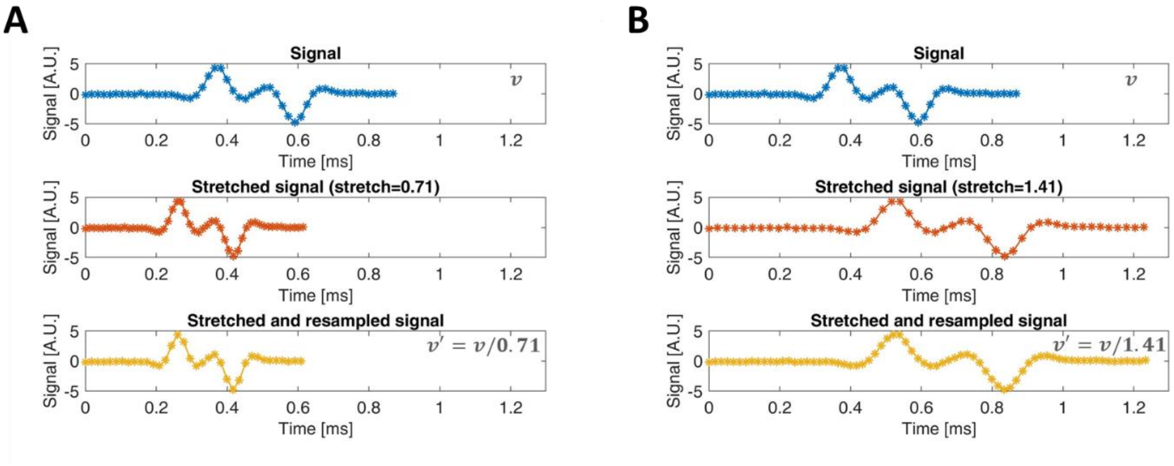

Fig. S3 Two examples of signal stretching and resampling at original sampling rate: (A) stretch=0.71, (B) stretch=1.41. Denoting by v the original particle velocity, the resulting particle velocities are $v' = v/0.71$ and $v' = v/1.41$, respectively.

Section S3: Neural networks for single-cell dielectric spectroscopy from raw impedance data streams

Figures S4 and S5 show details of the neural networks used for single-cell dielectric spectroscopy. The software was implemented in Matlab (R2019b). Default settings were used for parameters that are not explicitly mentioned. Computations were run on a processor Intel(R) Xeon(R) CPU E5-2660 v3 @ 2.60GHz (128 GB RAM) with Nvidia Titan X GPU.

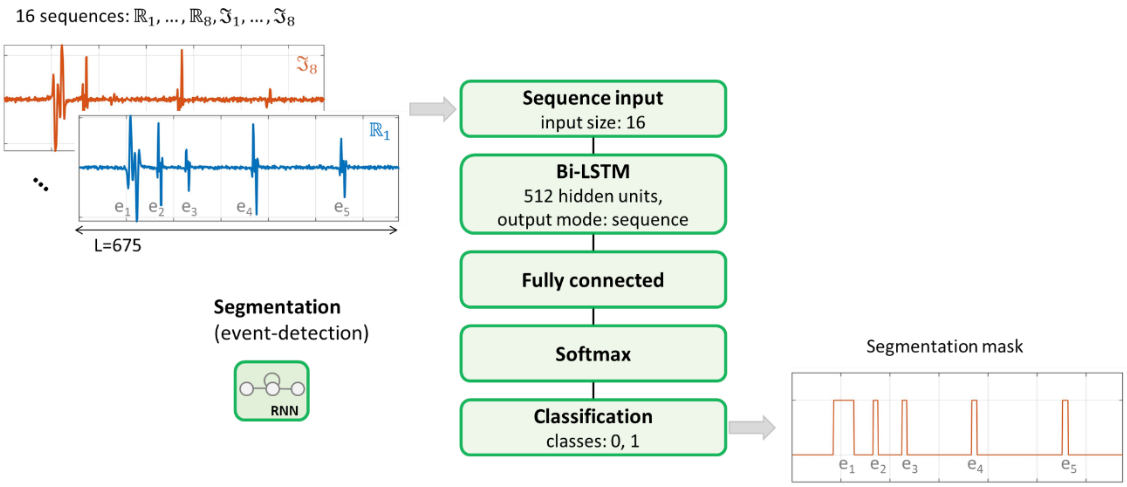

Fig. S4 Architecture of the classification Recurrent Neural Network (RNN) used for data stream segmentation (bi-LSTM = bidirectional Long Short-Term Memory). The data stream (~250 s, sampled at f_s = 57.6 kHz; 8 frequency channels) was split into ~20k portions of length L=675 samples (~12 ms). These portions were randomly divided into the training set (70%), validation set (15%), and testing set (15%). Each portion is composed of 16 sequences (real part $\mathbb{R}$ and imaginary part $\mathfrak{I}$, at each frequency), which are the input of the network. The final classification layer outputs a binary sequence wherein 0 indicates noise and 1 indicates particle signal, thus yielding the data stream segmentation mask. Training parameters were: 16 epochs, 128 mini-batch size. Data stream splitting may result in partially caught events at the beginning or at the end of a portion. Accordingly, detected events with length shorter than a threshold l_{min} were discarded (l_{min} was set to half of the median peak-to-peak time).

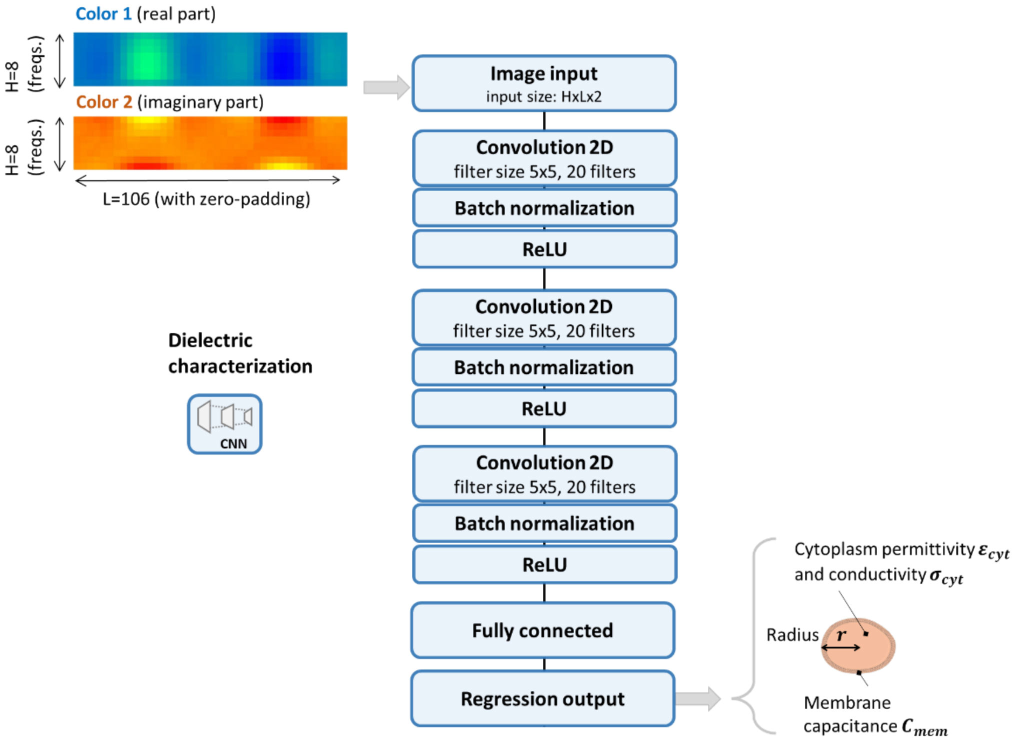

Fig. S5 Architecture of the regression Convolutional Neural Network (CNN) used for dielectric characterization (ReLU = Rectifier Linear Unit). For each detected cell, the network receives in input the 2-colour impedance images (i.e., the rows correspond to the eight frequency channels, the columns correspond to the different time samples, and the two colours are the real and imaginary signal components). The set of impedance images (N>20k) was randomly divided into the training set (70%), validation set (15%), and testing set (15%). The final regression layer predicts the cell size (radius, r) and the cell intrinsic dielectric properties, namely, membrane capacitance C_{mem}, cytoplasm conductivity σ_{cyt} and cytoplasm permittivity ε_{cyt}. Training parameters were: 512 epochs, 128 mini-batch size. The two CNNs used for system normalization (i.e., classification beads/others, and regression on bead amplitudes) share a similar architecture. In the classification network, the regression layer is replaced by a softmax layer followed by a classification layer.

Box S1 shows, as an example, the Matlab code for training and test of the regression CNN used for dielectric spectroscopy (cf. Fig. S5). Network training calls the routine **CNN_regression_dielectric_characterization.m**, whose script is shown in Box S2. Network test calls Matlab's built-in function **predict.m**.

Box S1

```
% Inputs ----------------------------------------------------------------
%
% H=8; % number of frequencies
% L=106; % maximum event length (sample), typical
% C=2; % number of colours (real and imaginary parts)
% inputSize=[H, L, C];
% filterSize=[5, 5];
% numFilters=20;
% numOutput=4; % number of features to be estimated
%
% Acquired signals (4-D: frequency, sample, real/imag, event)
% XTrain:      4-D double with size [H, L, C, NTrain]
% XValidation: 4-D double with size [H, L, C, NValidation]
% XTest:       4-D double with size [H, L, C, NTest]
% Target features (2-D: event, feature)
% YTrain:      2-D double with size [NTrain,      numOutput]
% YValidation: 2-D double with size [NValidation, numOutput]
% YTest:       2-D double with size [NTest,       numOutput]
%
% NTrain     =19k; % cardinality of train dataset, typical
% NValidation= 4k; % cardinality of validation dataset, typical
% NTest      = 4k; % cardinality of test dataset, typical
%
% values of learning parameters, typical
% maxEpochs=512;
% miniBatchSize=128;
% GradientThreshold=1;
%
% Outputs ---------------------------------------------------------------
%
% net:             trained neural network
% info:            training information
% training_time:   training time
%
% YPred:           2-D double with size [NTest, numOutput]
% prediction_time: prediction time

% Build and train network
[ ...
    ... % neural network, relevant information, training time
    net, info, training_time]=CNN_regression_dielectric_characterization( ...
    ... % training and validation datasets
    XTrain, YTrain, XValidation, YValidation, ...
    ... % layer parameters
    inputSize, filterSize, numFilters, numOutput, ...
    ... % learning parameters
    maxEpochs, miniBatchSize, GradientThreshold);

% Test trained network
t_start=tic;
YPred = predict(net,XTest, ...
    'MiniBatchSize',miniBatchSize, ...
    'SequenceLength','longest' ...
    );
prediction_time=toc(t_start);
```

Box S2

```
function [ ...
    ... % neural network, relevant information, training time
    net, info, training_time]=CNN_regression_dielectric_characterization( ...
    ... % training and validation datasets
    XTrain, YTrain, XValidation, YValidation, ...
    ... % layer parameters
    inputSize, filterSize, numFilters, numOutput, ...
    ... % learning parameters
    maxEpochs, miniBatchSize, GradientThreshold)

% -------------------------------------------------------------------------
% define network
% -------------------------------------------------------------------------

layers = [
    imageInputLayer(inputSize)

    convolution2dLayer(filterSize,numFilters,'Padding','same')
    batchNormalizationLayer
    reluLayer

    convolution2dLayer(filterSize,numFilters,'Padding','same')
    batchNormalizationLayer
    reluLayer

    convolution2dLayer(filterSize,numFilters,'Padding','same')
    batchNormalizationLayer
    reluLayer

    fullyConnectedLayer(numOutput,'Name','fc')
    regressionLayer('Name','regressionoutput')];

% validation frequency
validationFrequency = floor(numel(YTrain)/miniBatchSize);

% training options
options = trainingOptions('adam', ...
    'ExecutionEnvironment','auto', ...
    'GradientThreshold',GradientThreshold, ...
    'MaxEpochs',maxEpochs, ...
    'MiniBatchSize',miniBatchSize, ...
    'SequenceLength','longest', ...
    'Shuffle','every-epoch', ...
    'VerboseFrequency',validationFrequency, ...
    'ValidationData',{XValidation,YValidation}, ...
    'ValidationFrequency',validationFrequency, ...
    'Verbose',false, ...
    'Plots','training-progress');

% -------------------------------------------------------------------------
% train network
% -------------------------------------------------------------------------
t_start=tic;
[net,info] = trainNetwork(XTrain,YTrain,layers,options);
training_time=toc(t_start);

end
```

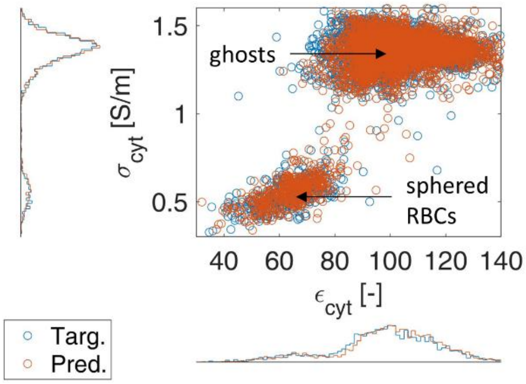

Fig. S6 Scatter plot of cytoplasm conductivity σ_{cyt} vs cytoplasm permittivity ε_{cyt}, along with their marginal histograms. Target (Targ.) features and predicted (Pred.) features are shown in blue and red, respectively. The two separate populations of ghosts and sphered RBCs are indicated.

Section S4: System normalization results

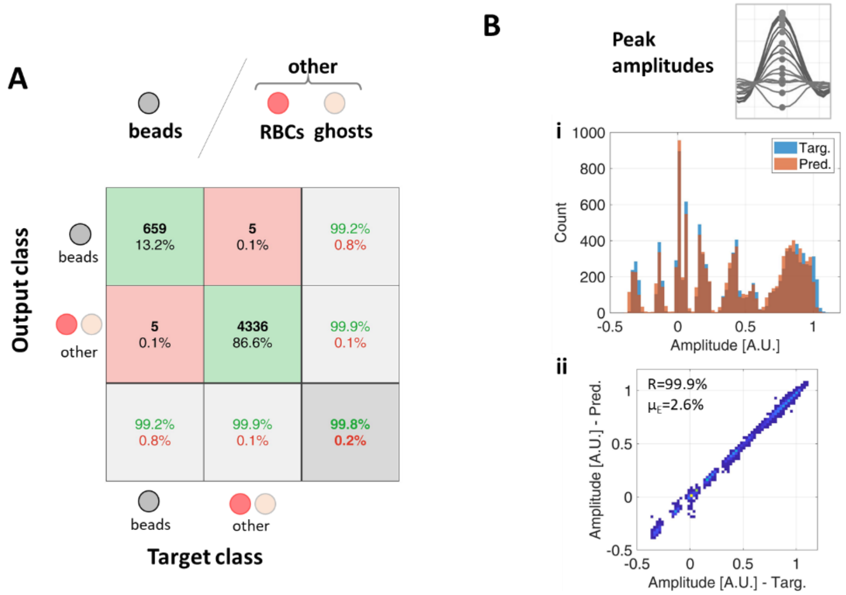

Fig. S7 (A) Confusion matrix for beads/non-beads classification. The rows correspond to the predicted class (Output Class), and the columns correspond to the true class (Target Class). Both the number of observations and the percentage of the total number of observations are shown in each cell. Normalizations by row (i.e., precision) and by column (i.e., recall) are also reported. (B) Histogram and density plot of target (Targ.) and predicted (Pred.) amplitudes (real and imaginary parts of the peak amplitudes of the eight frequency channels are collected). Correlation coefficient R and average relative errors μ_E are also reported within the density plot.

Section S5: Example of a triplet of red blood cells

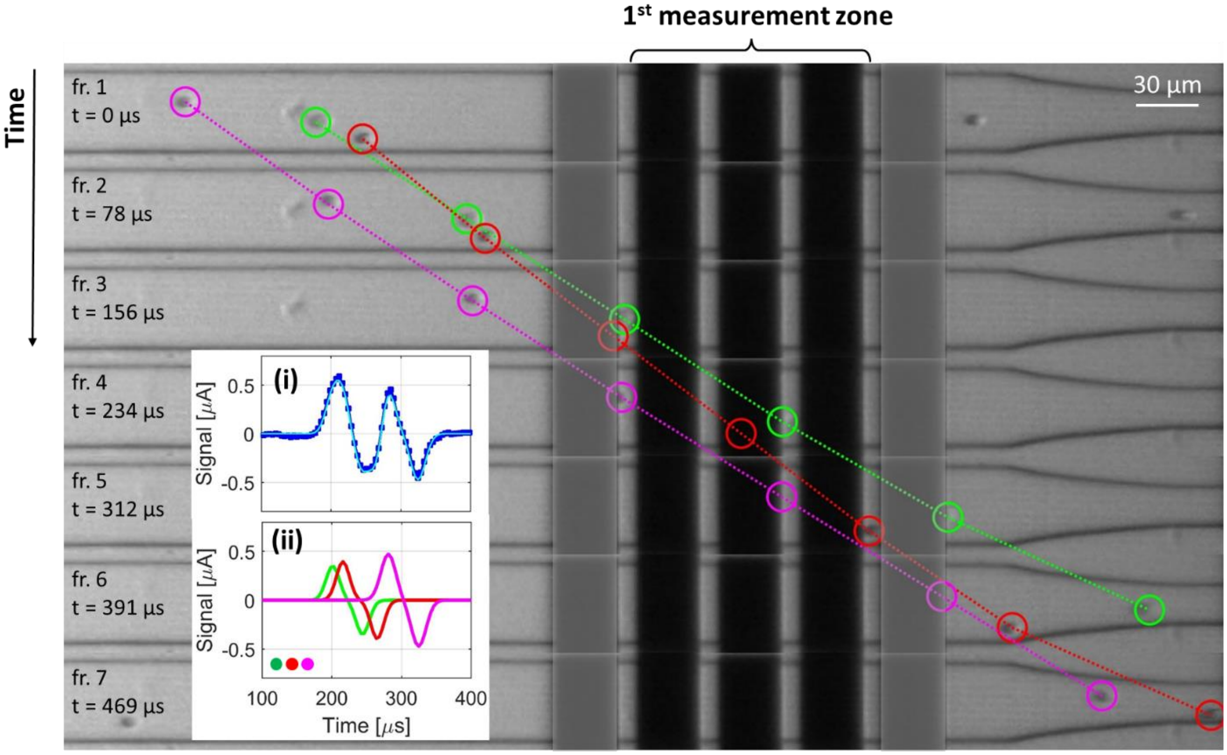

Fig. S8 Example of an event involving three red blood cells flowing through the first measurement zone (cf. Fig. 3A, main text). Flow direction is from left to right. Seven sequential frames (fr.) are shown (78 µs time interval), from top to bottom. Successive positions of every single cell are marked with a circle and linked with a dotted line. Unused lateral electrodes have been shadowed. Inset (i) is the measured signal $s^1(t)$. Inset (ii) is the composing single-particle signals (colours after circle markers). The cell marked in red and the cell marked in green almost simultaneously enter the measurement zone (fr. 3). However, the former is slower than the latter and leaves the measurement zone well after the cell marked in magenta has entered. Therefore, the signals produced by single particles interfere with each other, and only two pairs of positive/negative peaks appear in the recorded signal. Adapted with permission from Ref.[4]

Section S6: Generation of synthetic clusters (i.e., multi-particle signals)

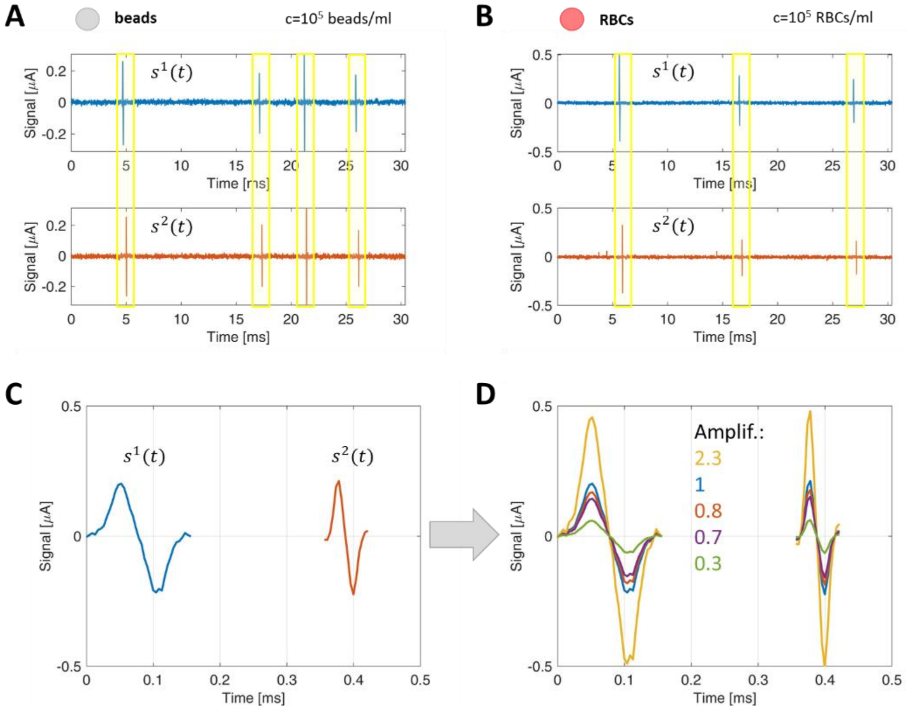

Fig. S9 Single-particle signals $s^1(t)$ and $s^2(t)$ are extracted from experimental data streams relevant to (A) beads or (B) RBCs at low concentration (c=10^5 particle/ml). More than 40000 single particles (equal proportion of beads and RBCs) are collected to build the training dataset. Each pair of single-particle signals ($s^1(t)$ and $s^2(t)$) is subjected to amplitude data augmentation, with five values of amplification factor. The logarithm of the amplification factor is randomly uniformly distributed in [0.25,4]. An example is shown in panel (C), original signals, and panel (D), augmented signals.

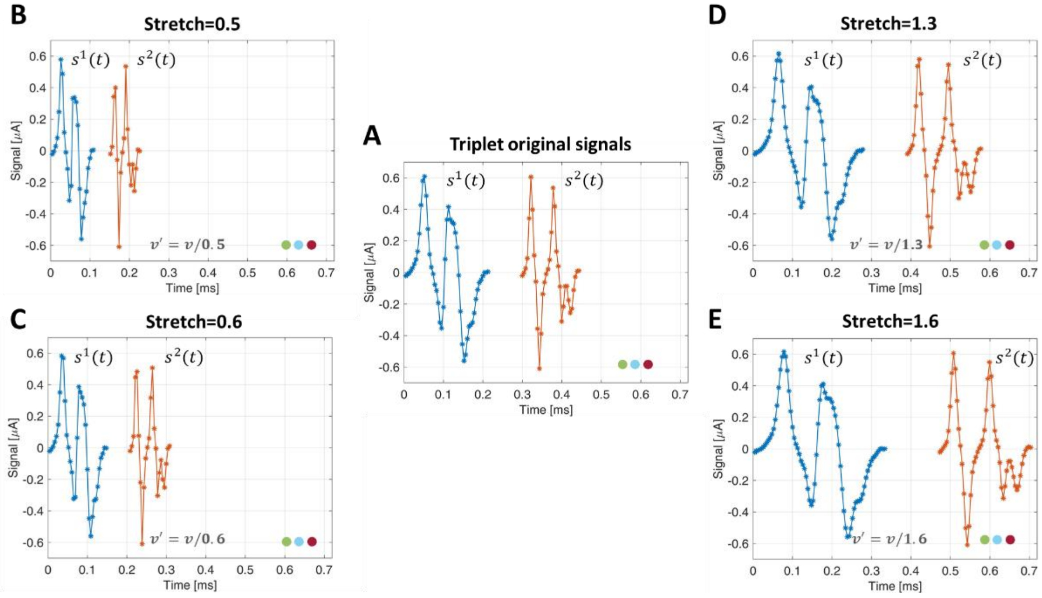

Fig. S10 Panel (A) shows the signals $s^1(t)$ and $s^2(t)$ of an exemplary synthetic triplet. Panels (B)-(E) show the stretched and resampled versions, along with the corresponding stretch values. The resulting particle velocities $v'$are obtained by dividing the original velocity v by the stretch, as indicated in panels (B)-(E).

Section S7: Neural networks coincidence resolution from two electrical snapshots

Figures S11 and S12 show details of the neural networks used for coincidence resolution. As for Section S3, the software was implemented in Matlab; default settings were used for parameters that are not explicitly mentioned, and a processor Intel(R) Xeon(R) CPU E5-2660 v3 @ 2.60GHz (128 GB RAM) with Nvidia Titan X GPU was used.

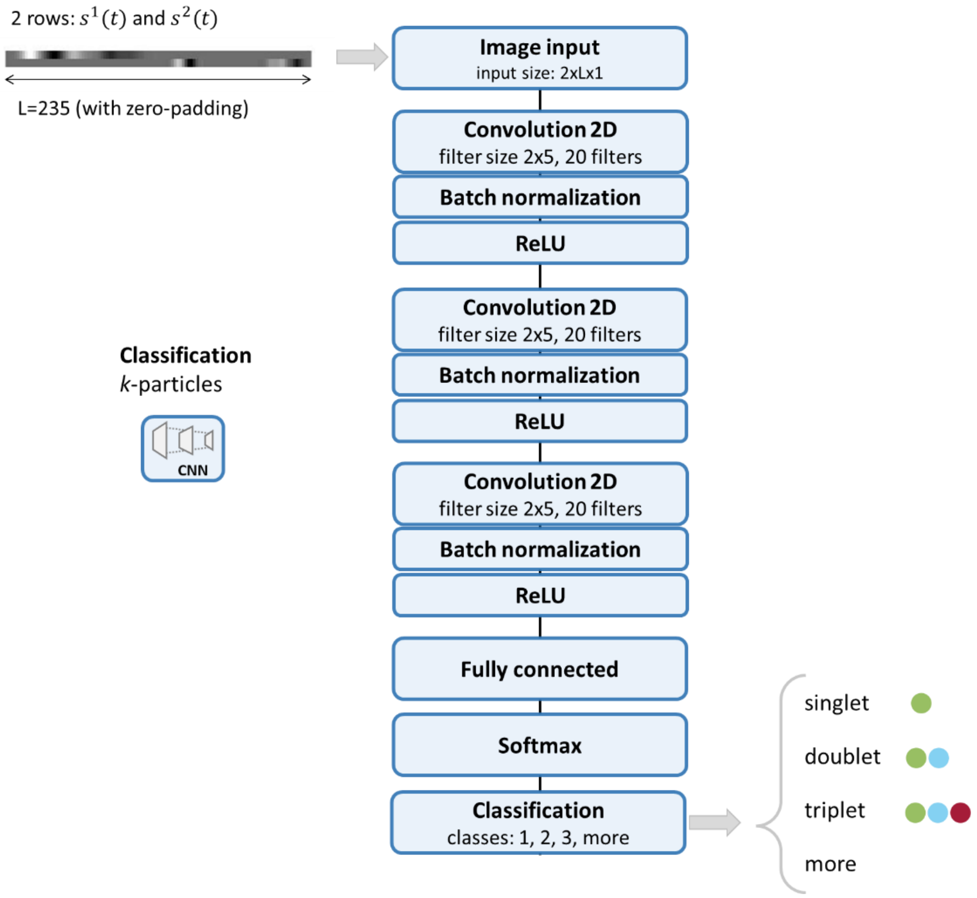

Fig. S11 Architecture of the classification CNN used for particle counting (i.e., identification of the number of individual composing particles). The network receives in input an image with two rows (the signals $s^1(t)$ and $s^2(t)$ relevant to the first and second measuring zones, respectively). The final classification layer predicts the number of involved particles: 1 (singlet), 2 (doublet), 3 (triplet), or more. Training parameters were: 64 epochs, 128 mini-batch size.

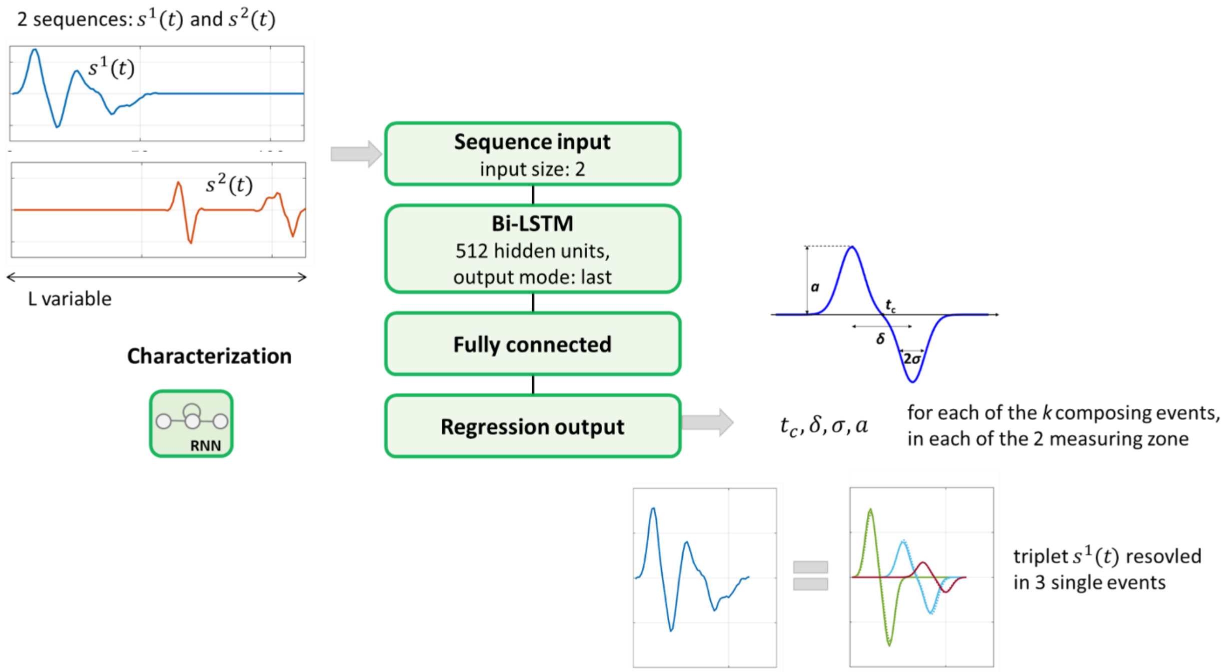

Fig. S12 Architecture of the regression RNNs used to characterise individual composing particles (RNN-s, for singlets; RNN-d, for doublets; RNN-t, for triplets; cf. Fig. 3B, main text). The networks receive in input two sequences (the signals $s^1(t)$ and $s^2(t)$ relevant to the first and second measuring zones, respectively). The final regression layer predicts the features (central time t_c, peal-to-peak time δ, pulse width σ, and peak amplitude a) of each composing event (1 for RNN-s, 2 for RNN-d, and 3 for RNN-t), in both measuring zones. Training parameters were: 64 epochs for RNN-s, 256 epochs for RNN-d, 512 epochs for RNN-t, and 128 mini-batch size (all networks).

Section S8: Coincidence resolution – density plots

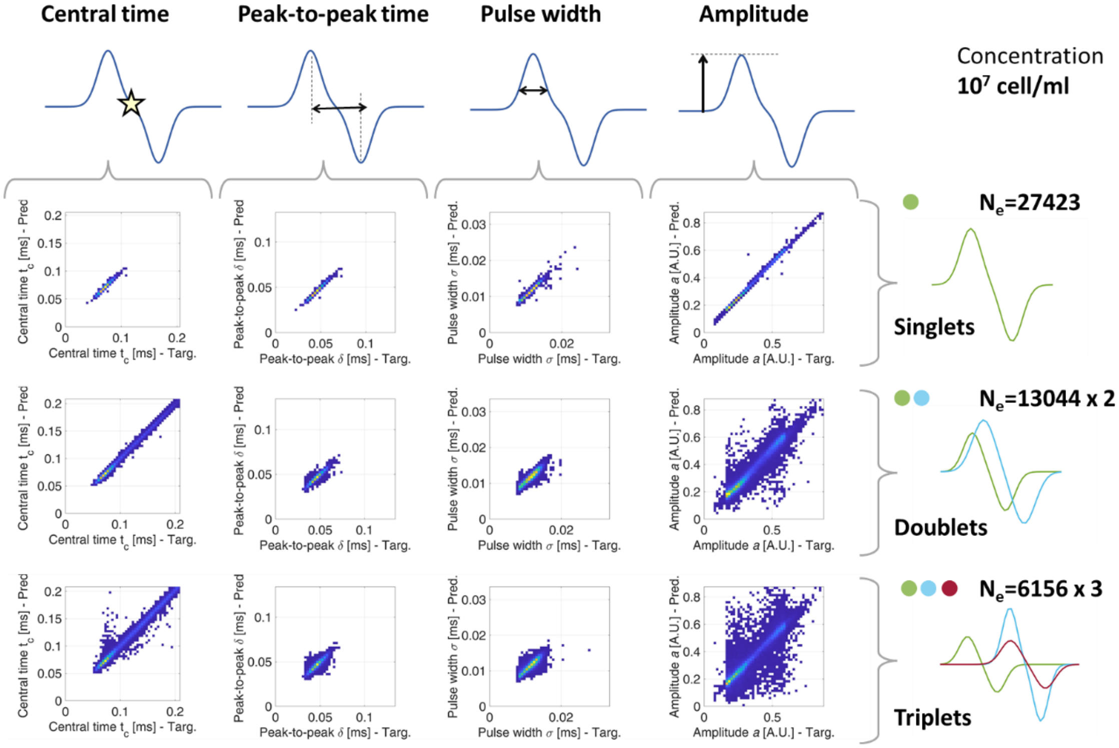

Fig. S13 Density plots of target (Targ.) and predicted (Pred.) features (i.e., central time, peak-to-peak time, pulse width, and amplitude) of the single-particle signals. The collection of single-particle features from singlets, doublets, and triplets are plotted on separate rows (from top to bottom).

References

1 D. Spencer and H. Morgan, *ACS Sensors*, 2020, **5**, 423–430.

2 H. Morgan, T. Sun, D. Holmes, S. Gawad and N. G. Green, *J. Phys. D Appl. Phys.*, 2007, **40**, 61–70.

3 C. Honrado, P. Bisegna, N. S. Swami and F. Caselli, *Lab Chip*, 2021, **21**, 22–54.

4 F. Caselli, A. De Ninno, R. Reale, L. Businaro and P. Bisegna, *IEEE Trans Biomed Eng*, 2020, **68**, 340–349.