

Microfluidic integrated DNA neural computing

Mengyao Cao¹, Na Li¹, Xiewei Xiong^{1*}, Sichen Xing¹, Jianing Wang¹, Bowen Li¹, Yun Zhu¹, Tong Zhu¹, Fei Wang², Siyi

Hu⁴, Hanbin Ma⁵, Longqian Xu⁵, Chunyu Chang⁶, Jiajian Ji⁴, Qunyan Yao⁷, Li Li¹, Hao Pei^{1*}, Chunhai Fan^{2,3*}

¹Shanghai Key Laboratory of Green Chemistry and Chemical Processes; Shanghai Center of Brain-inspired Intelligent Materials and Devices; Shanghai Frontiers Science Center of Molecule Intelligent Syntheses; School of Chemistry and Molecular Engineering, East China Normal University; 500 Dongchuan Road, Shanghai 200241, China

² State Key Laboratory of Synergistic Chem-Bio Synthesis, School of Chemistry and Chemical Engineering, New Cornerstone Science Laboratory, Frontiers Science Center for Transformative Molecules, Zhang Jiang Institute for Advanced Study and National Center for Translational Medicine, Shanghai Jiao Tong University, Shanghai 200240, China

³Renji Branch of National Center for Translational Medicine, Institute of Molecular Medicine, Shanghai Key Laboratory for Nucleic Acids Chemistry and Nanomedicine, Renji Hospital, School of Medicine, Shanghai Jiao Tong University, Shanghai 200127, China

⁴Suzhou Institute of Biomedical Engineering Technology, Chinese Academy of Sciences, 88 Keling Road, Suzhou, 215163, China

⁵School of Medicine, University of Electronic Science and Technology of China, Chengdu, 611731, China

⁶College of Computer Science and Technology, Jilin University, Changchun, 130012, China

⁷Department of Gastroenterology and Hepatology, Zhongshan Hospital, Fudan University, Shanghai 200032, P.R. China

*Corresponding authors. Email: xwxiong@chem.ecnu.edu.cn; peihao@chem.ecnu.edu.cn; fanchunhai@sjtu.edu.cn;

Contents

S1: Materials and Methods.....	2
S2: Design Details of the DMIC	14
S3: Design Details of the On-chip DNA Neuron	20
S4: Integration of Multiple DNA Neurons on a DMIC	29
S5: Demonstration of Supervised Learning	48
S6: Demonstration of Clinical Diagnostic Application.....	64
References	75

S1: Materials and Methods

Materials. All DNA strands and RNA strands were ordered from Sangon Biotech and Genewiz, Inc. (Suzhou, China). All DNA and RNA sequences used here are indicated in Supplementary table 1. Unmodified DNA oligonucleotides were purified by ultra-polyacrylamide gel electrophoresis (ULTRA PAGE) by Sangon, except for the purchased ssDNA and cT DNA strands used in Figs. S48-S49 and Fig. S51 that were purified by h-polyacrylamide gel electrophoresis (hPAGE) by Genewiz. The DNA strands modified with fluorophores were purified by high-performance liquid chromatography (HPLC) by Sangon. The DNA strands used in Fig. 4 were purified by t-polyacrylamide gel electrophoresis (tPAGE) by Genewiz. The synthetic RNA strands were purified under RNase-free conditions by HPLC by Sangon. All DNA strands were dissolved at 100 μ M in 1 $\times$ Tris-EDTA (TE) buffer (pH 8.0) with 12.5 mM Mg^{2+} and stored at 4 $^{\circ}$ C. DNA templates, used for the amplification and transformation, were resuspended at 100 μ M in 1 $\times$ TE buffer with 12.5 mM Mg^{2+} (nuclease free, pH 8.0) and stored at 4 $^{\circ}$ C. All RNA strands were dissolved at 20 μ M in nuclease-free water at -20° C until needed. Vent(exo-) DNA polymerase (M0257) and Nt.BstNBI nicking enzymes (R0607) were purchased from New England Biolabs (NEB) and stored at -20° C. Reverse transcription kit (miRNA First Strand cDNA Synthesis (Tailing Reaction) B532451) and excess primer (Universal PCR Primer R (10 μ M), B661601) for LATE-PCR were purchased from Sangon Biotech. RNA isolation kit was purchased from HYCEZBIO (cat #: RHYC71, Wuhan, China). Gel extraction kit (Poly-Gel DNA Extraction Kit D1250) was purchased from Solabio. The surfactant for droplet generation (10% Pluronic F68 solution) was purchased from Life Technologies (Carlsbad, CA). and stored at room temperature away from light.

Sequence design. The DNA sequence design was conducted at the domain level according to a series of design heuristics^{1,2}. All domain sequences used a three-letter code (A, C and T) to reduce secondary structures and unintended interactions. No more than four consecutive A's or T's in a row and no more than three consecutive C's were used in a row to reduce synthesis errors. All domain sequences with a 30% - 70% C content were

55 kept to ensure comparable melting temperatures. The maximum matching base number for any two domain
56 sequences in the pool cannot exceed 35% of the domain length, and more than 30% nucleotides are different.
57 Owing to the modularity of the proposed DNA switching gate architecture, all of the DNA single strands in
58 this work consist of a short toehold domain, long recognition domain and a specific stem-loop domain. A 11-
59 nucleotide sequence pool and 15-nucleotide sequence pool were generated and used for the recognition
60 domains, while a 27-nucleotide and 29-nucleotide sequence pool were generated and used for the stem-loop
61 domains. All sequences of DNA strands were implemented by directly concatenating these domains, and
62 verified using Nucleic Acids Package (NUPACK)³.

63 Six types of basic DNA components were used in this work (Fig. S1). In detail, the weight substrate
64 molecule $M_{w_i, l_{i,j}}$ consisting three single strands and the weight modulators S_{w_i} are designed for the weight
65 multiplication reaction, wherein all the A_{w_i, l_i} of $M_{w_i, l_{i,j}}$ shared a 6-nt or 7-nt universal toehold domain and all
66 the S_{w_i} shared a 7-nt universal toehold domain. The summation gates $S_{d_{j,k}}$ consisting two single strands are
67 designed for the summation reaction, all of which shared a 5-nt universal toehold domain. The subtraction
68 species Sub_{n, Y_i} consisting three single strands and $D_{d_{k, Y_i}}$ consisting two single strands are designed for the
69 subtraction reaction, in which all the Sub_{n, Y_i} shared a 7-nt toehold domain (a 5-nt universal toehold and 2-nt
70 extension that is complementary to the two nucleotides next to the toehold of the upstream complexes) and all
71 the $D_{d_{k, Y_i}}$ shared a 5-nt universal toehold domain. The reporters R_{Y_i} consisting two single strands modified
72 with fluorophore and quencher groups respectively, are designed for signal readout, wherein all the R_{Y_i} shared
73 a 5-nt universal toehold domain. Moreover, 2-nt clamps design were used in the bottom strands, which can
74 complement to the first two nucleotides in the tail of the DNA species to reduce the gate–gate leakage². We
75 used 2-nt spacer domains (‘TT’) to ensure an effective DSD reaction rate⁴, which was inserted between the
76 toehold domains and recognition domains of the input strands I_i and the single strand $D_{s_{k, Y_i}}$.

77 **Chip design and fabrication.** The chip hardware design featured a memory and a processor, is patterned with

an array of electrodes of varying size to accommodate varying droplet manipulation. The pattern of the electrode layer was designed using AutoCAD software. For the memory, an array of actuation electrodes connected to reservoir electrodes were designed for the introduction, storage and retrieval of molecular chiplets. For the processor, an array of driving electrodes for directional transportation of molecular chiplets and multiple reaction chamber electrodes for the combination of molecular chiplets, were designed for DNA computation. The electrodes were roughly rectangle, square or hexagon with a specified length on each side and inter electrode gaps. In detail, four different chips were designed and tested experimentally for varying on-chip computational functions: (1) The chip shown in Fig. S2 (80 mm by 60 mm by 5 mm) featured an array of 138 electrodes connected to 96 pin electrodes by 20 μm -wide wires, in which the memory contained an array of 49 actuation electrodes ($0.82\text{ mm} \times 0.82\text{ mm}$ and $1.30\text{ mm} \times 1.30\text{ mm}$) along with 40 reservoir electrodes ($1.60\text{ mm} \times 1.60\text{ mm}$ and $2.60\text{ mm} \times 2.60\text{ mm}$) and the processor contained an array of 41 actuation electrodes ($1.30\text{ mm} \times 1.30\text{ mm}$, $2.60\text{ mm} \times 2.60\text{ mm}$, and $3.80\text{ mm} \times 3.80\text{ mm}$) along with 8 reaction chamber electrodes ($2.60\text{ mm} \times 2.60\text{ mm}$). The reservoir droplet volumes on this chip were 2.5 μL and 1 μL , while the aliquoted droplet volumes were 0.25 μL and 0.1 μL . (2) The chip shown in Fig. S13 (86 mm by 86 mm by 5 mm) featured an array of 305 electrodes that connected to 192 pin electrodes by 40 μm -wide wires, in which the memory contained an array of 166 actuation electrodes (roughly hexagon with a side length of 0.78 mm and 1.75 mm) along with 112 reservoir electrodes ($1.19\text{ mm} \times 3.63\text{ mm}$, $0.57\text{ mm} \times 3.63\text{ mm}$, $2.66\text{ mm} \times 8.12\text{ mm}$, and $1.29\text{ mm} \times 8.12\text{ mm}$) and the processor contained an array of 23 actuation electrodes (roughly hexagon with a side length of 1.75 mm and 3.92 mm) along with 4 reaction chamber electrodes (roughly hexagon with a side length of 3.92 mm). The reservoir droplet volumes on this chip were 2.5 μL and 8.6 μL , while the aliquoted droplet volumes were 0.32 μL and 1.6 μL . (3) The chip shown in Fig. S22 (70 mm by 110 mm by 1.1 mm) featured an array of 921,600 electrodes (roughly square with a side length of 70 μm and 70 μm) controlled by independent a-Si thin-film transistor (TFT). Each electrode is individually addressable by

101 scanning the row and column control signals. (4) The chip shown in Fig. S31b (86 mm by 86 mm by 5 mm)
102 featured an array of 227 electrodes connected to 192 pin electrodes by 40 μm -wide wires, in which the memory
103 contained an array of 112 actuation electrodes (roughly hexagon with a side length of 0.78 mm and 1.75 mm)
104 along with 76 reservoir electrodes (1.19 mm $\times$ 3.63 mm, 0.57 mm $\times$ 3.63 mm, 2.66 mm $\times$ 8.12 mm, and 1.29
105 mm $\times$ 8.12 mm) and the processor contained an array of 30 actuation electrodes (roughly hexagon with a side
106 length of 1.75 mm and 3.92 mm) along with 9 reaction chamber electrodes (roughly hexagon with a side
107 length of 3.92 mm). The reservoir droplet volumes on this chip were 2.5 μL and 8.6 μL , while the aliquoted
108 droplet volumes were 0.32 μL and 1.6 μL .

109 All test chips were fabricated in the clean room facility by Dynamic Biosystems and Suzhou Institute of
110 Biomedical Engineering and Technology Chinese Academy of Sciences, each of which consists of a bottom
111 plate and a top plate. The bottom plates of DMF chip shown in Fig. S2 with an array of electrodes were
112 patterned by photolithography and wet etching. Specifically, 300-nm-thick Chromium and glass substrates
113 were coated with photoresist (AZ5214) and subjected to UV light. The exposed substrate was developed in
114 RZX-3338, etched in CR-4 and immersed in RBL-3368 to remove the remaining photoresist, forming the
115 electrodes. In addition, the bottom plates of DMF chips shown in Figs. S13 and S31b with an array of
116 electrodes were patterned by lift-off technique. Specifically, AZ52214 photoresist-coated glass substrates were
117 exposed to UV light through a photomask and developed in ZX238. A 100 nm layer of chromium and a 200
118 nm layer of gold were deposited onto the sample, after that the photoresist along with the metal was removed
119 in dimethyl sulfoxide, forming the electrodes. Then, the chips were coated with a ~ 1 μm height of SiN_x or a
120 ~ 0.8 μm height of SU-8 2000.5 photoresist as a dielectric layer. The top plate was indium-tin oxide (ITO)
121 glass with inlet and outlet holes, which was spin-coated with Teflon-AF [1% (w/w) in FC-40] as hydrophobic
122 layer. One top/bottom plate pair of the chip were joined with double-sided tape with a gap of 150 to 600 μm
123 and filled with silicone oil (PMX-200) for reducing the evaporation of droplets on the chip. The chip used in

Fig.S22 is designed with a sandwich-like structure, which features interlayers separated by a 50 μm double-sided adhesive gasket. The top plate with a reagent injection hole is a transparent glass, which coated with indium tin oxide (ITO) as a common electrode. The bottom plate incorporates an active-matrix (AM) (720 rows $\times$ 1280 columns) manufactured using standard a-Si:H thin-film transistor (TFT) fabrication processes⁵⁻⁹. The TFT backplane is fabricated using display-compatible amorphous-silicon thin-film transistor processes, with each pixel individually addressed through row/column scanning. The chip stack integrates the TFT electrode array, electrowetting dielectric and hydrophobic layers, a spacer-defined liquid chamber, and an ITO-coated counter electrode. The actuation system comprises the row/column addressing electronics, driving-voltage circuitry, control electronics, and software interface for translating programmed electrode patterns into droplet operations. The chip interfaces with the host computer via a flexible printed circuit (FPC). This chip enables the continuous DI-water droplet manipulation for more than 10 h, with a droplet-volume coefficient of variation of approximately 1%, demonstrating reliable long-term actuation and reproducible droplet handling⁵⁻⁹. All chips were spin-coated with Teflon-AF [1% (w/w) in FC-40] as a hydrophobic layer. Note that the volume of droplets was controlled by the electrode size and the gap of known height between the top and bottom plates.

The fabricated chips were integrated with external electronics to create a unified platform for electrically algorithmic control. This all-in-one platform included chip holder, computing chip, controller and customer-written control software. The predefined instruction set charts the activation plan of electrodes array, allowing for the mapping of user-defined set of algorithms into physical droplet operations. The chip was inserted into the holder, ensuring the pogo pin interface to make full contact with the pad on the chip. To electrically direct individual molecular chiplets on chip, a preprogrammed sequence of voltages (25 to 180 V) was applied between the top plate and electrodes in the bottom plate via a Pogo pin interface.

Molecular chiplets preparation. All molecular chiplets were first assembled by annealing the required DNA

147 strands to the corresponding final concentrations in $1\times$ TE buffer with 12.5 mM Mg^{2+} . All weight substrate
148 molecules were prepared at 50 μ M by mixing three DNA strands in a 1: 1: 1 ratio. All DNA duplexes were
149 annealed at 50 μ M with top and bottom strands in a 1: 1 ratio. All reporters were prepared by annealing at 50
150 μ M with top strands in 10% excess of bottom strands. For molecular chiplets required in Fig.4, all weight
151 substrate molecules were prepared at 96 μ M by mixing three DNA strands in a 1: 1: 1 ratio. All DNA duplexes
152 were annealed at 96 μ M with top and bottom strands in a 1: 1 ratio. All reporters were prepared by annealing
153 at 128 μ M with top strands in 10% excess of bottom strands. All complexes were thermally annealed in a
154 thermal cycler (Life Technologies) by heating up to 95 $^{\circ}$ C for 5 min and cooling from 95 $^{\circ}$ C to 25 $^{\circ}$ C at a rate
155 of 0.1 $^{\circ}$ C per 8 s, then stored at 4 $^{\circ}$ C for further use. Annealed DNA complexes were purified by 12%
156 polyacrylamide gel electrophoresis (PAGE). The varying DNA components and surfactant were mixed
157 together at constant concentrations in $1\times$ TE buffer with 12.5 mM Mg^{2+} , and then loaded into their designated
158 reservoirs by pipetting through the inlet and actuating the electrodes, generating spatially distinct molecular
159 chiplets immersed in the oil as droplet-based “molecular chiplets” on chip. Once the DMIC is assembled, the
160 electrodes can be sequentially activated to quantitatively retrieve, directional transport and combine molecular
161 chiplets for assembly of on-chip DNA circuit, which exhibit an excellent capacity for preprogrammed
162 operation.

163 **On-chip molecular chiplets retrieval characterization.** We evaluated the performance of the retrieval of
164 molecular chiplets by the uniformity of droplet volume aliquoted from a reservoir. By actuating the actuation
165 electrodes from the reservoir, a larger droplet was first electrostatically stretched. The middle electrode was
166 then switched off to split a larger droplet into a small droplet, achieving the precise retrieval of molecular
167 chiplets on chip. The aliquoted droplets were imaged, and then processed by ImageJ (<http://rsb.info.nih.gov/ij/>)
168 to measure cross-sectional area of each droplet and calculate the droplet volume. As the spacer height was
169 fixed, the volume of aliquoted droplet was directly related to its area. During one aliquoting operation, we

need average 5 s for electrostatically stretching and 1 s for splitting. Sixty aliquoted droplets were measured for each case.

Supervised learning details. The on-chip DNA-based ConvNet can be trained using the backpropagation rule¹⁰ to perform the perfect classification of 12-bit patterns into three classes. We constructed a training dataset, including three stylized letters ('L', 'T', 'J') and their 36 noisy counterparts, each generated by altering one pixel in the original images (Fig. S33). Each binary pixel value corresponds to one of the twelve network inputs, which is encoded using one chiplet $e\text{Drop}(I_i)$ (Fig. S31c). By this way, this training dataset were stored in the memory of DMIC, and electrically invoked and sequentially presented to the network. The flow chart of one epoch is shown in Fig.S34a, with details below.

For the training process, three identical DNA-based ConvNets were simultaneously configured in a single chip, and three input patterns $X_i(n)$ from the training dataset were applied to the network's input simultaneously. Note that a randomly weight values W_j were assigned for these networks at first epoch. At each epoch, these networks run independently in isolated reaction chambers to compute corresponding outputs ($y_i(n)$), where n is the pattern number, which are used to calculate the gradient of the error with respect to the weights W_j off-chip:

$$\delta_i(n) = y_i(n) - Y_i(n)$$

$$\frac{\partial E_n}{\partial W_j} = \delta_i(n) * X_i(n)$$

Here, $Y_i(n)$ is the target value of the i th output for the n th input pattern in our work. Note that these values were chosen to be 1.0 for the output corresponding to the correct patten, while 0.0 for the output corresponding to the wrong pattern.

Once all n patterns of the training dataset had been applied, we then calculated the delta-rule weight increments ΔW_j off-chip:

$$\frac{\partial E}{\partial W_j} = \sum_n \frac{\partial E_n}{\partial W_j}$$

$$\Delta W_j = -\frac{\eta}{N} \frac{\partial E}{\partial W_j}$$

where η is the learning rate, N is the total number of patterns.

After updating the weights, the process checks if the last epoch (all dataset were classified correctly) has been reached. If it is the last epoch, the training process is finished. If not, the training continues with the next epoch with the updated weights. Note that the weights in kernel were modified in parallel for each neural network on chip during operation, in which the weight value of each pixel can be individually reset by tuning the number of aliquoted weight modulator chiplets $e\text{Drop}(S_{wi})$ separately.

RNA extraction in serum samples. All human serum samples were collected from Zhongshan Hospital, Fudan University, (Shanghai, China) with informed consent. Total miRNA was extracted from serum samples (1 mL) using the Serum/Plasma miRNA Kit from HYCEZMBIO (Wuhan, China) following the manufacturer's instructions. The eluted total miRNAs were aliquoted and stored at -80 °C in RNase-free water until needed. All experiments and methods were complied with these approved guidelines.

Amplification and transformation. Synthetic or extracted miRNAs were first converted into cDNAs using a reverse transcription kit (Sangon Biotech) following the manufacturer's instructions. Specifically, the reverse transcription reactions were performed in a 10-μL system including reverse transcription enzyme and reaction buffer on the Roche Light Cycler 480 Real-Time PCR System (Life Technologies). The reverse transcription program consists of 37 °C for 60 min, and 85 °C for 5 min. Then, the acquired cDNAs were amplified by LATE-PCR without perturbing the original quantity information. Specifically, all templates including cDNAs, excess primer (1 μM), limiting primer (50 nM), and molecular beacon (1 μM) were mixed with 1× ThermoPol Reaction Buffer at 4 °C, and then Vent(exo-) DNA Polymerase (80 U/mL) was added and incubated in the Bio-Rad CFX96 Touch Thermocycler. The total volume of the amplification reactions was 20 μL. The

214 amplification program consists of 1 × at 95 °C for 5 min, 40 × cycles of 10 s at 95 °C, 10 s at 58 °C and 20 s
 215 at 72 °C, and 1 × at 72 °C for 10 min. We investigated the amplification efficiency by spiking 36 synthetic
 216 miRNAs (shown in Fig. S52) of 0.1 pM each to 36 separated 200-μL PCR tubes and added corresponding
 217 templates and enzymes. Lastly, the amplified cDNAs were transformed into the long ssDNA as input to the
 218 on-chip circuit. The cDNAs were mixed with 16 nM cT template strands, and the reaction buffer [20 mM tris-
 219 HCl (Aladdin), 10 mM (NH₄)₂SO₄ (Sinopharm Chemical Regent Co.), 40 mM KCl (Sinopharm Chemical
 220 Regent Co.), 10 mM MgSO₄ (Aladdin), 2 μM netropsin (Sangon Biotech), 200 μg/mL BSA (Aladdin), 0.1%
 221 Triton X-100 (Sangon Biotech), and 1 mM each dNTP (Sangon Biotech)] at 4 °C. After homogenization, 20
 222 U/mL Nt.BstNBI nicking enzymes and 180 U/mL Vent(exo-) DNA polymerase were added and incubated at
 223 50 °C for 120 min in the Bio-Rad CFX96 Touch Thermocycler. The transformed ssDNAs were characterized
 224 and purified using 8% PAGE. For bulk experiments, the fluorescence was recorded in real time.

225 **Network training and validation.** To train the ConvNet illustrated in Fig. 4, we constructed a dataset
 226 comprising 36 classes of handwritten symbols derived from the EMNIST (Extended MNIST) dataset,
 227 including 10 digits and 26 letters. We selected 1,000 images per class, yielding a total of 36,000 grayscale
 228 images (28 × 28 pixels) from the original dataset. The dataset was then divided into training (80%) and
 229 validation (20%) sets. All original handwritten symbols were resized to a size of 20 × 20 and then converted
 230 to binary patterns. The resulting 20 × 20 binary images were used as inputs for model training. For the
 231 classification of 36 handwritten symbols, 36 convolutional kernels of size 10 × 10 were first applied to 20 ×
 232 20 input pattern to compute the parallel convolutions (with a stride of 10), yielding 36 distinct feature maps
 233 with a size of 2 × 2. The ReLU activation function was then used to zero out any value less than zero in the
 234 feature maps. The following step was the operation of a global sum pooling operation, which transforms the
 235 output matrix into a vector by summing up all pixels in each feature map. We trained the model on TensorFlow
 236 platform (<https://www.tensorflow.org>). The Adam¹¹ optimizer was chosen to update the model's parameters.

237 During training, the learning rate, the batch size and the epoch numbers were set as 0.001, 64 and 1000,
238 respectively. Sparse categorical cross entropy was used as the loss function. Finally, the accuracy of this model
239 on the training set reached 84.28%, while the accuracy on the test set reached 80.64%. All hyperparameters
240 were selected empirically through comparative experiments and were chosen to balance model complexity
241 with predictive performance.

242 The two-step decision system illustrated in Fig. 5, including support vector machine (SVM) and fully
243 connected neural network (FCNN), were trained using Python library, including svm. LinearSVC function
244 and Keras (<https://keras.io>). To train these two classifier models, we obtained microarray data (GSE113740,
245 GSE137140, and GSE106817) for cancer patients with lung cancer (LC), ovarian cancer (OC), and
246 Hepatocellular carcinoma (HCC) and healthy individuals from National Center for Biotechnology Information
247 (NCBI). Among them, we selected female patients labelled only as LC, OC, HCC and healthy (808, 360, 158
248 and 1009 samples, respectively) as dataset and mathematically transforming the expression ratios by logarithm
249 (base 2) to estimate expression levels. miRNAs, with expression level lower than 2, in more than 50% of the
250 cancer and non-cancer samples in the selected dataset were excluded. The optimal combination of the
251 identified miRNAs was explored using differential expression analysis following by principal component
252 analysis (PCA). Briefly, we first performed differential expression analysis to identify the primary up and
253 down-regulated miRNAs, and selected miRNAs with a significance level (p-value) less than 0.01. The
254 principal component analysis (PCA) was then used to assess importance of each miRNA feature in the
255 diagnostic determination, and 36 miRNAs were finally selected as input features for classifier models. Using
256 this dataset, we trained a SVM model to differentiate between cancer and non-cancer samples, and a FCNN
257 model to identify the three cancer types (LC, OC and HCC). For each training model, 80% of the selected
258 datasets were put in the training dataset while the rest were put in the test dataset. We validated the discriminant
259 models using the test dataset. During training, the learning rate, the batch size and the epoch numbers were

260 set as 0.003, 36 and 5000, respectively. The diagnostic sensitivity, specificity, and accuracy were calculated
261 for each resulting model in the training dataset and test dataset.

262 **Imaging and analysis.** The incubated droplets within the reaction chamber were collected from the chip's
263 outlet and imaged by the fluorescence microscopy (Nikon ECLIPSE Ti, Tokyo, Japan) under $\times 4$ objective.
264 For the characterization of 26-DNA neuron network, the incubated droplets were similarly collected and
265 imaged using the laser confocal scanning microscope (Leica STELLARIS 8 STED, Germany) under $\times 10$
266 objective and $\times 0.75$ zoom. All images were acquired using identical settings on microscopy, and
267 corresponding fluorescence intensity was quantitatively measured using ImageJ. For the characterization of
268 on-chip DNA-based ConvNet shown in Fig. 4, the incubated droplets were collected and imaged using the
269 fluorescence microscopy (Nikon ECLIPSE Ni-U) under $\times 4$ objective. All raw data of fluorescence signals
270 were normalized to the standard concentration of the output signals. A set of parallel experiments for the same
271 on-chip DNA circuit with different input combinations were performed and normalized together for data
272 analysis. The maximum fluorescence level (output = 1) was determined by the highest signal produced from
273 the parallel experiments at the same time, while the minimum fluorescence level (output = 0) generated by the
274 on-chip DNA circuit without inputs, was measured as the baseline.

275 **Fluorescence kinetic experiments.** Fluorescence kinetics experiments were conducted using a
276 spectrofluorometer (Fluorolog-max, Horiba) at 25 °C in a 1 mL cuvette. This spectrometer allows four kinetics
277 experiments to be run simultaneously. Fluorescence kinetic data were measured every 5 seconds, 2 minutes,
278 or 5 minutes. The excitation/emission wavelength were 492 nm/518 nm for FAM. All DNA components,
279 except the inputs and weight modulators, were mixed first in a specific concentration ratio and then added to
280 the cuvette to measure the initial fluorescence intensity as baseline (output = 0). The inputs and weight
281 modulators were then added to trigger the DNA circuit. All raw data of fluorescence signals were normalized
282 to the standard concentration of the output signals. A set of parallel experiments for the same circuit with

different input combinations were performed and normalized together for data analysis. The maximum fluorescence level (output = 1) was determined by the highest signal produced from the parallel experiments at the same time.

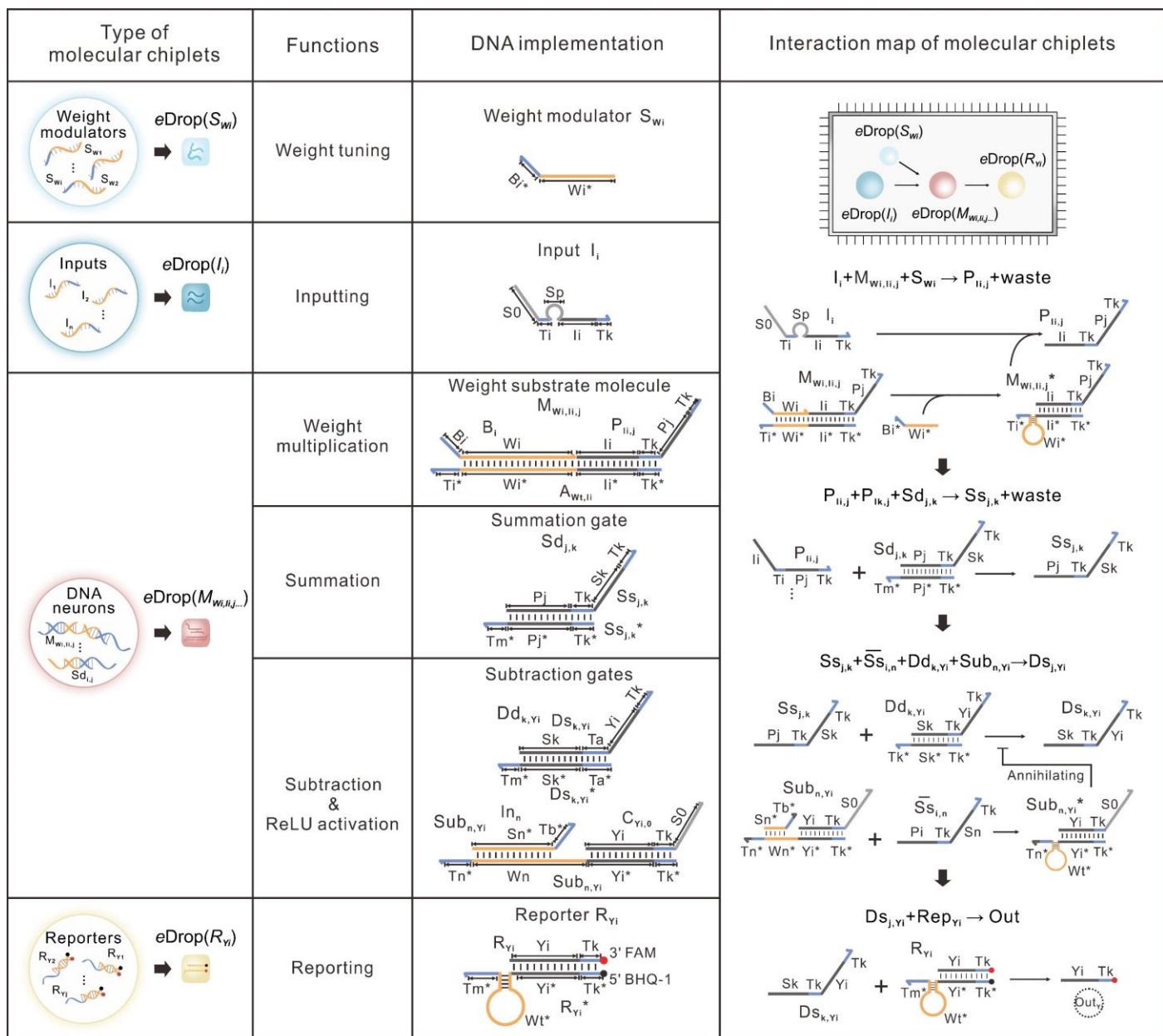


Fig. S1| Molecular chiplet library design. The left column lists four core molecular chiplet types and their functions: Weight modulators ($eDrop(S_{wi})$ for weight tuning), Inputs ($eDrop(I_i)$ for signal input), DNA neurons ($eDrop(M_{wi,i,j} \dots)$ performing weight multiplication, summation, subtraction and activation functions), and Reporters ($eDrop(R_{yi})$ for signal readout). The middle column illustrates the DNA implementation of molecular chiplets. Symbols with subscripts represent DNA species, while those without subscripts denote the elementary functional domains of DNA species. Each functional domain is represented by a coloured line. And a complementary sequence is indicated by an asterisk. The interaction map in the right column depicts the step-by-step reactions among different molecular chiplets during computation. The weight substrate molecule $M_{wi,i,j}$ and weight modulator S_{wi} are designed for the weight multiplication function of $x_i \times w_i$ (x_i is a binary input, and w_i is an analogue weight), in which weight substrate molecule can be stoichiometrically activated upon hybridization with weight modulator, producing the intermediate species $M_{wi,i,j}^*$, and allowing input species to react with $M_{wi,i,j}^*$ to generate an intermediate product $P_{i,j}$. The binary value ('1' or '0') of input corresponds to the absence or presence of input species. If input is absent, no $P_{i,j}$ will be produced; If input is present, the final concentration of $P_{i,j}$ will be determined by the concentration of weight modulator S_{wi} , thus setting the value of the weight multiplication. The summation gate $S_{d,j,k}$ is designed for the summation function of $\sum x_i$, wherein all the intermediate species $P_{i,j}$ with the same right recognition domain

convert the summation gate to a common weighted-sum species $S_{j,k}$. Subtraction gates, consisting of the double-stranded complex D_{k,Y_i} and annihilation species Sub_{n,Y_i} , are designed for the subtraction function of $x_i - x_j$. Specifically, the upstream positive weighted-sums $S_{j,k}$ reacts with D_{k,Y_i} to generate an intermediate species $D_{S_{j,k},Y_i}$. In the same time, the upstream negative weighted-sums $\bar{S}_{i,n}$ can activate Sub_{n,Y_i}^* by binding to the inhibitory strand In_n , allowing Sub_{n,Y_i}^* to annihilate $D_{S_{j,k},Y_i}$ each other. Note that only the remaining $D_{S_{j,k},Y_i}$ can react with the downstream gates, implementing the nonlinear activation function (ReLU). A reporter R_{Y_i} with a fluorophore/quencher pair at the end of the duplex, is designed for output readout. This tabular layout clearly articulates the design logic and inter-unit interactions within the molecular chiplet library, constituting a foundational “molecular instruction set” for on-chip programmable computation.

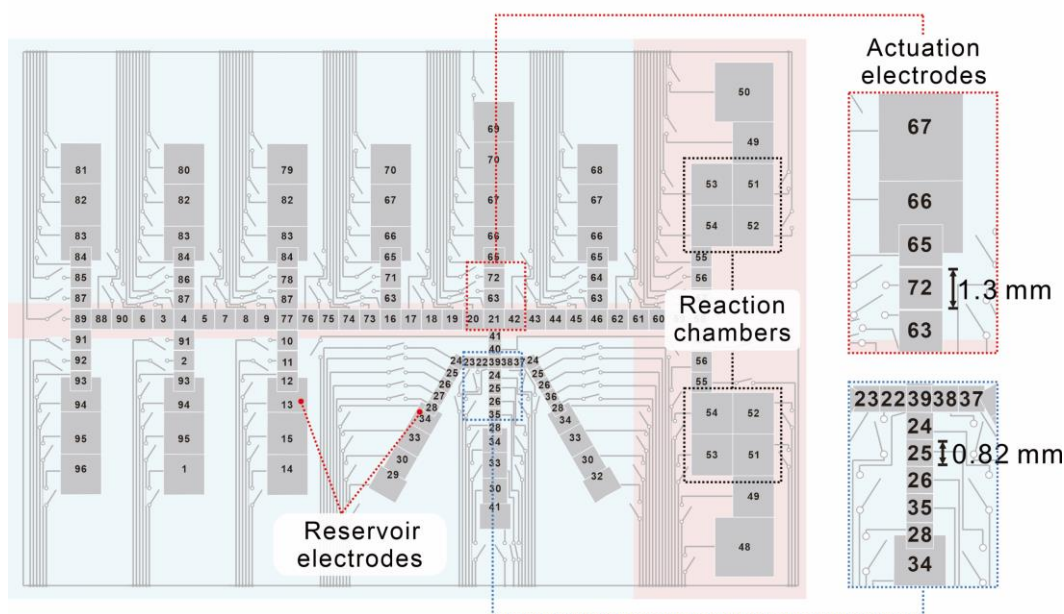
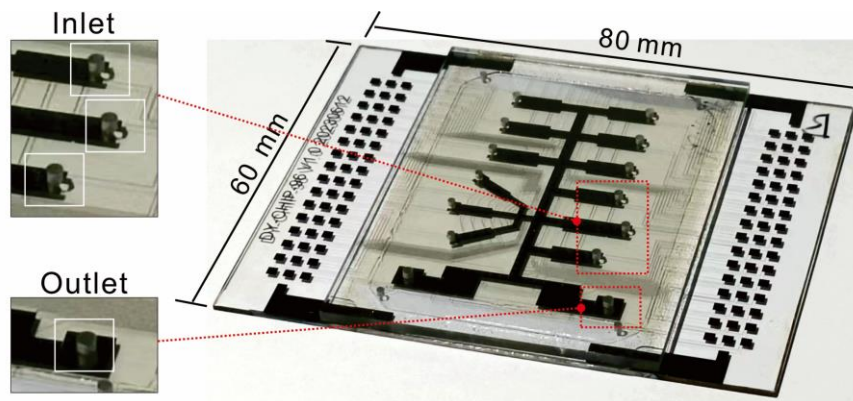


Fig. S2| Design details of chip hardware for on-chip DNA neuron computation. This chip comprises 12 available reservoirs in memory, 138 actuation electrodes, and two reaction chambers in processor, allowing 12 molecular chiplets to be stored simultaneously, and enabling two on-chip DNA circuits to run in parallel and asynchronously as they arrive. Each reservoir corresponds to a liquid inlet. Electrodes of different sizes are designed to accommodate different droplet volumes. The numbers identify the position of electrode on the chip. A customer-written control software 192MicroFluidics_v1.1 was used for the automatic execution of chip, in which the instruction set is first translated into the script file, and then uploaded to the software for automatic and sequential activation of electrodes.

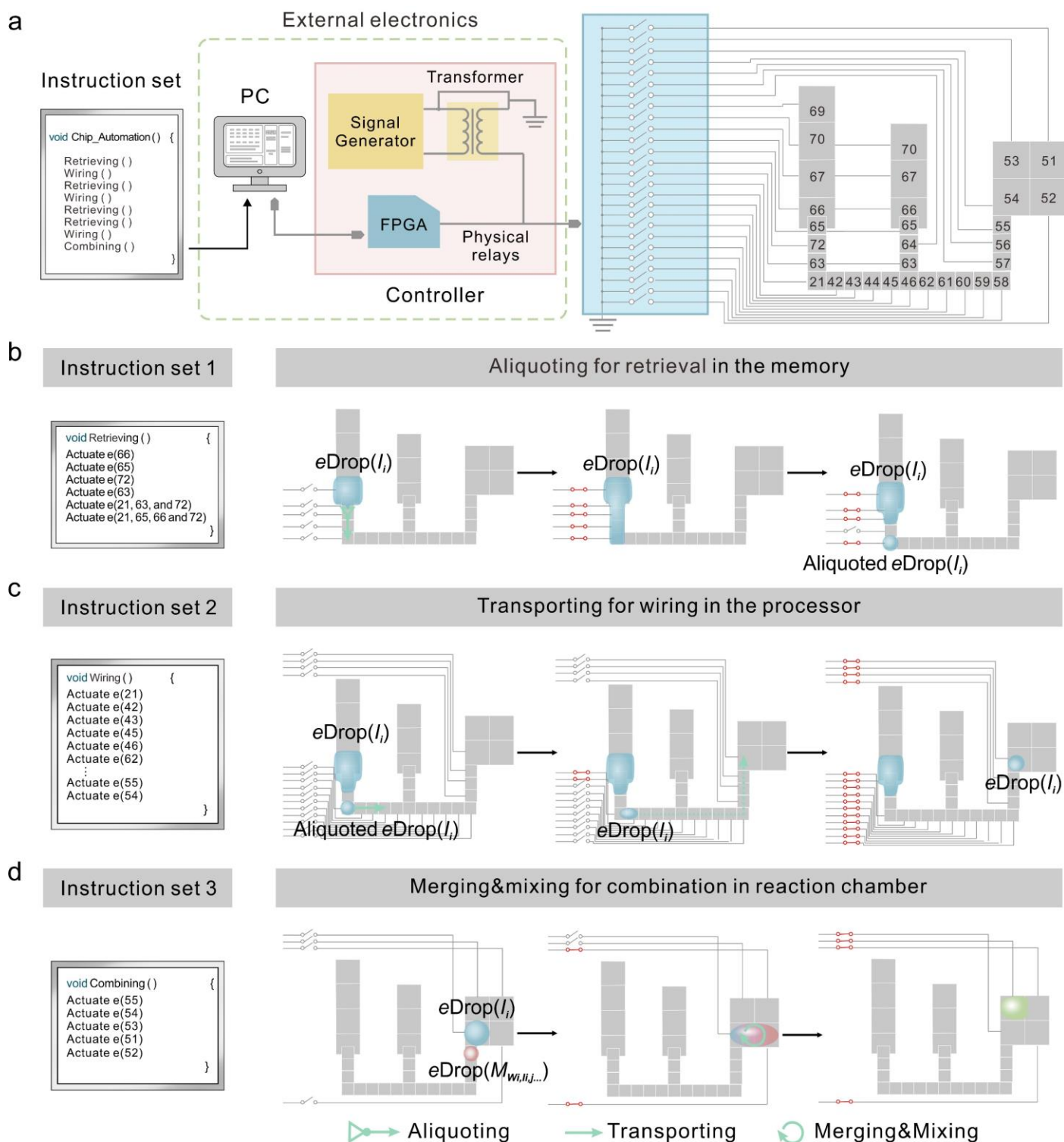


Fig. S3| The controller interfaced with instruction sets for automated molecular information processing. **a**, A software-programmable microcontroller, which follows a predefined instruction set, is interfaced with a bank of switching ICs to selectively activate the addressable electrodes, and subsequently, direct the molecular chiplets to perform the required operations. **b,c,d**, The manipulation of molecular chiplets, and its corresponding instruction set. The instruction set outlines the electrode activation scheme used to manipulate droplets.

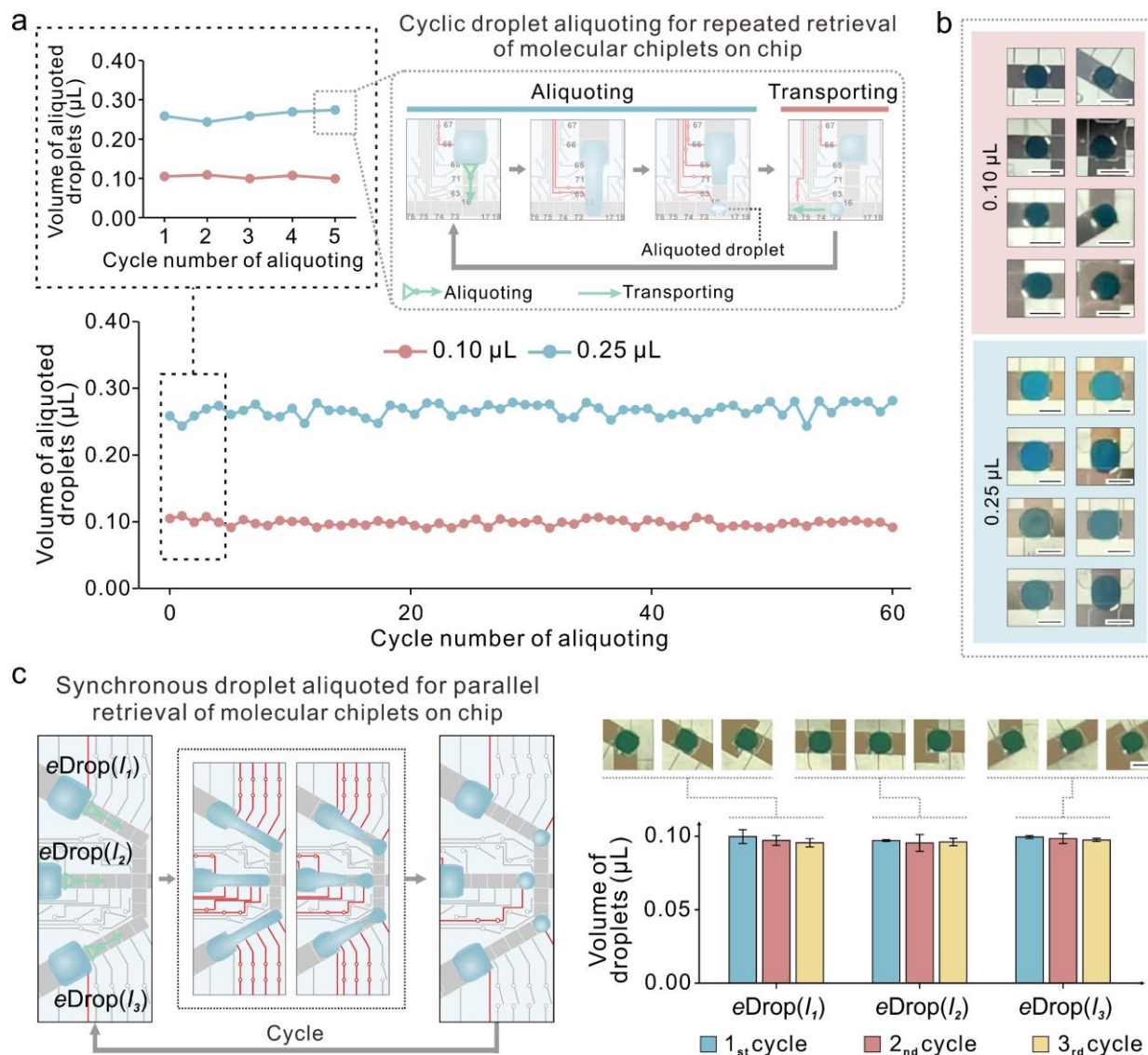


Fig. S4| The retrieval of molecular chiplets on DMIC via droplet operations. **a**, Characterization of the repeated retrieval of the molecular chiplets on DMIC, involving aliquoting and transporting operations, to evaluate the robustness of the retrieval operations. By programming the underlying actuation electrodes connected to the respective reservoir, the quantitative molecular data retrieval can be implemented based on the precise and tuneable sample aliquoting capability. Specifically, each molecular chiplet (for example, $e\text{Drop}(I_i)$) is initially stretched by actuating four electrodes connected to the respective reservoir. After that, the third electrode was switched off to force a small droplet to separate from the larger droplet, completing the quantitative retrieval of molecular chiplets. The aliquoted droplet is then transported to the surrounding areas, allowing cyclic aliquoting operations for repeated retrieval of the molecular chiplets on chip. The aliquoted droplets, ranging in volume from 0.10 μL to 0.25 μL , with an average coefficient of variation (CV) of 3.70% and 4.94% (characterized optically), respectively, which exhibits a narrow size distribution and long stability. **b**, Example images of the aliquoted droplets (scale bar, 1 mm). **c**, Characterization of the parallel retrieval of the molecular chiplets on chip. Three different molecular chiplets can be parallelly retrieved through synchronous aliquoting operations, with a CV of 3.11%. The inset shows corresponding images of three aliquoted droplets under different cycles (scale bar, 1 mm). Error bars indicate mean $\pm$ s.e. ($n = 3$ independent experiments). Image of the test chip used for the experiments is shown in Fig. S2.

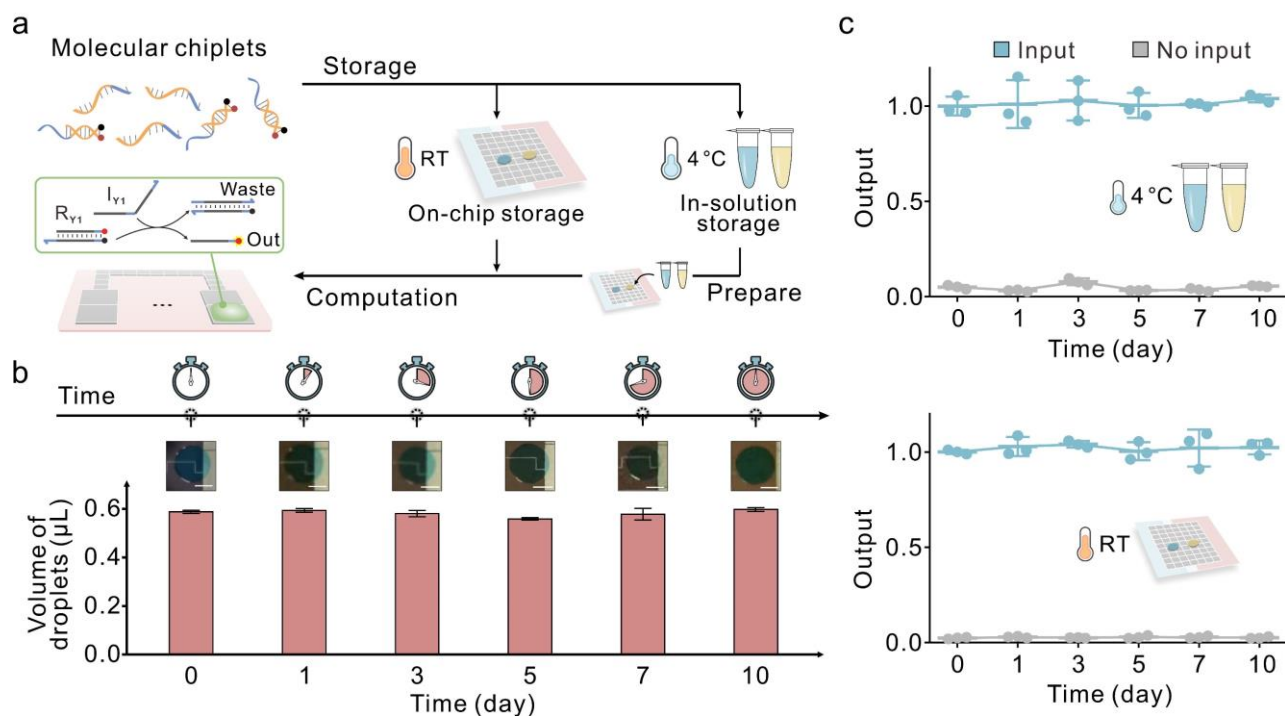
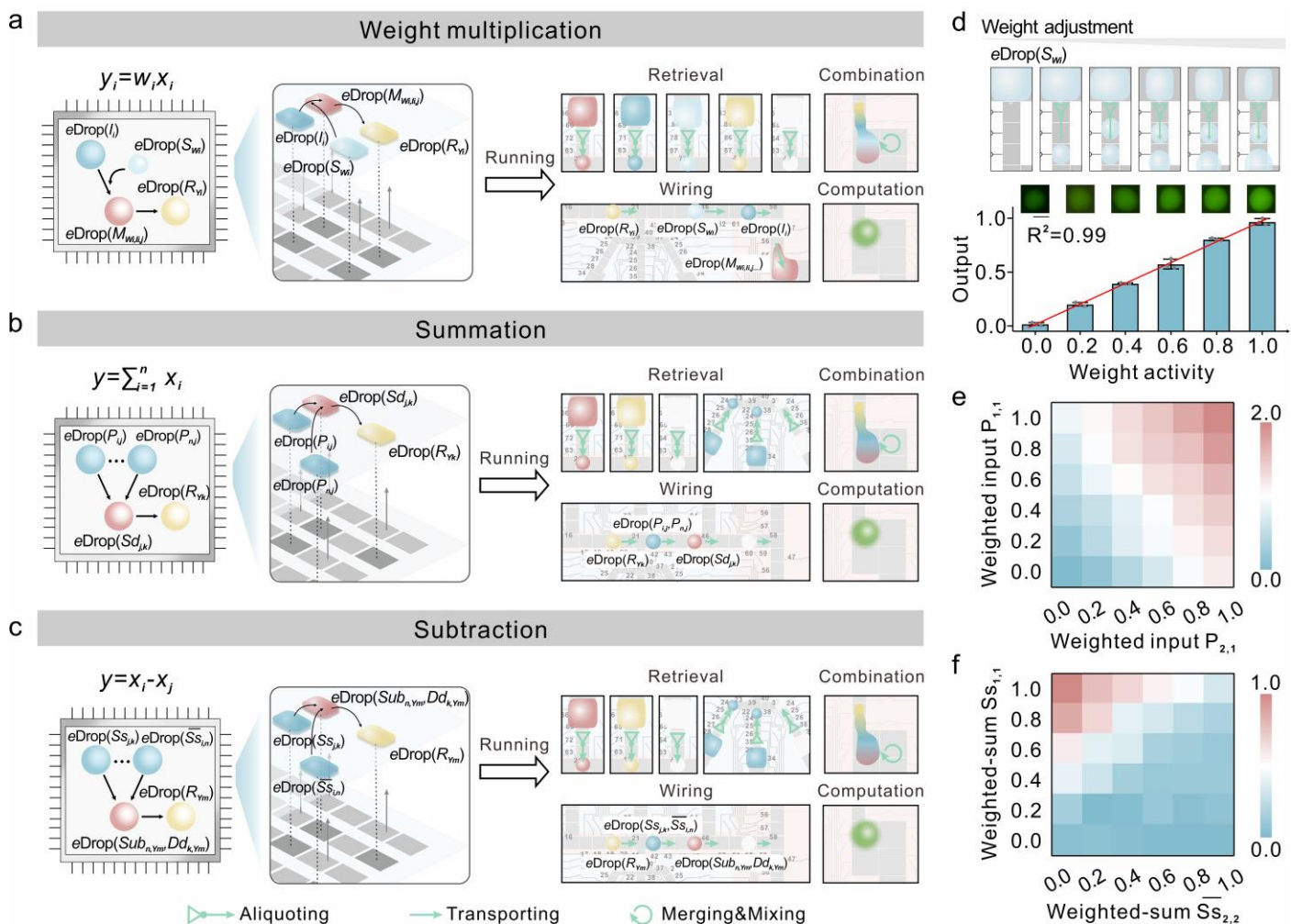


Fig. S5| The investigation of on-chip storage of molecular chiplets. **a**, Two types of storage modes studied here. (1) The assembled molecular chiplets were individually encapsulated and directly stored on chip at room temperature, which can be directly used for computation without additional manual operations. (2) The assembled molecular chiplets were first stored at 4 °C, and then loaded onto chip for computation by the pipette. The green box shows the details of molecular chiplets (reporters R_{Y1}) used for testing these two types of storage modes. Coloured lines indicate DNA strands at the domain level, whereas colors indicate distinct DNA sequences. The arrowheads mark their 3' ends. **b**, The characterization of the volume of droplet from day 0 to 10. Error bars indicate mean $\pm$ s.e. ($n=3$ independent experiments). Corresponding images of the droplet stored directly on chip for different periods of time are shown on the top (scale bar, 1 mm). The data indicates that the droplets hold great potential to act as a stable microcontainer for storing DNA molecules. **c**, The characterization of the behaviour of the on-chip DNA circuit after 0.5 h, from day 0 to 10, to evaluate the stability of molecular chiplets stored directly on chip. Error bars indicate mean $\pm$ s.e. ($n=3$ independent experiments). The fluorescence intensity is basically unchanged in 10 days, which means that molecular chiplets can be stably stored directly on chip at room temperature over 10 days. The volume of ($e\text{Drop}(R_{Y1})$ and $e\text{Drop}(I_{Y1})$) located in the memory is 2.50 μL (DNA species R_{Y1} and I_{Y1} concentration: 20 $\times$). The volume of droplet aliquoted from the memory is 0.25 μL . The standard concentration is 50 nM (1 $\times$). Image of the test chip used for the experiments is shown in Fig. S2.

S3: Design Details of the On-chip DNA Neuron

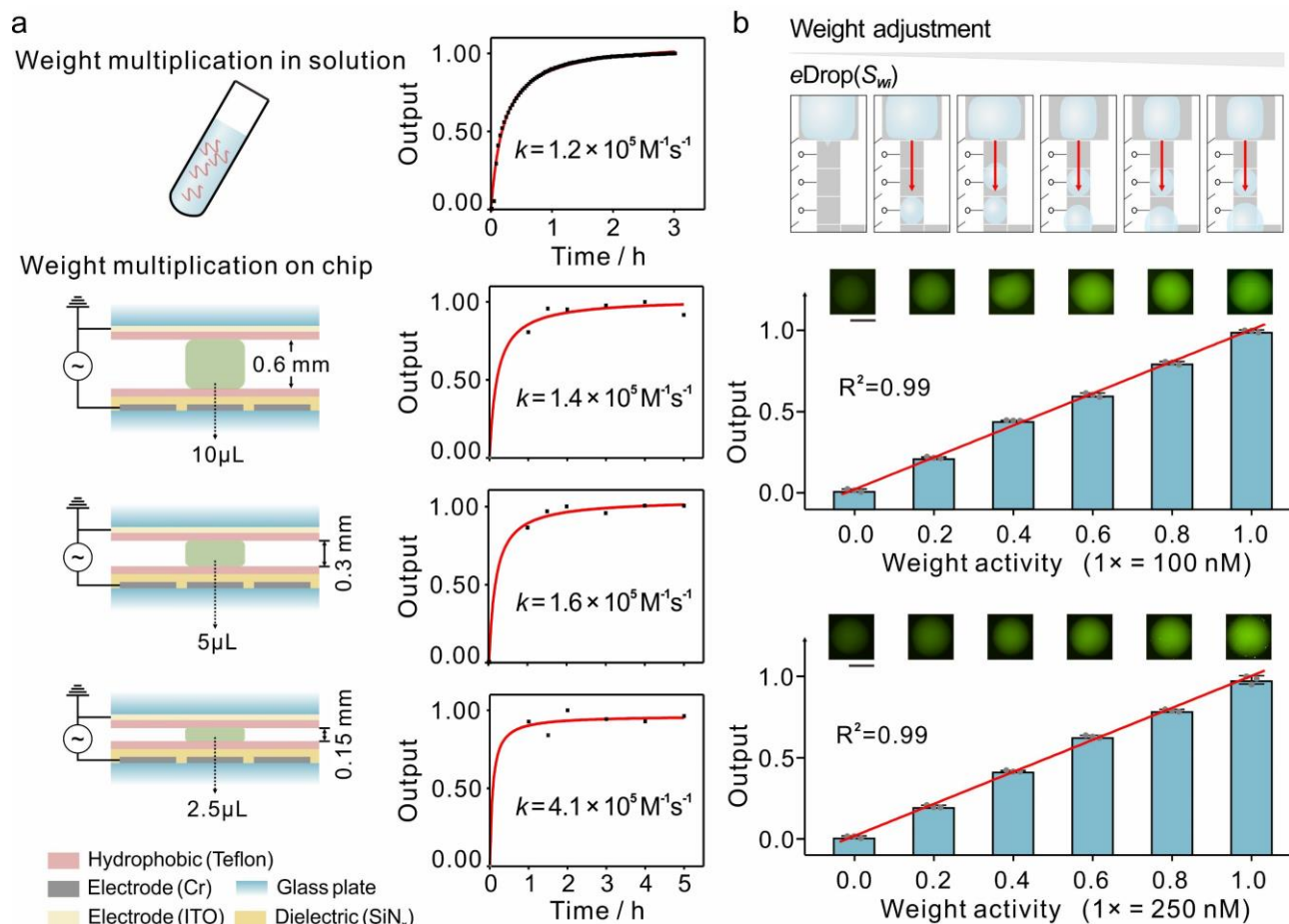


393

Fig. S6| On-chip implementation and experimental characterization of the subfunctions required for neuron computation. **a,b,c**, Schematic of subfunctions on chip. Each required DNA component is individually encapsulated into the droplets, forming the digitally controllable molecular chiplets $eDrop(v)$ that can perform the subfunctions such as weight multiplication (**a**), summation (**b**), subtraction (**c**) and reporting. Different colored droplets indicate different molecular chiplets. These chiplets are arranged onto the memory (blue area on the chip) to form a library of molecular chiplets on chip, each of which can be selective retrieved to perform the mathematical function in an automated manner. The gray rectangle represents the electrode array. The inputs for the weight multiplication ($x_i \times w_i$) remains binary, with each value of 1 or 0 indicating whether a droplet $eDrop(I_i)$ stored in the memory is retrieved or not. Each analogue weight value is implemented with the number of times the weight modulator chiplet $eDrop(S_{w_i})$, is aliquoted. A test chip shown in Fig. S2 was used for the experimental testing of the subfunctions. The sequential expanded views show the automated droplet manipulation in the process. The aliquoted droplets are shown as colored solid circles on the chip. **d**, The characterization of weight multiplication on chip ($x_i=1$, measured after 12 h). The volume of molecular chiplets located in the memory is 2.50 μL (DNA species I_i and M_{w_i, I_i} concentration: 20 $\times$, DNA species S_{w_i} concentration: 2 $\times$, and DNA species R_{y_i} concentration: 10 $\times$). The volume of molecular chiplets aliquoted from the memory is 0.25 μL . **e**, The characterization of summation on chip (measured after 1 h). The volume of molecular chiplets located in the memory is 2.50 μL (DNA species $S_{d_{j,k}}$ and R_{y_i} concentration: 20 $\times$), while the volume of $eDrop(P_{i,j})$ and $eDrop(P_{n,j})$ is 1.00 μL (DNA species $P_{i,j}$ and $P_{n,j}$ concentration: 5 $\times$). The volume of molecular chiplets aliquoted from the memory is 0.25 μL , while the volume of aliquoted $eDrop(P_{i,j})$ and $eDrop(P_{n,j})$ is 0.10 μL . **f**, The characterization of subtraction on chip (measured after 12 h). The volume of molecular chiplets located in the memory is 2.50 μL (DNA species Dd_{k, y_i} and R_{y_i} concentration: 10 $\times$, DNA species Sub_{n, y_i} concentration: 20 $\times$), while the volume of $eDrop(S_{s_{j,k}})$

413
414
415
416
417

and $e\text{Drop}(\bar{S}_{i,n})$ is 1.00 μL (DNA species $S_{j,k}$ and $\bar{S}_{i,n}$ concentration: 5 $\times$). The volume of molecular chiplets aliquoted from the memory is 0.25 μL , while the volume of aliquoted $e\text{Drop}(S_{j,k})$ and $e\text{Drop}(\bar{S}_{i,n})$ is 0.10 μL . Error bars indicate standard deviations (mean $\pm$ s.d., $n=3$), and fluorescence microscopy images are shown along the top (d), with scale bars of 1 mm. The standard concentration is 50 nM (1 $\times$).



418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437

Fig. S7| Experimental characterization of weight multiplication. **a**, Kinetics experiments of reaction volume (from 2.50 μL to 10.00 μL) on the operation of the weight multiplication shown in Fig. S6a. **b**, The experimental characterization of weight multiplication on chip under different concentration ($x_i=1$, measured after 12 h). Error bars indicate standard deviations (mean $\pm$ s.d., $n=3$). The fluorescence microscopy images are shown along the top, with scale bars of 1 mm.

This automatic on-chip DNA neuron computation proceeds in the following order: the required molecular
 chiplets were first prepared and located on the memory (Fig. S8). The actuation electrodes connected to
 reservoirs, were then turned on and off sequentially as planned to electrostatically stretch the chiplets (except
 for $e\text{Drop}(R_{YI})$), implementing the quantitative retrieval of molecular chiplets (from (1) to (14)); Subsequently,
 we electrically directed the chiplets to compute within the processor. These aliquoted chiplets were transported
 towards a designated reaction chamber through the continuous actuation of electrodes in the processor, while
 aliquoted $e\text{Drop}(S_{Wi})$ were in a standby state (from (15) to (16)). When corresponding aliquoted droplet
 reached specified electrode, $e\text{Drop}(R_{YI})$ was aliquoted (17). Then, all chiplets (except for aliquoted $e\text{Drop}(S_{Wi})$)
 continued to travel alongside the electrodes towards the reaction chamber (from (18) to (19)). The aliquoted
 $e\text{Drop}(S_{Wi})$ initiated their movement on chip as corresponding aliquoted droplet reached specified electrodes
 (20). After that, all chiplets continued to move (from (21) to (22)). These chiplets entered one reaction chamber
 sequentially, which allows the autonomous integration of multiple elements in a microcompartment. By
 merging-mixing the first successive aliquoted droplets, a chiplet $e\text{Drop}(M_{Wi, Ii, j...})$, that computes the function
 of an artificial neuron, was first configured (from (23) to (24)). After that, $e\text{Drop}(I_i)$ and $e\text{Drop}(S_{Wi})$ were
 coupled into $e\text{Drop}(M_{Wi, Ii, j...})$ to provide the input of the on-chip neuron and activation of weights, while
 $e\text{Drop}(R_{YI})$ was coupled together for signal readout (from (25) to (27)). This aggregated droplet (shown in
 (27)) serves as microcontainer for the compartmentalized DNA strand displacement cascades implementation
 (shown in Fig. S9) to compute the single neuron computation. Finally, the aggregate droplet was moved to the
 detection area for readout (28). The order of retrieval and wiring of molecular chiplets on chip was carefully
 arranged to minimize the delay of interconnects among the network. To automate the process, the instruction
 set was initially implemented and then converted into a script file (Fig. S10), which can be uploaded to the
 computer to automatically activate the electrodes and perform on-chip DNA computation as scheduled.

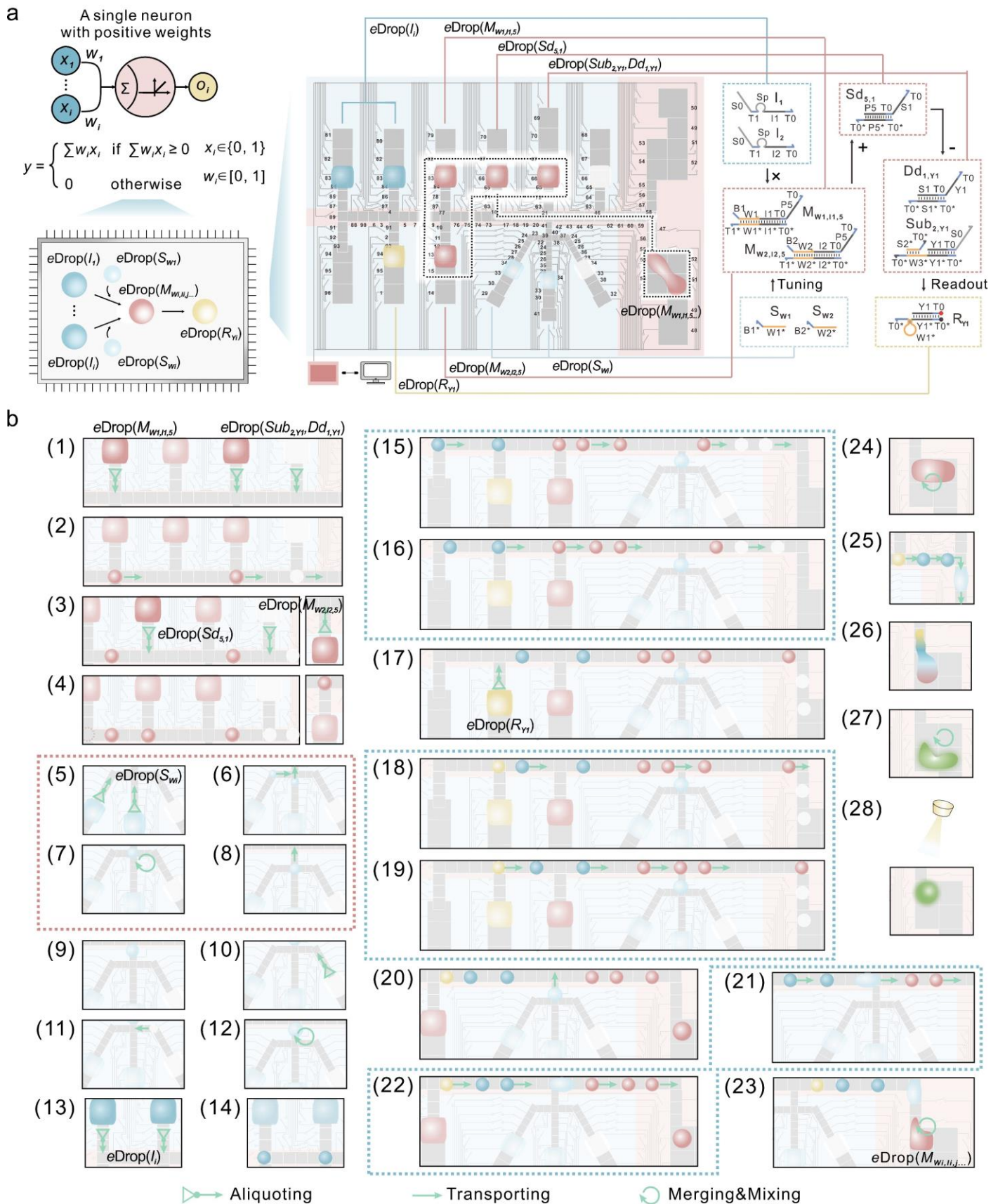


Fig. S8| An on-chip DNA neuron with positive weights. **a**, The on-chip implementation of a single DNA neuron. The encapsulated species are shown in the colored dashed boxes, with arrows indicating the flows of the strand displacement reactions within the large droplet (2.75 μ L) formed in the reaction chamber. The black dotted box shows that four molecular chiplets are merged into one droplet that perform neuron computation. In detail, $eDrop(M_{w1,i,j,...})$ and $eDrop(M_{w2,i,j,...})$ are designed to implement two parallel weight multiplications, followed by $eDrop(Sd_{5,i})$ and $eDrop(Sub_{2,y1}, Dd_{1,y1})$ to complete

the summation. The values of variables x_i and w_i correspond to the concentrations of DNA species X_i and S_{Wi} within the large droplet formed within the reaction chamber, respectively. The volume of droplets located in the reservoir region is 2.5 μL (DNA species I_i and $M_{Wi, Ii, j}$ concentration: 22 $\times$, DNA species $S_{d, j, k}$, $D_{d, k, Yn}$ and R_{Yn} concentration: 33 $\times$, DNA species $Sub_{j, Yn}$ concentration: 66 $\times$), except for the volume of $e\text{Drop}(S_{Wi})$ is 1 μL (DNA species S_{Wi} concentration: 5.5 $\times$). The numbers identify the position of electrode on the chip. Image of the fabricated chip is shown in Fig. S2. **b**, Schematic illustrating the detail workflow of on-chip DNA neuron computation ($w_1 = 0.4$, $w_2 = 0.4$). The detail automated droplet operations process spans steps (1) to (28). The pink dashed box shows the example droplet operations needed to retrieve the required weight modulator chiplets $e\text{Drop}(S_{Wi})$. Note that these droplet operations (from (5) to (8)) needed to repeat many times for varying weight settings (for example, two times repeat operations for $w_1 = 0.4$, $w_2 = 0.4$). The blue dashed boxes show the example droplet operations needed to transport all molecular chiplets. All molecular chiplets traveled alongside the electrodes by the sequential actuation of electrodes towards the reaction chamber. The volume of droplets aliquoted from the reservoir is 0.25 μL , while the volume of aliquoted $e\text{Drop}(S_{Wi})$ is 0.1 μL .

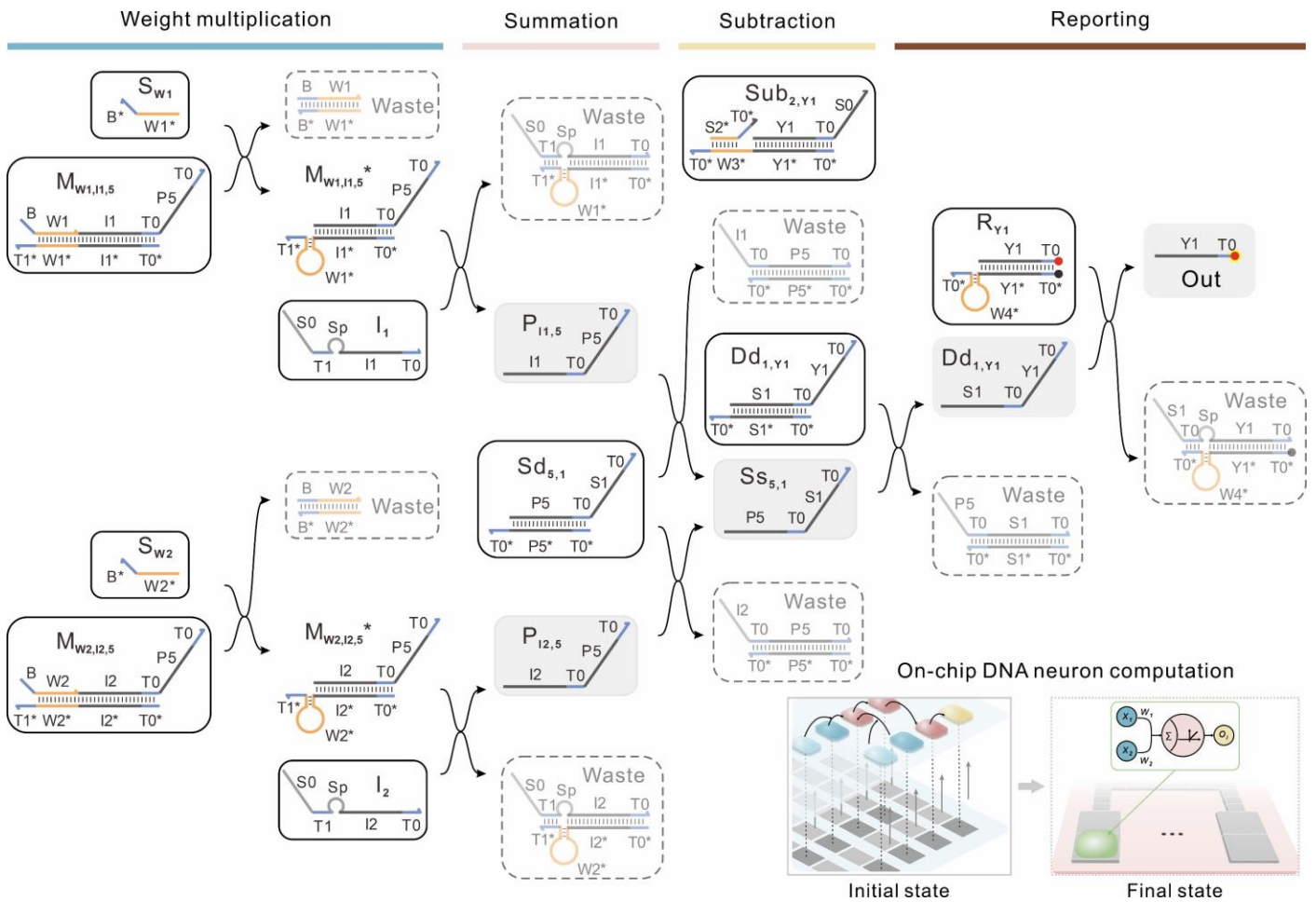


Fig. S9| DNA implementation of a single DNA neuron with two inputs and its reaction pathway within reaction chamber. The outlined boxes highlight the initial DNA species; the shaded boxes highlight the immediate species; the dotted boxes highlight the generated inert species. The arrows indicate the flows of the forward reactions. All initial DNA species were individually encapsulated into droplets, as controllable molecular chiplets on chip. An aggregated droplet was ultimately obtained after coupling of multiple droplets, which serves as microcontainer to allow the compartmentalized DNA strand displacement cascades implementation to compute the DNA neuron operation. In detail, input species I_i initially reacted with activated weight substrate molecule $M_{w_i,i,j}^*$ to set the value of the weighted input, in which a certain concentration of weight modulators S_{w_i} activated the weight substrate molecules $M_{w_i,i,j}$ to set the value of weights. After weighting, all intermediate products $P_{i,j}$ were combined to a single DNA strand with a summation and a subtraction gates, summing up the weighted inputs, and leading to the triggering of reporter.

```

void Chip_Automation ( )
{
    Retrieving (eDrop( $M_{w1,i1,s}$ ), eDrop( $Sub_{2,y1}$ ,  $Dd_{1,y1}$ ), reaction buffer 1)
    Actuate e(83, 66)
    Actuate e(84, 65)
    Actuate e(78, 72, 64)
    .....
    Wiring (eDrop( $M_{w1,i1,s}$ ), eDrop( $Sub_{2,y1}$ ,  $Dd_{1,y1}$ ), reaction buffer 1)
    Actuate e(76, 42, 62)
    Actuate e(75, 43, 61)
    Retrieving (eDrop( $M_{w2,i2,s}$ ), eDrop( $Ss_{s,1}$ ), reaction buffer 1)
    Actuate e(13, 66)
    Actuate e(12, 65)
    Actuate e(11, 71, 64)
    .....
    Retrieving (eDrop( $S_{w1}$ ), eDrop( $S_{w2}$ ))
    Actuate e(34, 83, 66, 94, 13)
    Actuate e(28)
    Actuate e(27, 35)
    .....
    Wiring (eDrop( $S_{w1}$ ), eDrop( $S_{w2}$ ))
    Actuate e(23)
    Actuate e(22)
    Actuate e(39)
    .....
    Retrieving (eDrop( $S_{w1}$ ), eDrop( $S_{w2}$ ))
    Actuate e(34)
    Actuate e(28)
    Actuate e(27, 35)
    .....
    Wiring (eDrop( $S_{w1}$ ), eDrop( $S_{w2}$ ))
    Actuate e(23)
    Actuate e(22)
    Actuate e(39)
    .....
    Retrieving (reaction buffer 2)
    Actuate e(34, 41)
    Actuate e(28, 36)
    Actuate e(36, 26)
    .....
    Wiring (reaction buffer 2)
    Actuate e(37)
    Actuate e(38)
    Actuate e(39)
    .....
    Retrieving (eDrop( $I_1$ ), eDrop( $I_2$ ))
    Actuate e(83, 66, 94, 13)
    Actuate e(84)
    Actuate e(85, 86)
    .....
    Wiring (eDrop( $I_1$ ), eDrop( $M_{w1,i1,s}$ ), eDrop( $Sub_{2,y1}$ ,  $Dd_{1,y1}$ ),
    eDrop( $I_2$ ), eDrop( $M_{w2,i2,s}$ ), eDrop( $Ss_{s,1}$ ), reaction buffer 1)
    Actuate e(88, 74, 44, 60)
    Actuate e(5, 76, 17, 62)
    Actuate e(90, 73, 45, 59)
    .....
    Retrieving (eDrop( $R_{y1}$ ))
    Actuate e(94)
    Actuate e(93)
    Actuate e(2)
    .....
    Wiring (eDrop( $I_1$ ), eDrop( $M_{w1,i1,s}$ ), eDrop( $Sub_{2,y1}$ ,  $Dd_{1,y1}$ ),
    eDrop( $I_2$ ), eDrop( $M_{w2,i2,s}$ ), eDrop( $Ss_{s,1}$ ), eDrop( $R_{y1}$ ), reaction buffer 1)
    Actuate e(8, 21, 58, 55)
    Actuate e(5, 74, 19, 44)
    Actuate e(9, 42, 47, 54)
    .....
    Wiring (eDrop( $S_{w1}$ ), eDrop( $S_{w2}$ ))
    Actuate e(20, 21)
    Wiring (eDrop( $I_1$ ), eDrop( $M_{w1,i1,s}$ ), eDrop( $Sub_{2,y1}$ ,  $Dd_{1,y1}$ ),
    eDrop( $I_2$ ), eDrop( $M_{w2,i2,s}$ ), eDrop( $Ss_{s,1}$ ), eDrop( $R_{y1}$ ), eDrop( $S_{w1}$ ,  $S_{w2}$ ),
    reaction buffer 1)
    Actuate e(73, 21, 42, 62)
    Actuate e(75, 18, 45, 59)
    Actuate e(16, 42, 43, 61)
    .....
    Combining (eDrop( $M_{w1,i1,s}$ ), eDrop( $Sub_{2,y1}$ ,  $Dd_{1,y1}$ ), eDrop( $M_{w2,i2,s}$ ),
    eDrop( $Ss_{s,1}$ ), reaction buffer 1)
    Actuate e(18, 43, 59, 54)
    Actuate e(21, 62, 61, 56)
    Actuate e(19, 44, 58)
    .....
    Combining (eDrop( $M_{w1,i1,s}$ ), eDrop( $S_{w1}$ ,  $S_{w2}$ ), eDrop( $I_1$ ), eDrop( $I_2$ ),
    eDrop( $R_{y1}$ ))
    Actuate e(54, 55, 58)
    Actuate e(54, 47, 60)
    Actuate e(56, 59, 62)
    .....
}

```

Script file.txt - Notepad

File Edit View

```

1 2 83 66 1 150 100 10 0
2 2 84 65 1 150 100 10 0
3 3 78 72 64 1 150 100 10 0
4 2 87 63 1 150 100 10 0
5 8 77 21 46 87 63 78 72 64 2 150 160 10 0
6 10 77 21 46 78 72 64 84 65 83 66 2 150 180 10 0
7 3 76 42 62 1 150 100 10 0
8 3 75 43 61 1 150 100 10 0
9 2 13 66 1 150 100 10 0
10 2 12 65 1 150 100 10 0
11 3 11 71 64 1 150 100 10 0
12 2 10 63 1 150 100 10 0
13 8 77 16 46 10 63 11 71 64 2 150 160 10 0
14 10 77 16 46 11 71 64 12 65 13 66 2 150 160 10 0
15 5 34 83 66 94 13 1 150 120 10 0
16 1 28 1 150 100 10 0
17 2 27 35 1 150 100 10 0
18 1 26 1 150 100 10 0
19 1 25 1 150 100 10 0
20 3 26 25 24 2 150 120 10 0
21 7 33 34 28 27 26 24 35 2 150 160 10 0
22 1 23 1 150 100 10 0
23 1 22 1 150 100 10 0
24 1 39 1 150 100 10 0
25 1 40 1 150 100 10 0
26 1 41 1 150 100 10 0
27 1 34 1 150 100 10 0
28 1 28 1 150 100 10 0
29 2 27 35 1 150 100 10 0
30 1 26 1 150 100 10 0
31 1 25 1 150 100 10 0
32 3 26 25 24 2 150 120 10 0
33 7 33 34 28 27 26 24 35 2 150 160 10 0
34 1 23 1 150 100 10 0
35 1 22 1 150 100 10 0
36 1 39 1 150 100 10 0
37 1 40 1 150 100 10 0
38 2 34 41 1 150 150 10 0
39 2 28 36 1 150 150 10 0
40 2 36 26 1 150 150 10 0
41 2 26 25 1 150 150 10 0
42 2 25 24 1 150 100 10 0
43 3 26 25 24 2 150 150 10 0
44 6 24 26 36 28 34 33 2 150 150 10 0
45 1 37 1 150 100 10 0
46 1 38 1 150 100 10 0
47 1 39 1 150 100 10 0
48 4 83 66 94 13 1 150 120 10 0
49 1 84 1 150 100 10 0
50 2 85 86 1 150 100 10 0
51 1 87 1 150 100 10 0
52 5 85 87 89 4 86 2 150 160 10 0
53 6 83 84 85 89 86 4 2 150 160 10 0
54 9 89 4 77 75 16 43 46 61 39 2 150 160 10 0
55 4 88 74 44 60 1 150 120 10 0
56 4 5 76 17 62 1 150 120 10 0
57 4 90 73 45 59 1 150 120 10 0
58 4 7 75 18 61 1 150 120 10 0
59 4 6 16 46 58 1 150 120 10 0
60 5 8 74 19 60 39 1 150 120 10 0
61 4 3 17 62 47 1 150 120 10 0
62 4 9 73 20 59 1 150 120 10 0
63 4 4 18 61 56 1 150 120 10 0
64 4 7 7 16 21 58 1 150 120 10 0
65 5 5 19 39 60 55 1 150 120 10 0
66 4 7 6 17 42 47 1 150 120 10 0
67 4 7 20 59 54 1 150 120 10 0
68 4 7 5 18 43 56 1 150 120 10 0
69 1 94 1 150 100 10 0
70 1 93 1 150 100 10 0
71 1 2 1 150 100 10 0
72 1 91 1 150 100 10 0
73 4 91 4 2 93 2 150 100 10 0
74 4 4 2 93 94 2 150 120 10 0
75 4 8 21 58 55 1 150 120 10 0
76 4 5 74 19 44 1 150 120 10 0
77 4 9 42 47 54 1 150 120 10 0
.....

```

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

Fig. S10| The instruction set and corresponding script file for the automatic on-chip DNA neuron computation.

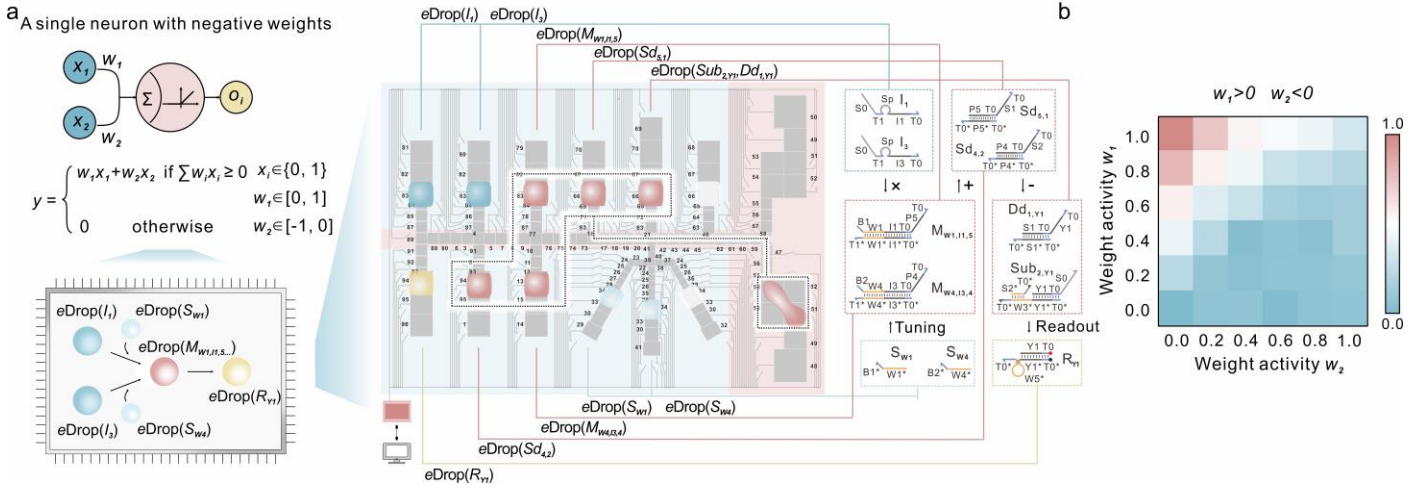


Fig. S11| An on-chip DNA neuron with negative weights. **a**, The on-chip implementation. Each droplet corresponds to the specific DNA species, indicated by different colors. The encapsulated species are shown in the colored dashed boxes, with arrows indicating the flows of the strand displacement reactions within the large droplet (2.75 μL) formed in the reaction chamber. The electrodes are indicated by the number-coded blocks, in which the number identifies the position of electrodes on the chip. The black dotted box shows that five molecular chiplets are merged into one large droplet that perform neuron computation. $e\text{Drop}(M_{W1,11,5})$ and $e\text{Drop}(M_{W4,13,4})$ are designed to implement two parallel weight multiplications. The negative weights can be implemented using distinct output sequence for the negative weight substrate molecules. Two addition chiplets $e\text{Drop}(Sd_{5,1})$ and $e\text{Drop}(Sd_{4,2})$ are designed for the individual sums of negative and positive weights, followed by one subtraction chiplet $e\text{Drop}(Sub_{2,Y1},Dd_{1,Y1})$ to complete effective subtraction of two weighted sums. Two inputs x_1 and x_2 are encoded using two droplets, $e\text{Drop}(I_1)$ and $e\text{Drop}(I_3)$. Two weight modulator chiplets $e\text{Drop}(S_{W1})$ and $e\text{Drop}(S_{W4})$ are designed to modify the connection weights (w_1 and w_2) of neuron independently. Additionally, two reaction buffer droplets are employed (white colored droplet) to maintain consistent DNA molecule concentration throughout the reaction process. We implement a specific instruction set to automate a series of precise droplet manipulations (including aliquoting, transporting, merging, and mixing), thereby integrating the required molecular chiplets into one droplet form within the reaction chamber and performing the on-chip DNA neuron computation. The values of variables x_i and w_i correspond to the concentrations of DNA species I_i and S_{Wi} within the large droplet formed in the reaction chamber, respectively. The volume of droplets located in the memory is 2.50 μL (DNA species I_i and $M_{Wi,li,j}$ concentration: 22 $\times$, DNA species $Sd_{j,k}$, $Dd_{k,Yi}$ and R_{Yi} concentration: 33 $\times$, DNA species $Sub_{j,Yi}$ concentration: 66 $\times$), while the volume of $e\text{Drop}(S_{Wi})$ is 1.00 μL (DNA species S_{Wi} concentration: 5.5 $\times$). Image of the fabricated chip is shown in Fig. S5. **b**, Characterization of the DNA neuron circuitry on chip for different combinations of two weights ($x_1=1$, $x_2=1$, measured after 12 h since the final input was added). The x- and y- axis represent the absolute value of weights ($|w|=0.0, 0.2, 0.4, 0.6, 0.8$ or 1) implemented using the weight modulator chiplets, with colours indicating the corresponding output signal intensity. The color scale (0-1) represents the normalized analog fluorescence outputs mapped to the corresponding theoretical output values for the tested weight combinations. The upper limit of 1 is determined by the maximum theoretical output values accessible in this experiment. The standard concentration is 50 nM (1 $\times$).

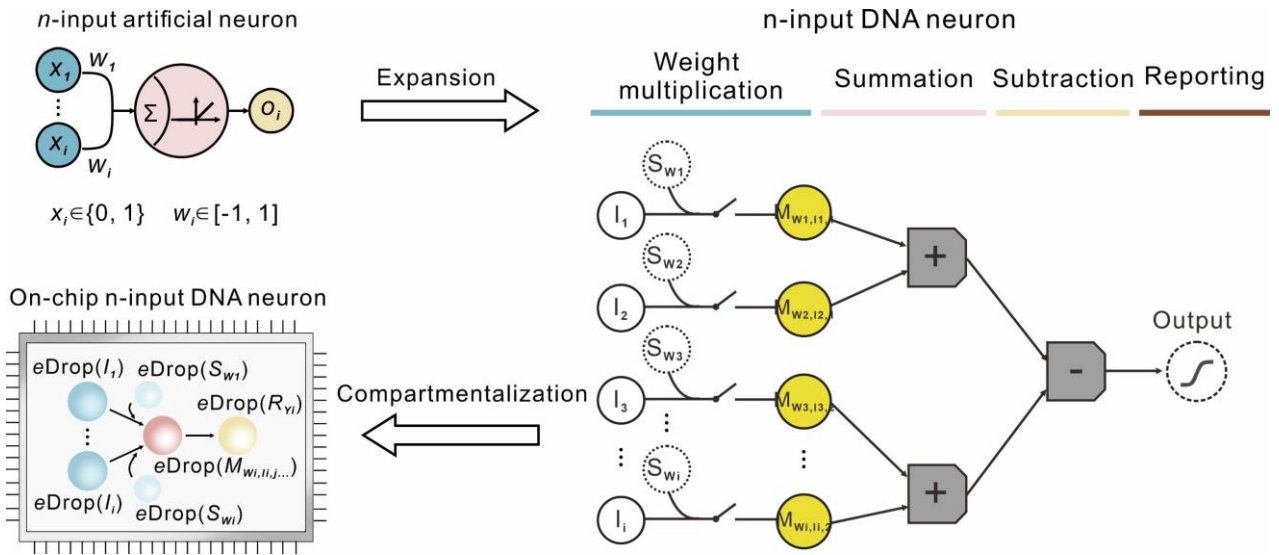


Fig. S12| The on-chip implementation of a DNA neuron with n inputs and one output. In general, using this integrated design, an on-chip DNA neuron capable of responding to n inputs can be implemented by encapsulating up to $n+4$ molecules within a single droplet $eDrop(M_{w_i, I_i, j} \dots)$ (where n weight substrate molecules $M_{w_i, I_i, j}$ for parallel weight multiplication for n inputs, two summation gates $S_{d_j, k}$ for the individual sums of positive and negative weights, one annihilation species Sub_{n, Y_i} and one duplex Dd_{k, Y_i} for subtraction, are required). In addition, n inputs x_i can be encoded using n input chiplets $eDrop(I_i)$, with n weight modulator chiplets $eDrop(S_{w_i})$ to modify the connection weights w_i independently. One designated reporter chiplet $eDrop(R_{y_i})$ is used for signal readout on chip.

S4: Integration of Multiple DNA Neurons on a DMIC

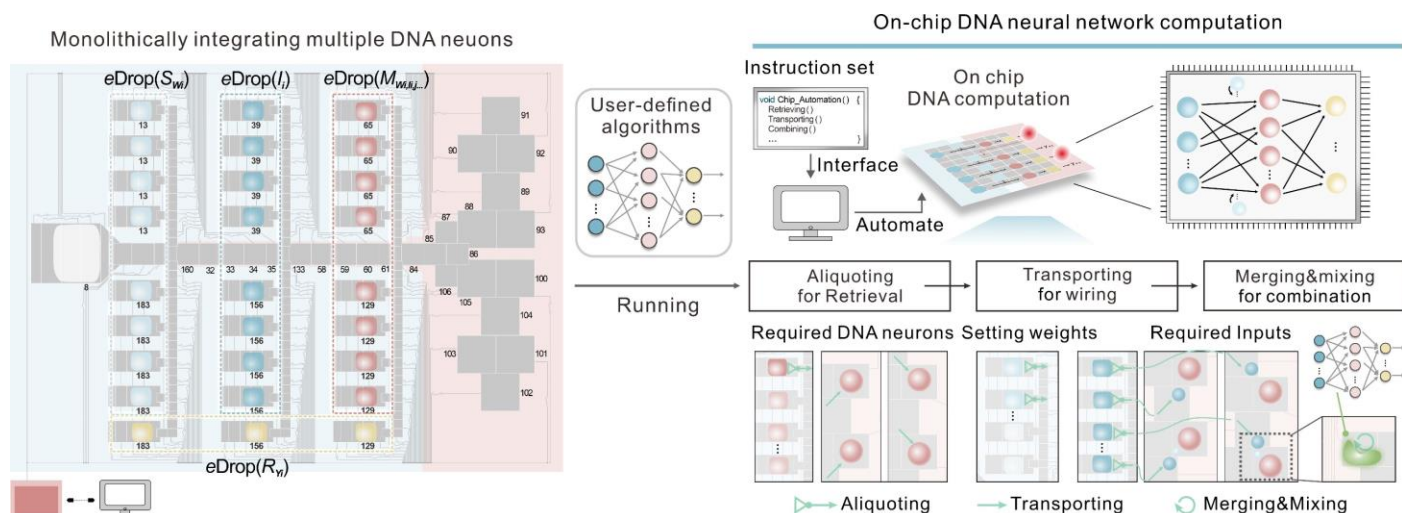


Fig. S13| DMIC enables the system-level integration of DNA neurons into on-chip DNA neural network. For on-chip implementation, each neuron in a layer is instantiated as an electrode-addressable molecular chiplet (e.g., $eDrop(I_i)$ serves as an input neuron), so droplets act as independently addressable network nodes. Inter-neuronal connections are created physically by routing the appropriate droplets from the reservoir into a reaction chamber and fusing them into one large droplet (indicated by green circle): upon activation of a predefined electrode array, the required molecular chiplets are precisely moved and merged (green circle), co-localizing the molecular components in a droplet microcontainer that establish the network's physical wiring via strand-displacement cascades. A set of weight modulator chiplets $eDrop(S_w)$ can be independently configured and controlled to flexibly adjust connection strength between neurons. Stored molecular chiplets ($eDrop(v)$) in the reservoir can be selectively accessed to assemble and reconfigure network topologies automatically, enabling flexible, on-demand construction of complex DNA neural networks.

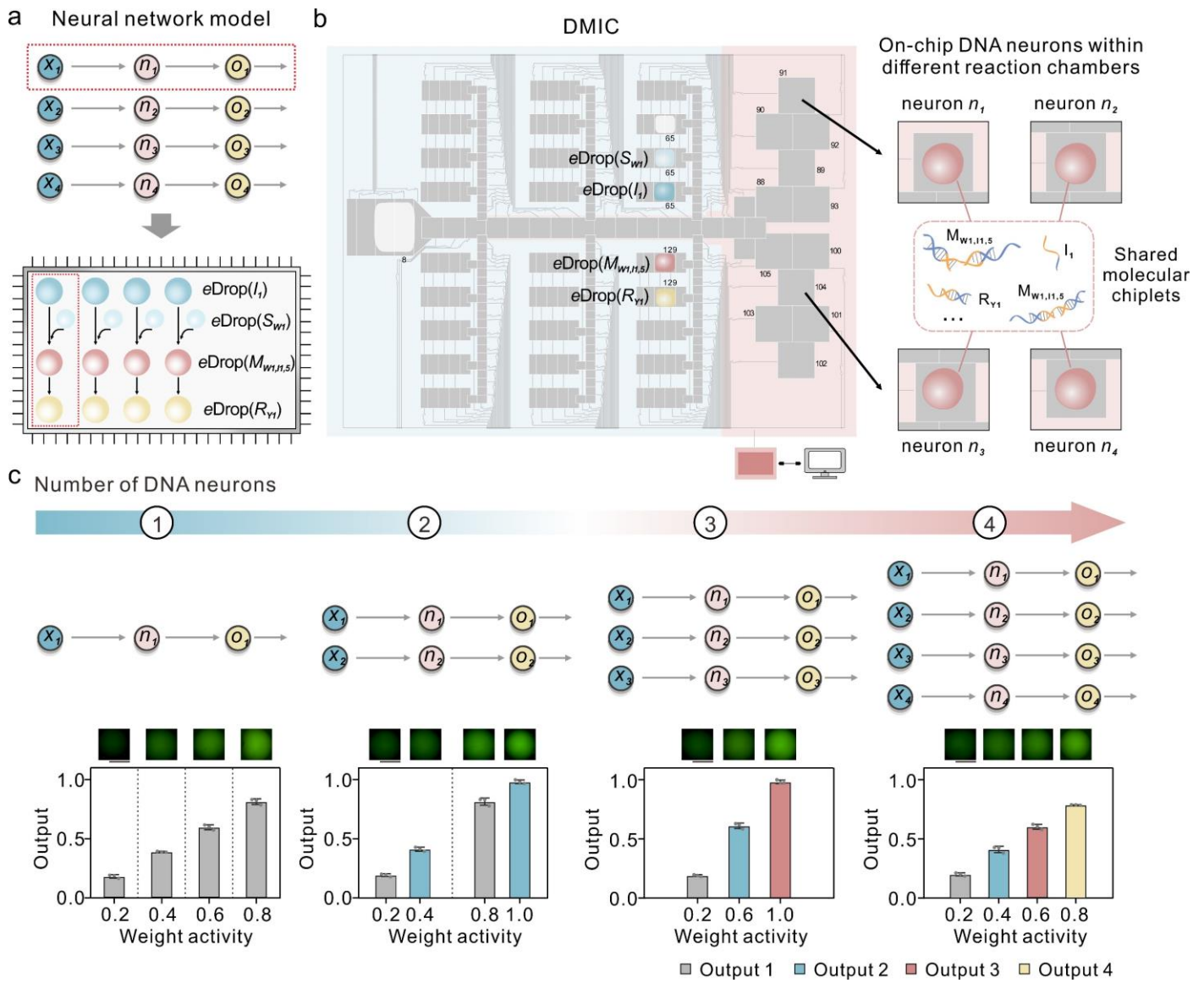


Fig. S14| The implementation of different DNA neurons within one DMIC. **a**, Schematic of the neuron network architecture and its on-chip implementation, each neuron of which responds to one input. **b**, Design of DMIC used here. Four molecular chiplets are designed for the on-chip implementation of distinct DNA neuron computation on chip, and each droplet corresponds to the specific DNA species. Note that $eDrop(M_{w_i, l_{i,j}, \dots})$ contains the weight substrate molecule $M_{w_1, l_{1,5}}$ for weight multiplication for one input, one summation gates $Sd_{5,1}$ for the sums of weighted inputs, one annihilation species $Sub_{2,Y1}$ and one duplex $Dd_{1,Y1}$ for subtraction. Additionally, two reaction buffer droplets (white colored droplet) are employed to maintain consistent DNA molecule concentration throughout the reaction process. As we can isolate each DNA neuron computation with physical separation by droplets in different reaction chambers, these different on-chip DNA neurons can share the same DNA implementation (dashed pink box) to keep the overall chemical complexity low, enabling different neurons (n_1, n_2, n_3 and n_4) to be encoded by one identical chiplet $eDrop(M_{w_i, l_{i,j}, \dots})$ but with different spatial coordinates (different reaction chambers) in the processor. **c**, Characterization of four distinct neuron networks on chip for different combinations of weights ($x_i=1$, measured after 12 h since the input was added). Error bars indicate standard deviations (mean $\pm$ s.d., $n = 3$). Fluorescence microscopy images are shown along the top, with scale bars of 1 mm.

n -neuron network	Experimentally implemented function	Molecular chiplets retrieved from memory	Configuration of droplet network
Two-neuron network 	X_1, X_2, X_3 AND-AND y X_1, X_2, X_3 AND-OR y X_1, X_2, X_3 OR-AND y X_1, X_2, X_3 OR-OR y	$eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,1}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,2}) (1)$ $eDrop(I_i) (1)$ $eDrop(R_{in}) (1)$	
Three-neuron network 	X_1, X_2, X_3, X_4 AND-AND-AND y X_1, X_2, X_3, X_4 AND-OR-AND y	$eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,1}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,2}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,3}) (1)$ $eDrop(I_i) (1)$ $eDrop(R_{in}) (1)$	
Four-neuron network 	Three-layer circuit with four AND/OR gates X_1, X_2, X_3, X_4, X_5 X_1, X_2, X_3, X_4, X_5	$eDrop(I_i) (1)$ $eDrop(S_{in}) (10)$ $eDrop(M_{out,1}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,2}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,3}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,4}) (1)$ $eDrop(I_i) (1)$ $eDrop(R_{in}) (1)$	
Eight-neuron network 	Three-layer circuit with eight AND/OR gates $X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8$ $X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8$	$eDrop(I_i) (2)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,1}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,2}) (1)$ $eDrop(I_i) (2)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,3}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,4}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,5}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,6}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,7}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,8}) (1)$ $eDrop(R_{in}) (1)$ $eDrop(R_{in}) (1)$	
Ten-neuron network 	A square-root circuit $X_1^0, X_2^0, X_3^0, X_4^0, X_5^0, X_6^0, X_7^0, X_8^0, X_9^0, X_{10}^0$ $X_1^0, X_2^0, X_3^0, X_4^0, X_5^0, X_6^0, X_7^0, X_8^0, X_9^0, X_{10}^0$	$eDrop(I_i) (4)$ $eDrop(S_{in}) (3)$ $eDrop(M_{out,1}) (3)$ $eDrop(I_i) (3)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,2}) (1)$ $eDrop(I_i) (2)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,3}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,4}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,5}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,6}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,7}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,8}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,9}) (1)$ $eDrop(I_i) (1)$ $eDrop(S_{in}) (1)$ $eDrop(M_{out,10}) (1)$ $eDrop(R_{in}) (2)$ $eDrop(R_{in}) (2)$	

This table is continued on the following page.

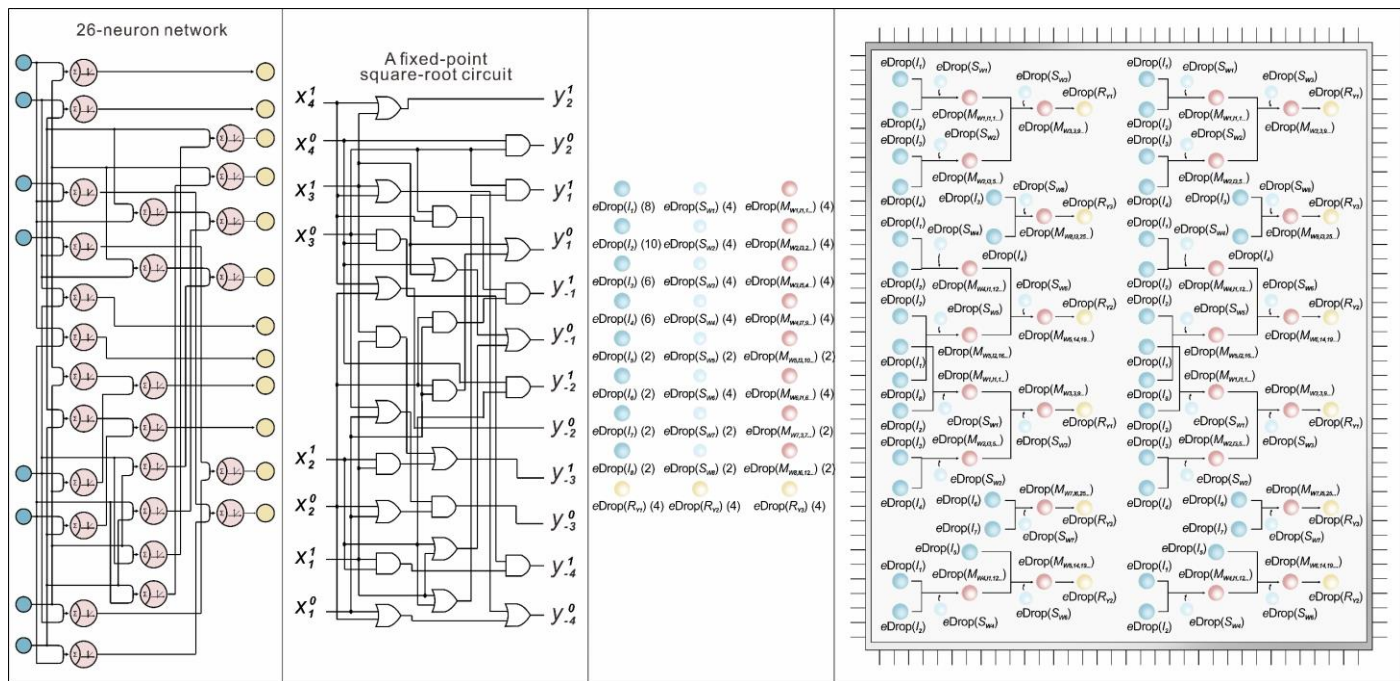


Fig. S15| Representative multi-DNA neuron network experimentally implemented on a DMIC via programmable routing of molecular chiplets. Each neuron corresponds to a specific molecular chiplets $eDrop(v)$, with corresponding weight modulator chiplets $eDrop(S_{wi})$ to adjust the connection strength between neurons. For each network, the required number of aliquoted chiplets, retrieved from memory, is listed in parentheses. The volume of droplets located in the memory is 2.50 μL (DNA species I_i , $M_{wi,li,j}$, $S_{d,j,k}$ and $Fuel_{k,l}$ concentration: 50 $\times$, DNA species $F_{k,Yn}$ and R_{Yn} concentration: 25 $\times$, DNA species Th_k and S_{wi} concentration: 12.5 $\times$). The volume of droplets aliquoted from the memory is 0.32 μL , while the volume of aliquoted reaction buffer droplet is 1.60 μL . Note that the concentration of DNA species I_1 for on-chip 26-DNA neuron network is 100 $\times$. The standard concentration is 50 nM (1 $\times$).

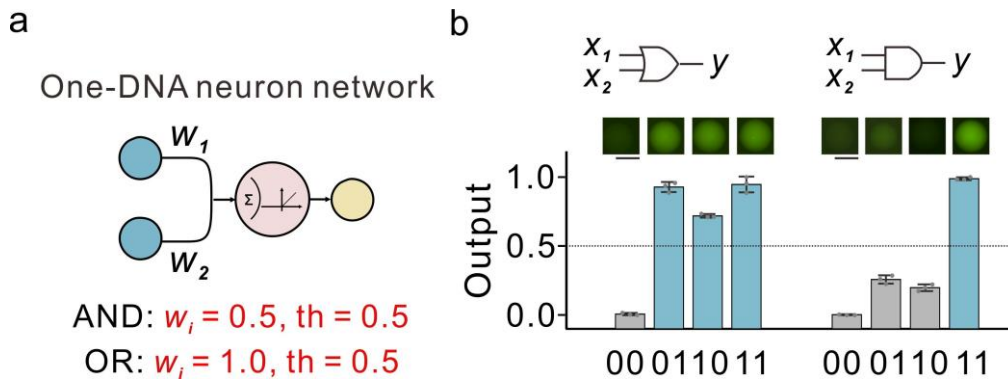
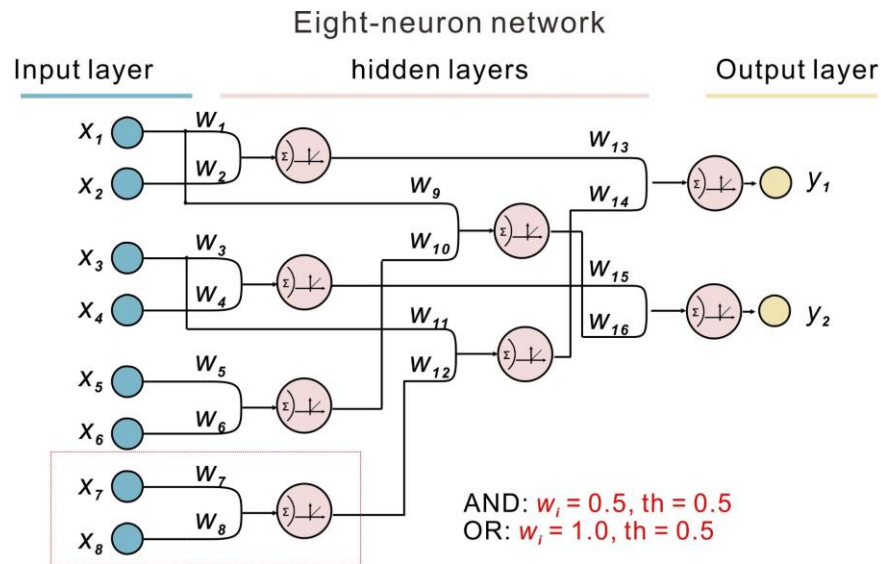


Fig. S16| Experimental characterization of the one-DNA neuron network on a DMIC. The one-DNA neuron network (**a**) can be programmed to either a linearly separable two-input OR or AND logic gate by adjusting the connection weights, with their characterization of the on-chip DNA circuit behaviour of 4 input combinations after 12 h since the inputs were added (**b**). Input strands were at 0× (0, logic OFF) or 1× (1, logic ON). Error bars indicate standard deviations (mean ± s.d., n = 3). Fluorescence microscopy images are shown along the top, with scale bars of 1 mm.

a



b

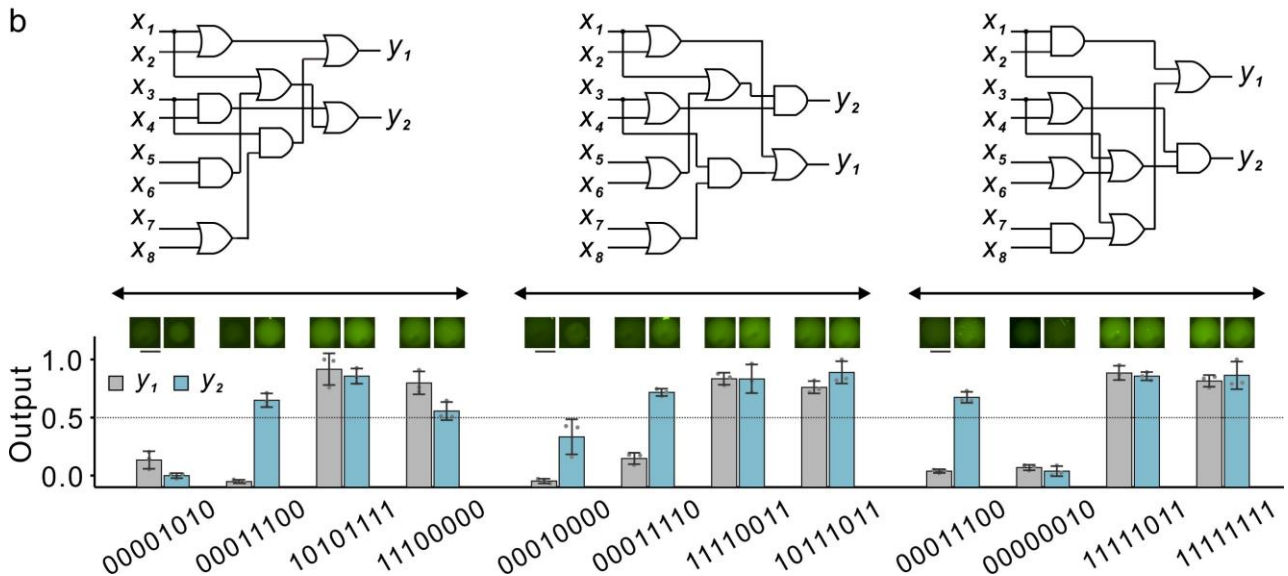


Fig. S17| Experimental characterization of the eight-DNA neuron network on DMIC. An on-chip eight-DNA neuron network (a) can be programmed to compute cascaded logic functions, with their characterization of the on-chip DNA circuit behaviour of 4 input combinations after 12 h since the inputs were added (b). Note that the required input chiplets $eDrop(I_i)$ were first merged and then routed to the target reaction chamber to minimize timing variability in input delivery. The weights of each individual neuron in the network can be independently tuned to change their propensity to fire any given inputs, enabling a network to adapt to different functions. Input strands were at $0\times$ (0, logic OFF) or $1\times$ (1, logic ON). Error bars indicate standard deviations (mean $\pm$ s.d., $n = 3$). Fluorescence microscopy images are shown along the top, with scale bars of 1 mm.

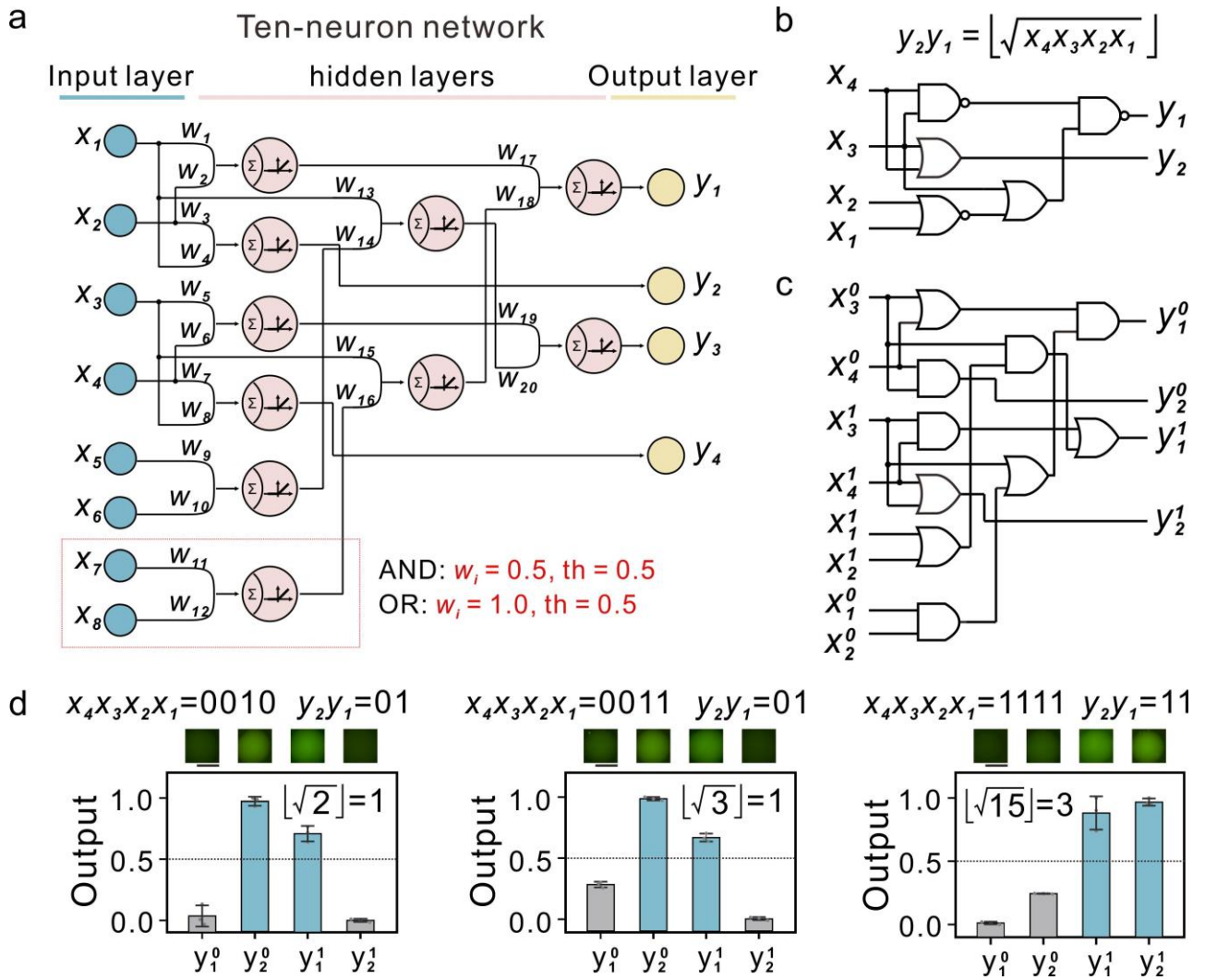
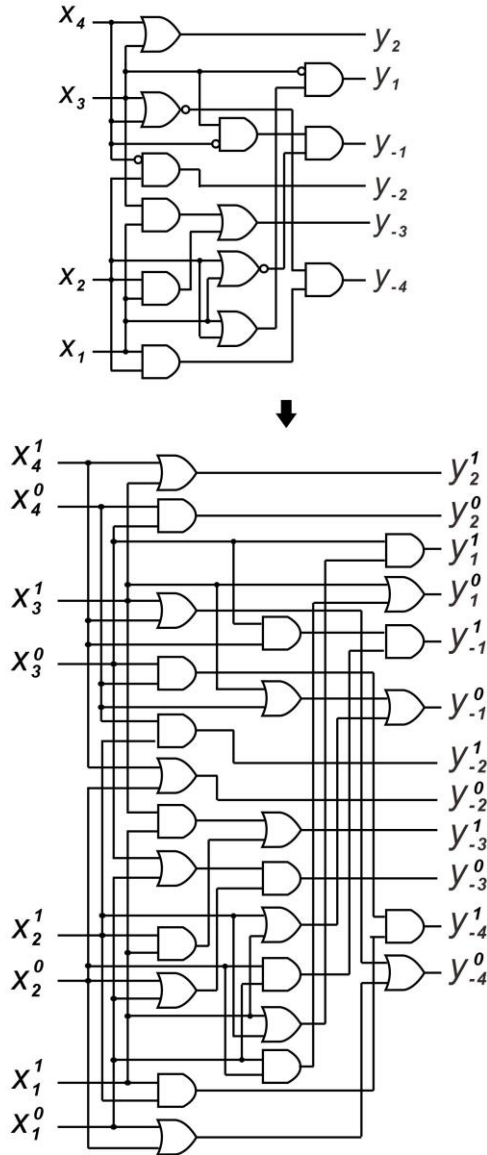


Fig. S18| On-chip ten-DNA neuron network and its experimental characterization. **a**, An on-chip ten-neuron network that is programmed to the equivalent dual-rail circuit. **b**, A digital circuit that performs the 4-bit square root computation. **c**, A dual-rail circuit that is equivalent to the square-root circuit. x_i^0 (or y_i^0) and x_i^1 (or y_i^1) are dual-rail inputs (or outputs) of x_i (or y_i), which represent logic '0' and '1', respectively. **d**, Fluorescence level of output at 12 h after the inputs were added. Note that the required input chiplets $eDrop(I_i)$ were first merged and then routed to the target reaction chamber to minimize timing variability in input delivery. Input strands were at $0\times$ (0, logic OFF) or $1\times$ (1, logic ON). Error bars indicate standard deviations (mean $\pm$ s.d., $n = 3$). Fluorescence microscopy images are shown along the top, with scale bars of 1 mm.

a

$$\frac{y_2 y_1 \cdot y_{-1} y_{-2} y_{-3} y_{-4}}{\text{Ones} \quad \text{Tenths}} = \sqrt{x_4 x_3 x_2 x_1}$$



b

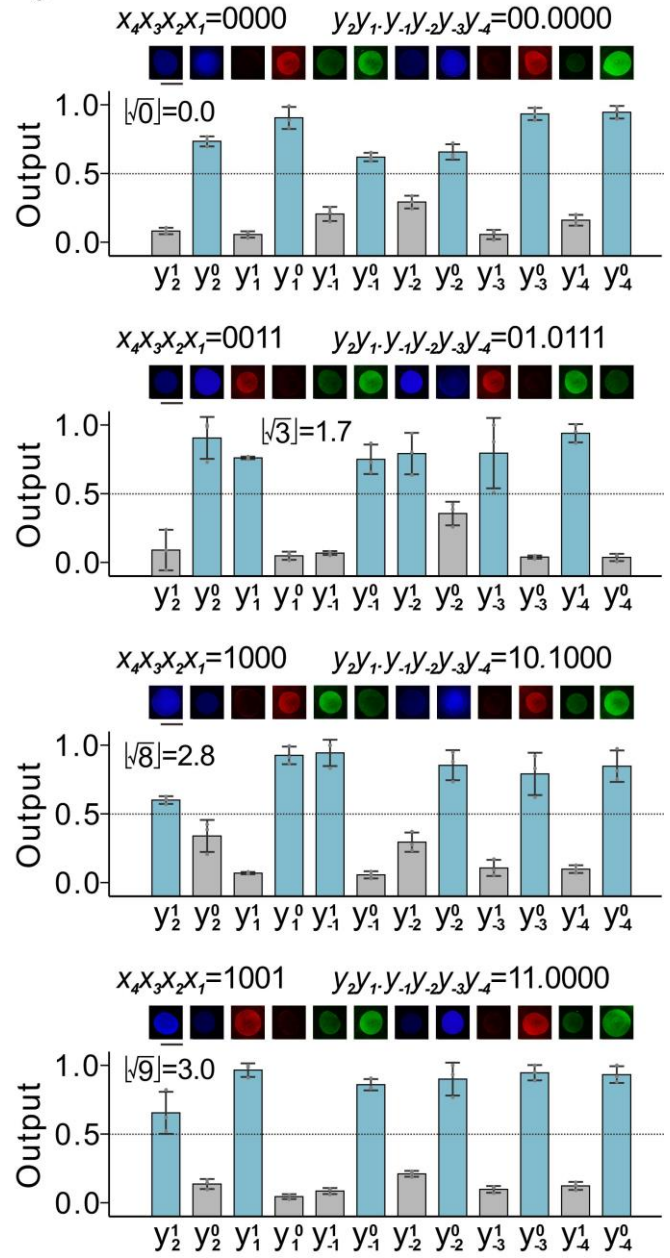


Fig. S19| On-chip 26-DNA neuron network and its experimental characterization. **a**, A digital circuit and an equivalent dual-rail circuit that performs the fixed-point square root computations. x_i^0 (or y_i^0) and x_i^1 (or y_i^1) are dual-rail inputs (or outputs) of x_i (or y_i), which represent logic '0' and '1', respectively. **b**, Fluorescence level of output at 12 h after the inputs were added. Note that the required input chiplets $eDrop(I_i)$ were first merged and then routed to the target reaction chamber to minimize timing variability in input delivery. Input strands were at $0\times$ (0, logic OFF) or $1\times$ (1, logic ON). Error bars indicate standard deviations (mean $\pm$ s.d., $n = 3$). Fluorescence microscopy images are shown along the top, with scale bars of 1 mm.

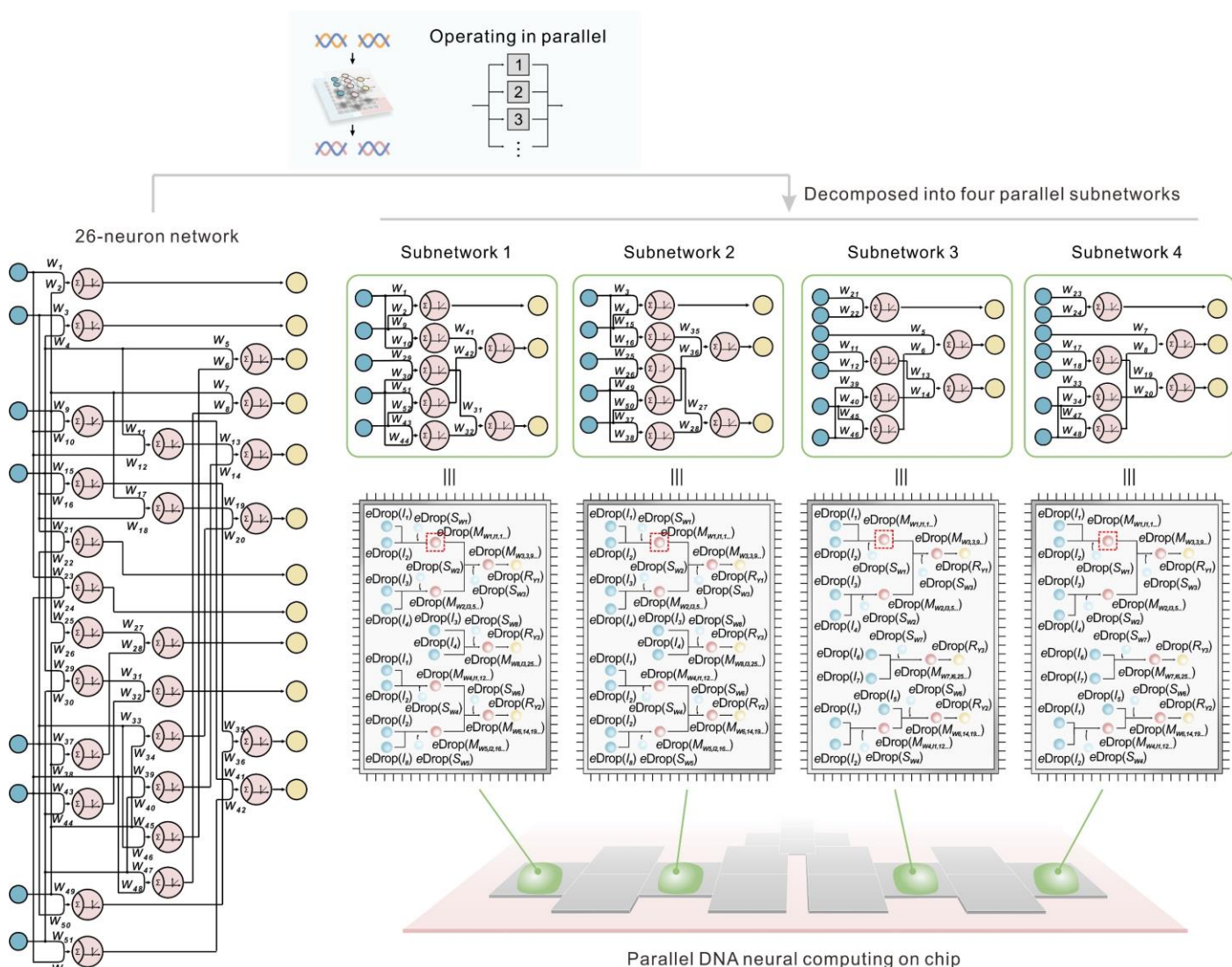


Fig. S20| DMIC enables DNA neural computation in a parallel manner. Take 26-DNA neuron network that performs the fixed-point square root computations for example, this network is decomposed into four parallel subnetworks—two seven-neuron subnetworks and two six-neuron networks—each sharing the same architecture but programmed with distinct weight parameters. These four subnetworks execute in four separate chambers in parallel, each generating three outputs (12 in total), which are then combined to yield the final computation of the full network. That is, our DMIC allows the DNA neurons of a large network to be divided into multiple groups, each physically isolated in a separate chamber during operation, thereby parallelizing computation across groups. In this way, the number of DNA strands involved in each reaction volume is vastly reduced, thereby minimizing potential crosstalk between reactions. Moreover, the subnetworks in different chambers can share the same DNA neurons (highlighted by the dashed red boxes), enable sequence reuse and simplify molecular design.

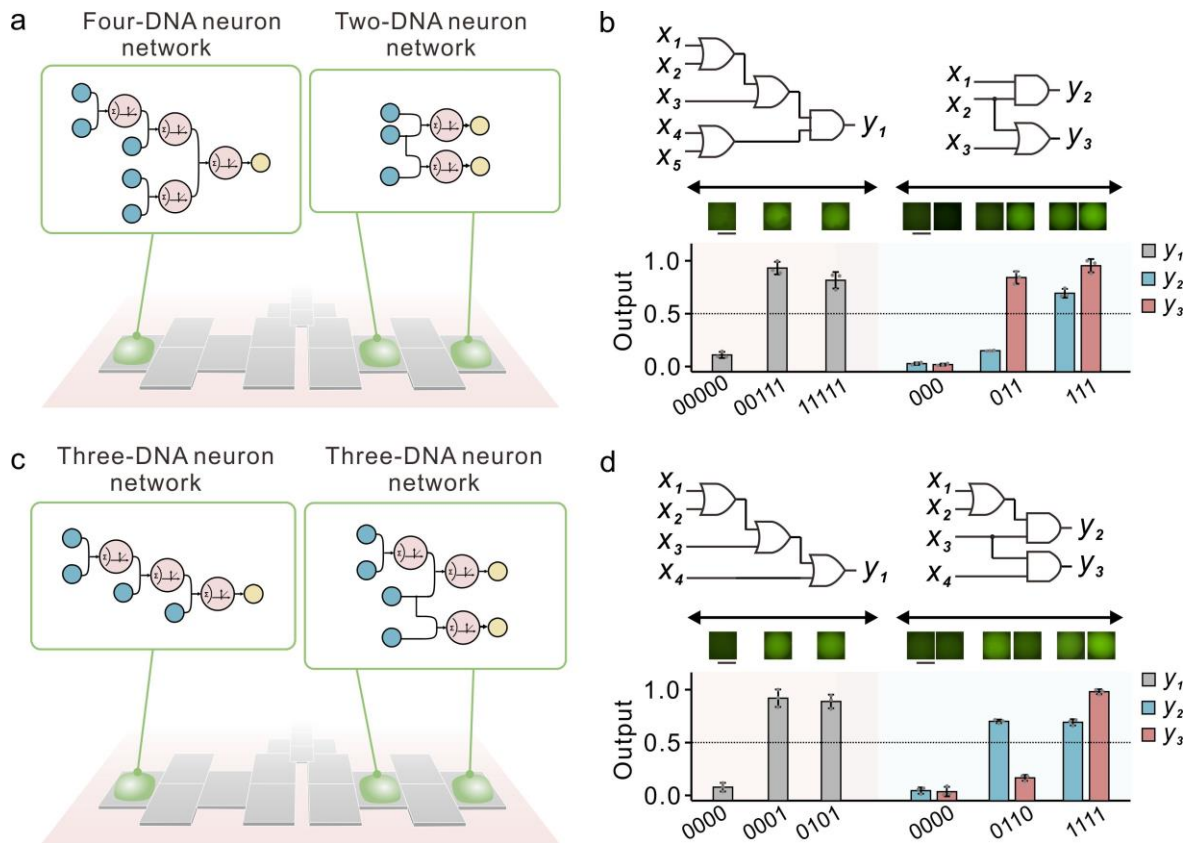


Fig. S21| Integration of multiple multi-DNA neuron networks on a DMIC. **a,c**, Diagram of the concurrent deployment of multiple different multi-DNA neuron networks on a DMIC. Note that two-DNA neuron network shown in **a** and three-DNA neuron network shown in **c** can both be divided into two parallel subnetworks that run synchronously in different reaction chambers. **b,d**, Experimental validation. These parallel multi-DNA neuron networks can be programmed to compute distinct cascaded logic functions, depending on the initial value of weights. Fluorescence level of output at 12 h after the inputs were added. Note that the required input chiplets $e\text{Drop}(I_i)$ were first merged and then routed to the target reaction chamber to minimize timing variability in input delivery. Input strands were at $0\times$ (0, logic OFF) or $1\times$ (1, logic ON). Error bars indicate standard deviations (mean $\pm$ s.d., $n = 3$). Fluorescence microscopy images are shown along the top, with scale bars of 1 mm.

It is worth noting that our previous constructed DMIC were based on passive-matrix digital microfluidics (PM-DMF), which employ a grid of large electrodes, each wired to an individual control line to manipulate molecular chiplets (Fig. S2, S13, and S31b). However, this wiring designs constrains electrode density and thereby limits the number of independently addressable molecular chiplets and the achievable DNA neuron-integration scale⁵⁻⁹. We therefore developed a large-scale DMIC based on active-matrix digital microfluidic (AM-DMF) featuring 720×1280 active electrode array (Fig. S22). Each pixel is individually addressable by scanning the row and column control signals, enabling the parallel configuration of hundreds of molecular chiplets to support substantially higher neuron integration density.

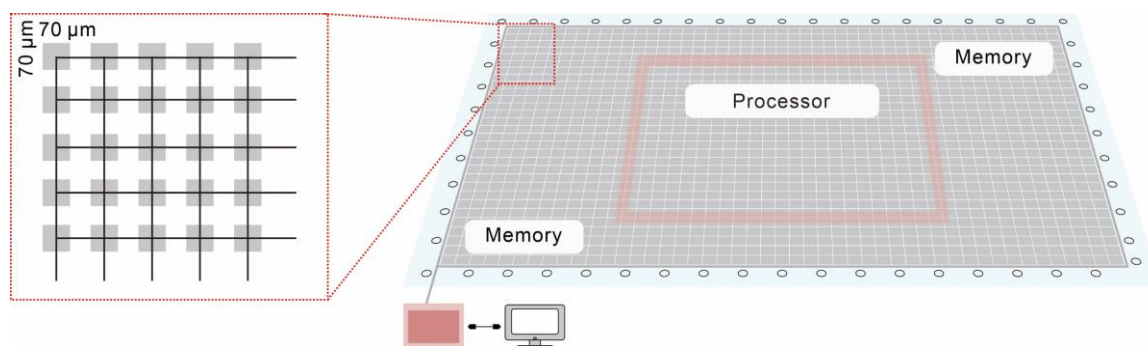


Fig. S22| Chip hardware design for the on-chip DNA-based 144-neuron ConvNet computation. The gray circles around the chip indicate the liquid inlets for loading molecular chiplets. The central pink box denotes the processor. The gray squares represent active electrodes. A customer-written control software DM Ctrl V3.5.2 was used for the automatic execution of chip, in which the instruction set is first translated into the script file, and then uploaded to the software for automatic and sequential activation of electrodes.



Fig. S23| Determination of 36 input patterns. Top row: All original symbols were rescaled to a 20×20 grid and binarized into 400-bit patterns, with each pixel assigned a value of 1 when exceeding a threshold. Bottom row: The number of 1s in each input pattern.

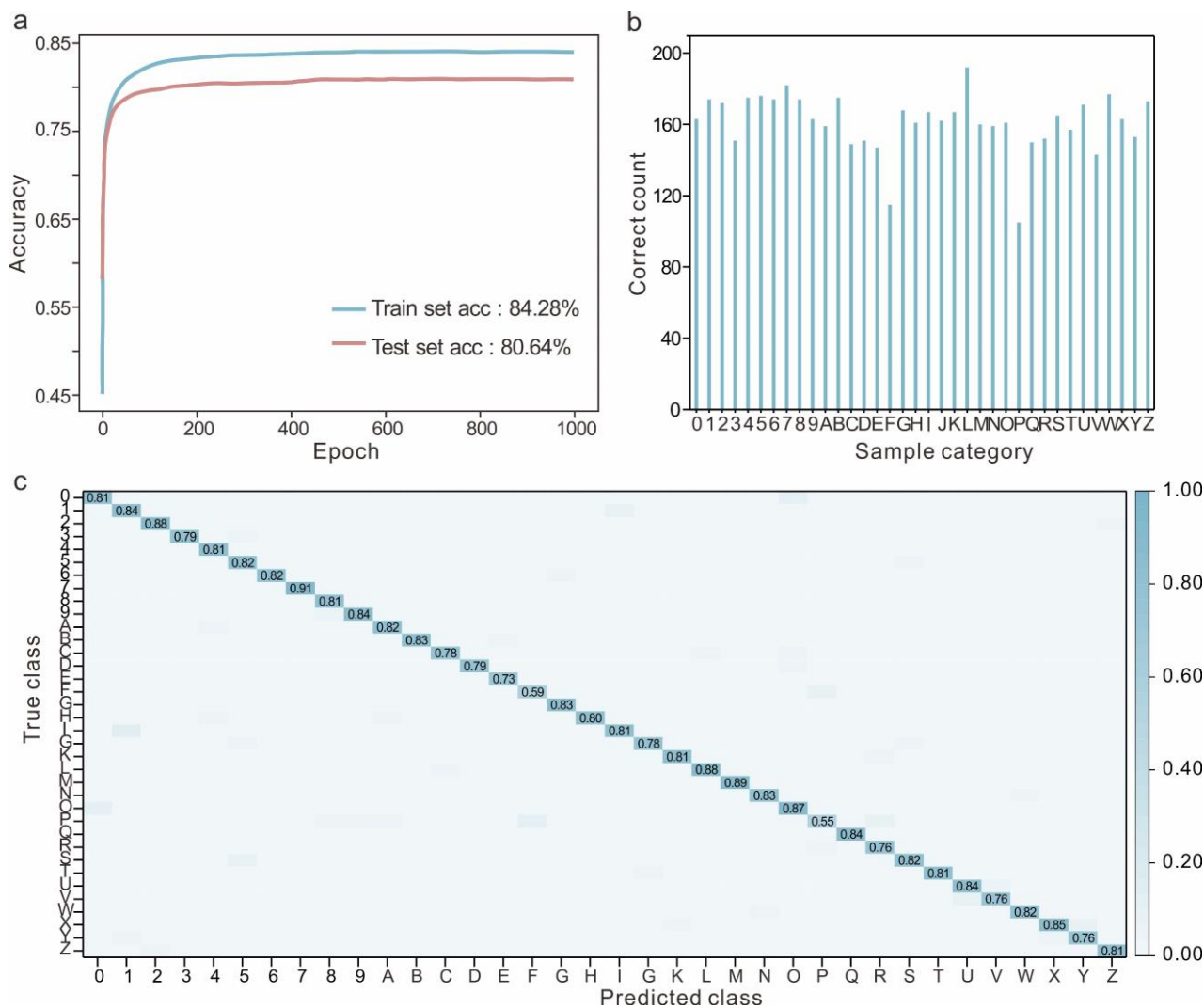


Fig. S24| *In silico* training of a 144-neuron ConvNet for 36-category classification. **a**, Accuracy evolution curves as a function of training epochs derived from *in-silico* 144-neuron ConvNet on EMNIST database, achieving classification accuracy of 84.28% on training set (blue) and 80.64% on test set (red). **b**, Results for 36-category classification task on test set. **c**, The confusion matrix for trained 144-neuron ConvNet on the test set (rows = predicted class, columns = true class; values are normalized per row).

Number of species		Number of DNA-encoded droplets	
Input, I_i	$b n = 29\sim100 400$	$e\text{Drop}(I_i)$	$b n = 29\sim100 400$
Weight, S_{wt}	$e = 200$	$e\text{Drop}(S_{wt})$	$e\text{Drop}(1S_{wt})$ $e = 200$
weight, $M_{wt,li,j}$	$514 \leq 8e$		$e\text{Drop}(2S_{wt})$ $e = 200$
Summation gate, $Sd_{j,k}$	$2m = 8$		
Subtraction gate, $Dd_{k,l}$	$m = 4$		
Subtraction gate, $sub_{n,yi}$	$m = 4$	$e\text{Drop}(M_{wt,li,j...})$	$m = 4$
Conversion gate, $T_{i,yi}$	$m = 4$	$e\text{Drop}(T_{i,yi})$	1
Reporter, R_{yi}	1	$e\text{Drop}(R_{yi})$	1
Total	764~836 1121	Total	144~342 506

n , number of bits in a pattern
 b , number of 1s in a pattern
 m , number of patterns
 e , number of bits in a convolution kernel

Fig. S25| The number of distinct species required on the DMIC for 144-neuron ConvNet computation. Left column: Number of distinct DNA species required for the implementation of 144-neuron ConvNet on a DMIC. For the total number of species, the two values correspond to the number of species for a specific number of inputs (left) and for all possible inputs (right). Right column: Number of distinct molecular chiplets for the on-chip implementation of 144-neuron ConvNet. For the total number of species, the two values correspond to the number of molecular chiplets for a specific number of inputs (left) and for all possible inputs (right). Because DNA neurons that respond to different local receptive regions are processed independently and in parallel on the chip, the same set of DNA input strands can be reused across spatial positions. Consequently, only 100 distinct DNA input sequences are required to encode arbitrary 400-bit binary input patterns. These input strands are individually encapsulated as 100 input chiplets $e\text{Drop}(I_i)$, which can be retrieved on demand to implement any 400-bit input pattern on-chip. 200 weight modulators were designed and encapsulated as weight modulator chiplets $e\text{Drop}(S_{wt})$ to encode all 10×10 convolutional kernels on-chip. Here, we use a two-level droplet scheme. We prepare two types of $e\text{Drop}(S_{wt})$ that contain identical weight modulators but at different encapsulated concentrations: $e\text{Drop}(1S_{wt})$ implements an incremental weight unit 0.10 and $e\text{Drop}(2S_{wt})$ implements an incremental weight unit 0.01. The effective on-chip weight assignments is realized by routing and fusing the required numbers of $e\text{Drop}(1S_{wt})$ and $e\text{Drop}(2S_{wt})$ into a single large droplet. More importantly, we designed four types of chiplets $e\text{Drop}(M_{wt,li,j...})$, each containing the complete set of molecular components required for a 100-input DNA neuron (red box).

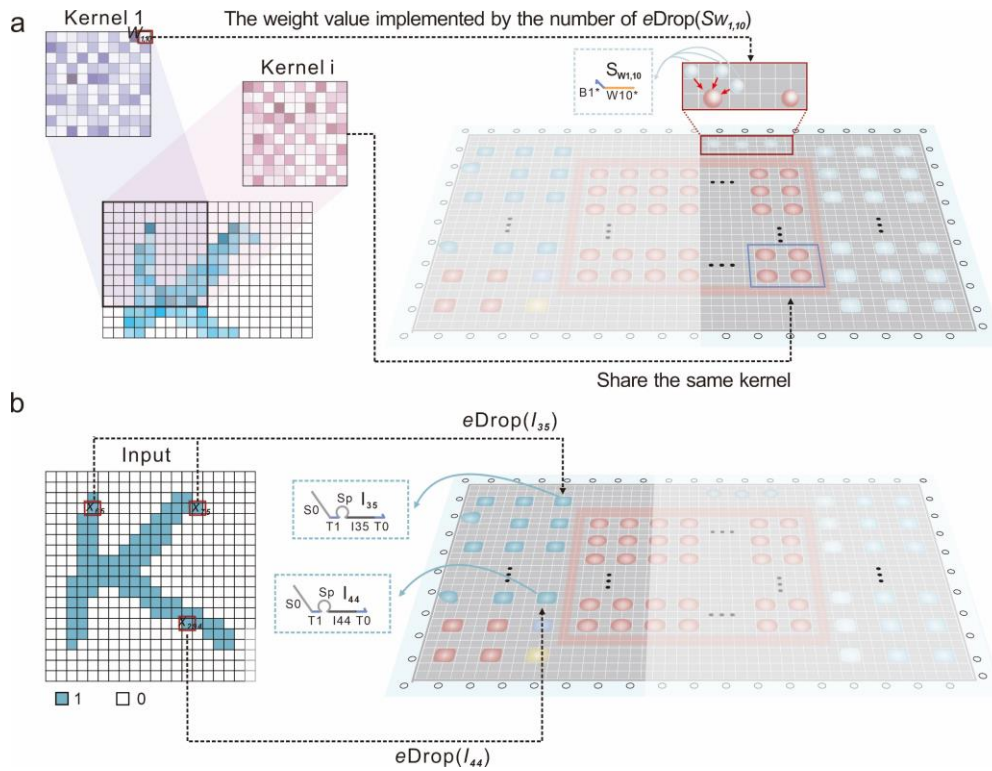


Fig. S26| The on-chip implementation of DNA-based 144-neuron ConvNet for molecular pattern recognition. a, Each pixel w_i in the kernel is encoded using a weight modulator chiplet $eDrop(Sw_i)$ to determine the neuron connection strength, with its values implemented by the number of aliquots of $eDrop(Sw_i)$ merged into the same droplet microcontainer (indicated by red box). DMIC can configure an array of identical weight modulator chiplets and route them to droplets that contain different DNA neurons, thereby implementing weight sharing across neurons. As the network runs, DNA neurons that respond to different local receptive regions operate in parallel at distinct spatial locations, which allows the same set of DNA sequences (encapsulated in individual chiplets) to encode multiple convolutional kernels. According to the kernel weight values, the corresponding number of weight modulator chiplets are routed and merged to set the weights of DNA neurons. In this way, our DMIC not only enables processing different inputs with shared weights, but also supports parallel convolutions using different kernels with shared inputs. **b,** The pixel in each local receptive region can be encoded with an input chiplet $eDrop(I_i)$, and the pixels interacting with the same kernel in different local receptive regions can be implemented with the same input chiplets (for example, x_{65} and x_{75} is programmed with input chiplet $eDrop(I_{25})$).

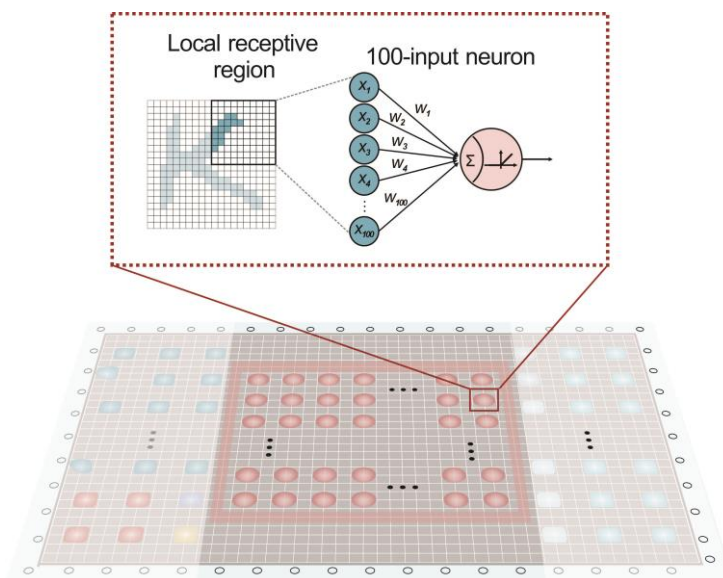


Fig. S27| The on-chip implementation of DNA-based 144-neuron ConvNet for molecular pattern recognition. Each DNA neuron reads 100 inputs from local receptive regions works in parallel within different reaction chambers.

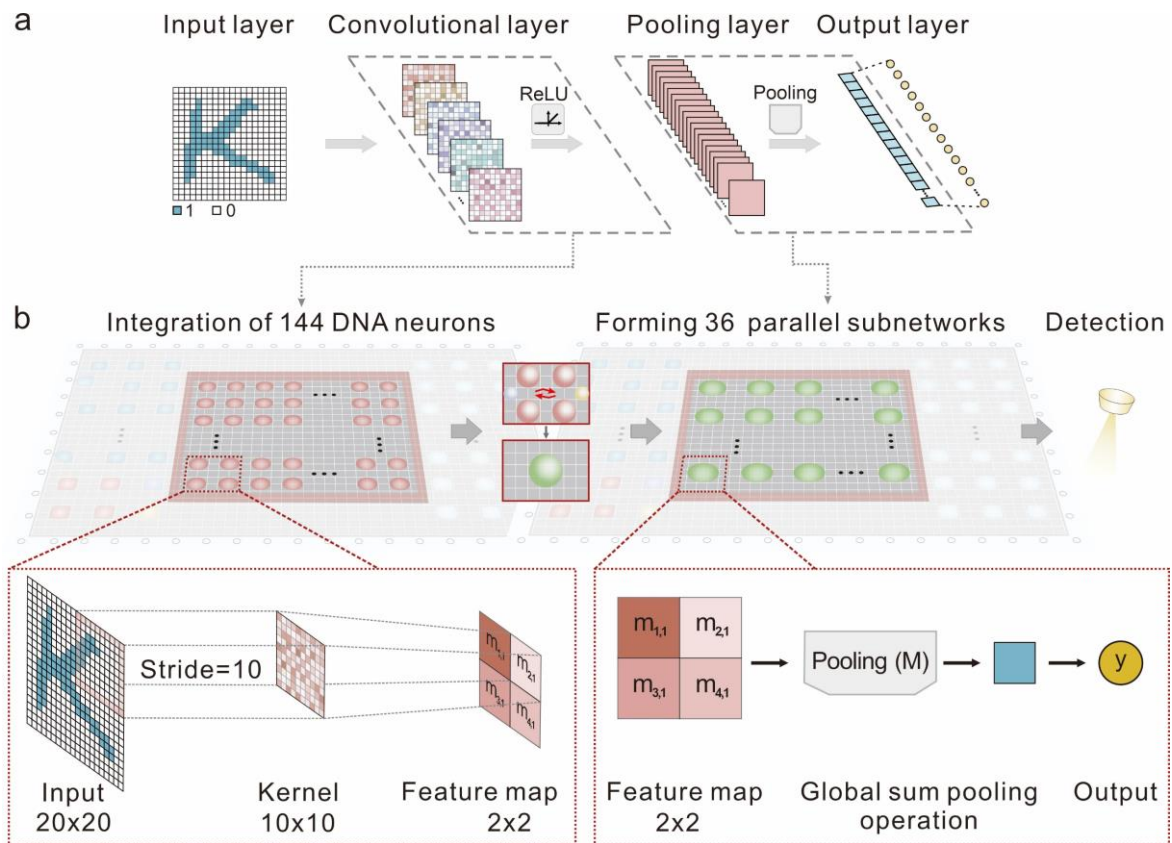


Fig. S28| The on-chip DNA-based 144-neuron ConvNet processing workflow. **a**, The trained 144-neuron ConvNet architecture. **b**, Diagram of the configuration of the DNA-based 144-neuron ConvNet on a DMIC. The engineered DNA neurons are encapsulated individually into droplets, which are aliquoted and routed on-chip to their assigned locations to form an array of 144 molecular chiplets. Each droplet contains a 100-input DNA neuron corresponding to a 10×10 local receptive region and implements a designated convolutional kernel to produce the local feature response for that patch. The large-scale integration of 144 DNA neurons implements a convolutional layer composed of 36 distinct 10×10 kernels, with each kernel instantiated at four spatial positions ($4 \times 36 = 144$) to compute a convolution on chip (the pink dotted box on the left). These 144 aggregated droplets were incubated at room temperature for 24 h, enabling DNA strand displacement cascades to generate outputs corresponding to each pixel of the feature map (left column). After incubation, groups of four droplets were routed and merged with an $e\text{Drop}(\bar{T}_i)$ (purple solid circle) containing a conversion gate, enabling the computation of a global sum pooling operation on four pixels in each feature map for on-chip pooling (right column). For signal readout, a reporter chiplet $e\text{Drop}(R_{\bar{Y}})$ (yellow solid circle) was simultaneously merged together and the resulting signal was measured after incubated at room temperature for ~ 2 h.

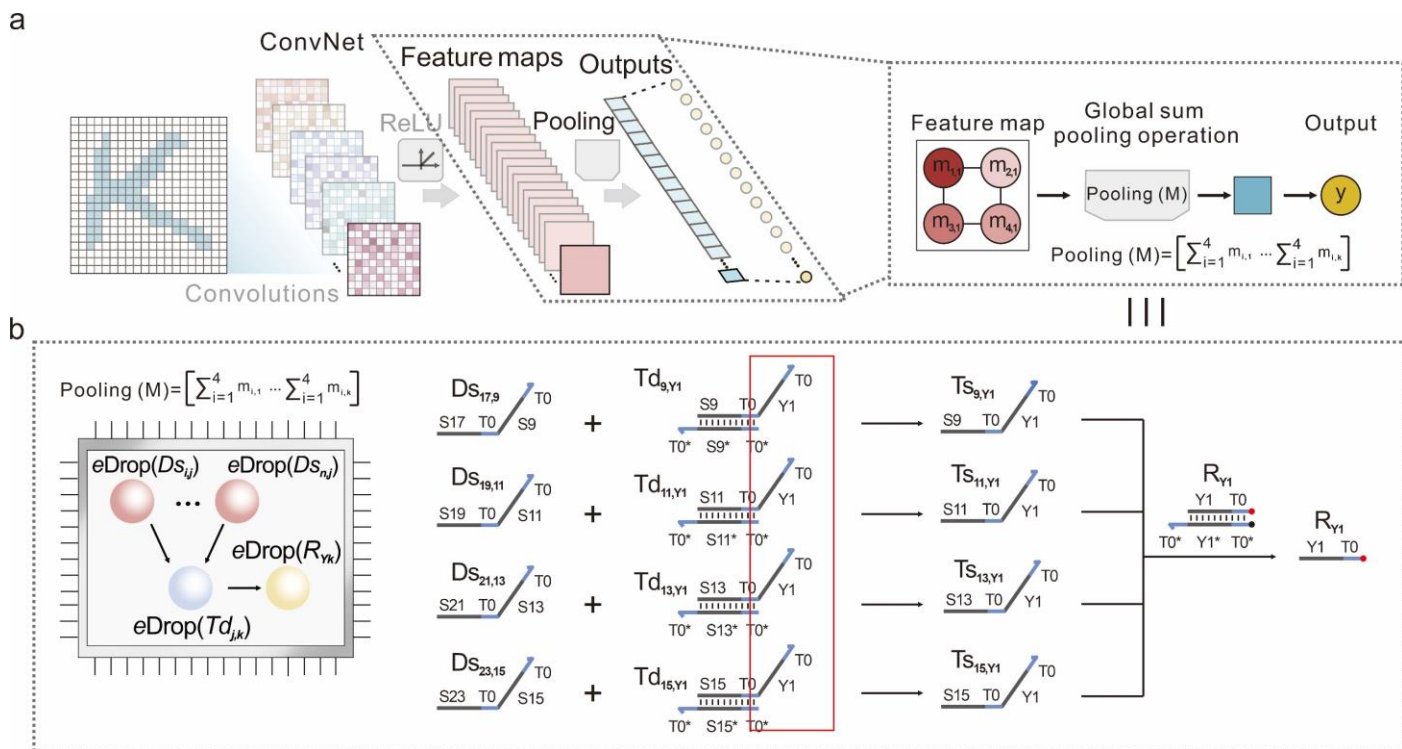


Fig. S29| The on-chip implementation of pooling operation. **a**, The pooling layer transforms each feature map into a single output by summing all pixels of the feature map. Here, we apply global sum pooling to consolidate information across the spatial (height $\times$ width) dimensions, yielding one scalar value per feature map. **b**, To complete the global sum pooling operation, we need to compute the sum of the four pixels in each feature map. These four pixels in a feature map are encoded as concentrations of four distinct nucleic acids sequences ($DS_{i,j}$). The conversion gate ($Td_{j,Y1}$) implements pooling at the molecular level by converting an upstream input strand into a downstream intermediate species that carries a common recognition domain (highlighted by the red box) at its 3' end. Each conversion gate has a unique recognition domain at its 5' end that selectively responds to one upstream input. Note that the conversion gates can be encapsulated as $eDrop(Td_{j,k})$ (shown as purple solid circles) to realize the pooling operation on chip. Coloured lines in DNA strands indicate distinct functional domains.

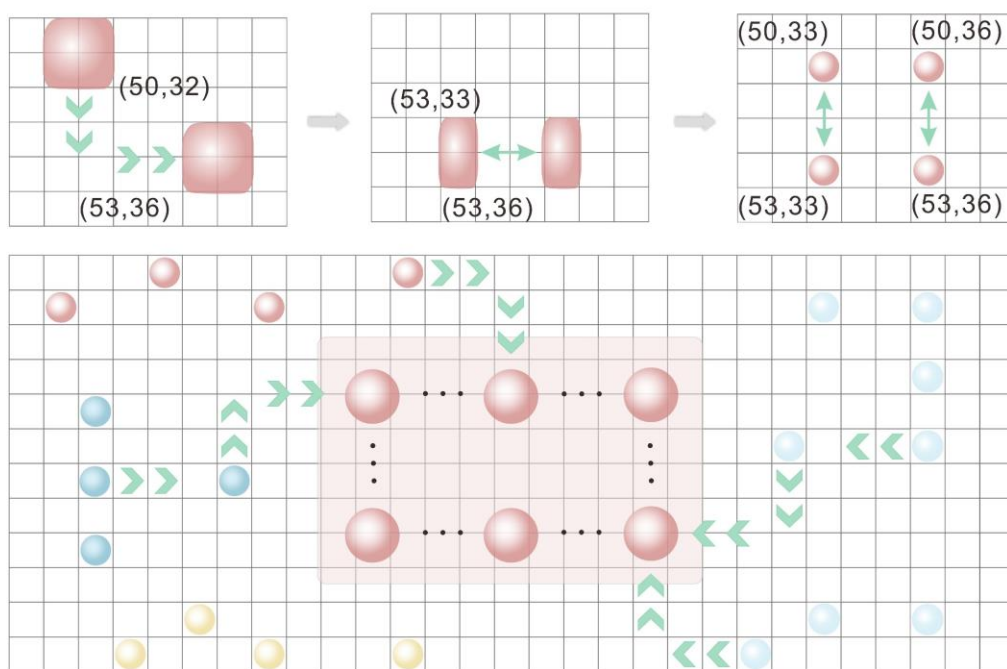
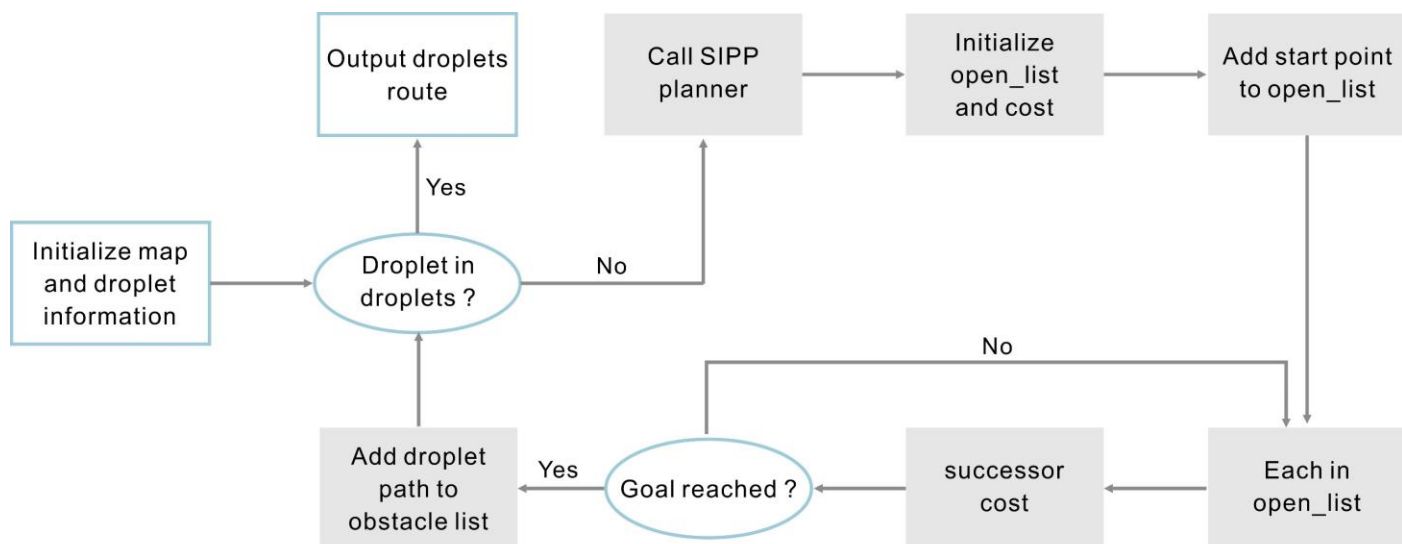


Fig. S30| Program flowchart for the SIPP-based droplet routing algorithm. Our DMIC employs a Safe Interval Path Planning algorithm (SIPP)¹² to generate the path plans for molecular chiplets, ensuring collision-free routing to predetermined positions required for DNA circuit assembly on the chip. To implement this, we enforced a minimum inter-droplet spacing equivalent to two electrodes and modified the SIPP algorithm to treat occupied electrodes as dynamic obstacles. In details, once a candidate path is computed for a droplet, the planner registers the droplet's predicted spatio-temporal occupancy as a dynamic obstacle and derives the corresponding safe intervals for those electrodes; subsequent planning queries must respect these reservations, so multiple droplets are routed without temporal or spatial conflict. Operationally, after the droplet array is generated on-chip, an object-detection model locates and classifies each droplet and maps each classification to its target electrodes. Those targets are fed to the SIPP-based planner, which returns time-stamped, collision-free trajectories and the associated actuation schedule.

S5: Demonstration of Supervised Learning

Machine learning (ML) refers to the use of algorithms that iteratively learn from problem-specific training data. Depending on the task and data availability, ML is typically categorized into supervised learning, unsupervised learning, and reinforcement learning, with supervised learning being the most widely applied *in-silico* training¹³.

Here, we experimentally demonstrate a supervised learning scheme implemented directly on a DMIC, thereby enabling a DNA neural network to be trained. While weight updates in our workflow are computed off-chip using predefined mathematical formulas, all major training steps—including weight instantiation, forward propagation, and network reconfiguration—are executed physically within the chip. We believe that this supervised learning demonstration on DMIC represents a critical conceptual shift from previous molecular computing paradigms, where network parameters were optimized entirely *in silico* and then compiled into a static molecular implementation. In contrast, our DMIC supports iterative, on-chip adaptation of DNA neural networks, analogous to how *in-silico* models are trained, highlighting the key conceptual shift it represents.

Despite these advances, our present implementation faces several practical limitations. First, the supervised training process on a DMIC still requires several days to complete. In our work, the training time of DMIC is primarily determined by two parameters: t , the time required to complete a single forward pass for one sample, and n , the maximum number of input samples that can be processed in parallel on a DMIC. If we use a database of size m to train a DNA neural network, it requires k iterations to reach the desired model. Then the total training time

$$T = \left\lceil \frac{m}{n} \right\rceil \times k \times t$$

which reduces to $T = k \times t$ when $m \leq n$ (all samples processed in one batch).

Reducing the training time can be approached from two directions: (i) shortening the time t per forward propagation step, which could be achieved through further improvements in molecular design; (ii) increasing

the maximum number of parallel input samples n , which would require scaling up the chip size. Second, the current chip size and supported network architectures limit the complexity of DNA neural networks that can be implemented and trained.

Nevertheless, our supervised learning scheme on our DMIC offers an inherent scalability advantage. For large datasets ($m > n$), increasing the chip parallelism n reduces total training time roughly proportionally. Importantly, because the dominant per-iteration cost t is executed concurrently across up to n samples, the per-network latency remains essentially constant as more networks are trained in parallel; the system therefore trades fixed per-network latency for increased throughput. The parallelizability provided by our DMIC enables scaling training throughput without proportionally increasing training time per network, which is a key advantage over serial or manually operated workflows.

Looking ahead, we believe that DMIC holds great potential to evolve into a fully automated training platform for DNA neural networks. A dedicated supporting software could be developed to automate the entire training workflow—from the initial sample input to the final trained outcome—without human intervention. In this manner, once the training task is defined and the execution program is compiled in the control software, DNA neural networks will be autonomously configured in parallel, input patterns will be loaded, and training outputs will be processed. The software will determine whether the training objective has been achieved; if not, it will compute updated weights and initiate the next training cycle automatically. Such integration would allow DNA neural networks to be autonomously configured, trained in parallel, and iteratively refined until target performance is reached.

Our current work is therefore a proof of concept: a physically implemented, hybrid bioelectronic computing system that makes supervised training of DNA neural networks possible, while laying the conceptual and technical foundation for future automated, large-scale *in-molecule* learning platforms. We believe that combining molecular and electronic computation will enable capabilities neither modality can

950 achieve alone, and our DMIC exemplifies this complementarity.

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

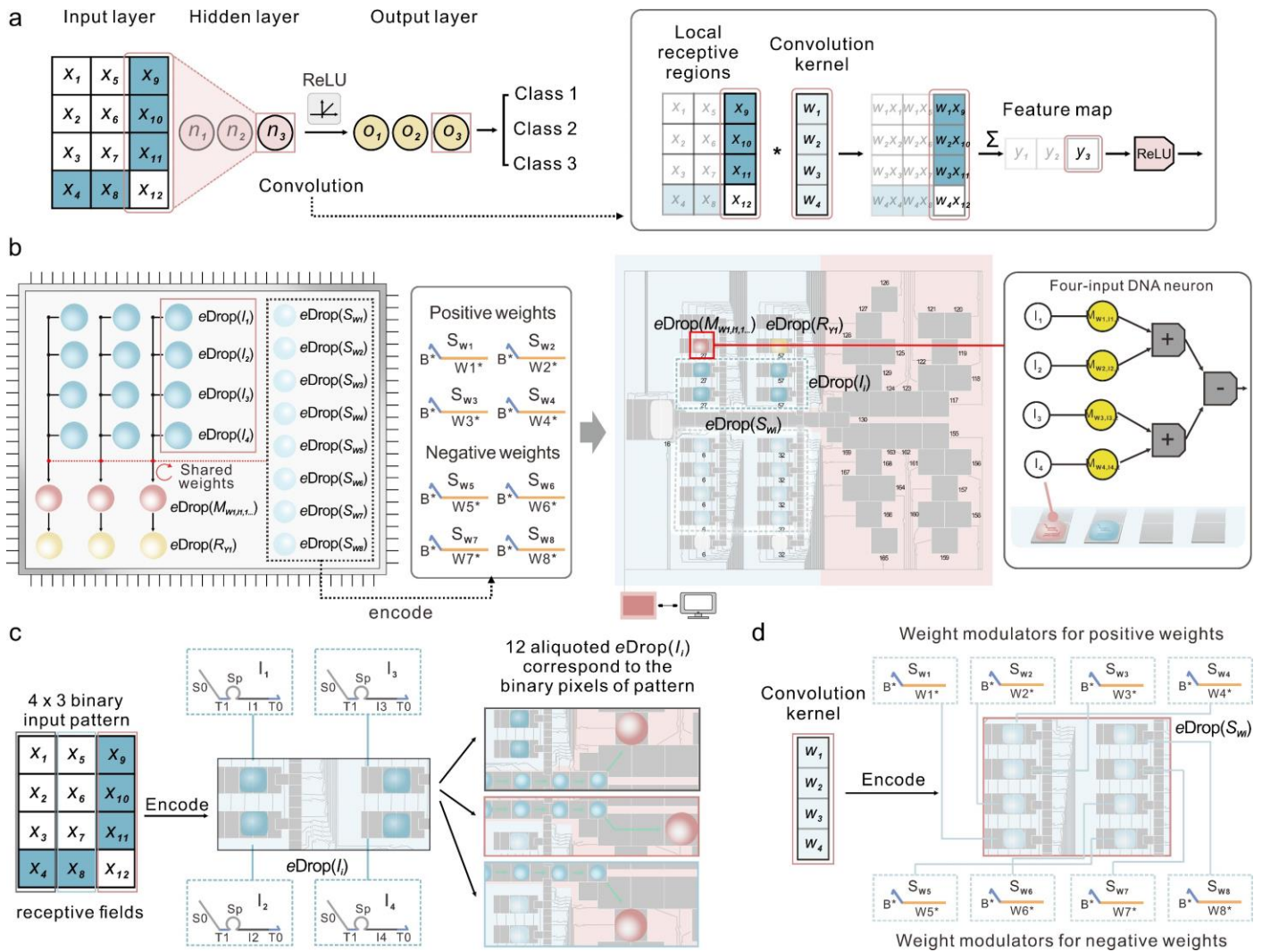


Fig. S31| On-chip DNA-based ConvNet used to demonstrate supervised learning. **a**, A trained convolution neural network model and a typical case of convolution operation with a 4×1 -sized image kernel (stride=1). The pink solid box in the image indicates the individual receptive region of this sample case. **b**, The diagram illustrating the on-chip implementation of DNA-based ConvNet. Left: abstract diagram of the ConvNet demonstration on chip. Right: a DMIC monolithically integrating DNA neurons to implement on-chip DNA-based ConvNet computation. Three reaction buffer droplets (white colored droplet) are employed to maintain consistent DNA molecule concentration throughout the reaction process. The expanded view of the four-input DNA neuron. **c**, The pixel in each local receptive region can be encoded with four input chiplets $eDrop(I_i)$, and the pixels interacting with the same kernel in different local receptive regions can be implemented with the same chiplet (for example, x_1 , x_5 and x_9 is programmed with $eDrop(I_1)$, shown in pink dashed box). During operation, this on-chip DNA-based ConvNet was divided into three subnetworks, each of which is responsible for processing inputs from individual local receptive regions, thereby enabling parallel convolutional computations within three separate reaction chambers. Each DNA neuron reads four inputs (shown in right column of **b**) from local receptive regions works in parallel within different reaction chambers. **d**, Each pixel w_i in the kernel is encoded using a pair of weight modulator chiplets $eDrop(S_{wi})$ representing positive and negative weights separately. As the network runs, only four weight modulator chiplets are utilized to determine each neuron connection strength, allowing for the implementation of the parallel neuron computation with shared weights in distinct reaction chambers.

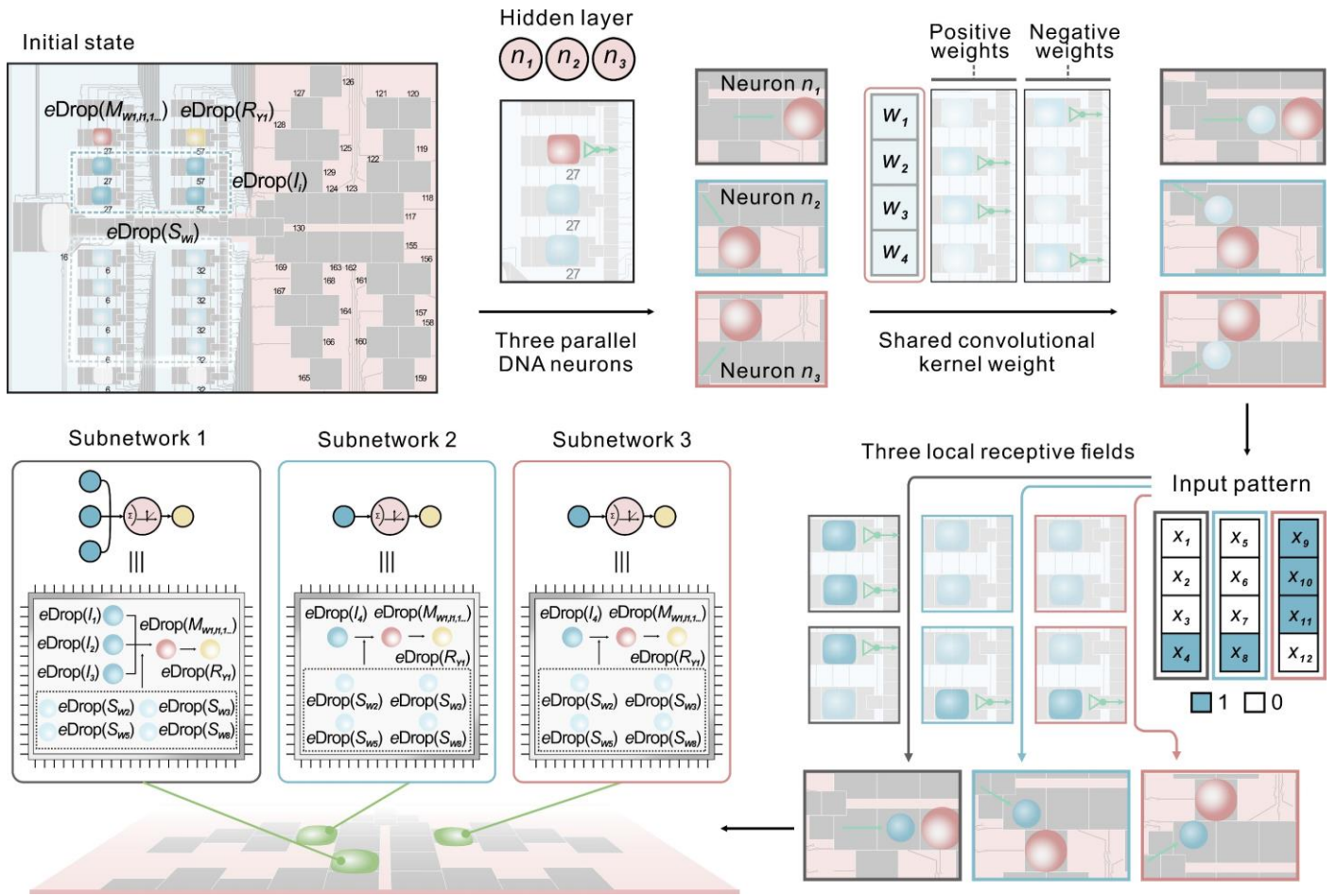


Fig. S32| DMIC enables the DNA-based ConvNet computation. The configuration of a DNA-based ConvNet on a DMIC. During operation, this network can be divided into three groups, enabling independent and parallel execution of each compartmentalized DNA subnetwork within each reaction chamber, allowing the integration of isolated neuron computations to implement DNA-based ConvNet computation on chip. For the on-chip implementation of three hidden neurons, the same DNA neuron can be placed at different locations to function as distinct neurons. Each hidden neuron corresponds to the local receptive region of an element in the feature map accordingly. The shared weight value of each pixel in the kernel can be precisely set and individually reset by weight modulator chiplets $eDrop(S_{wi})$. Binary input patterns with '1' (blue square) and '0' (white square), corresponding to the presence and absence of input chiplet $eDrop(I_i)$, respectively.

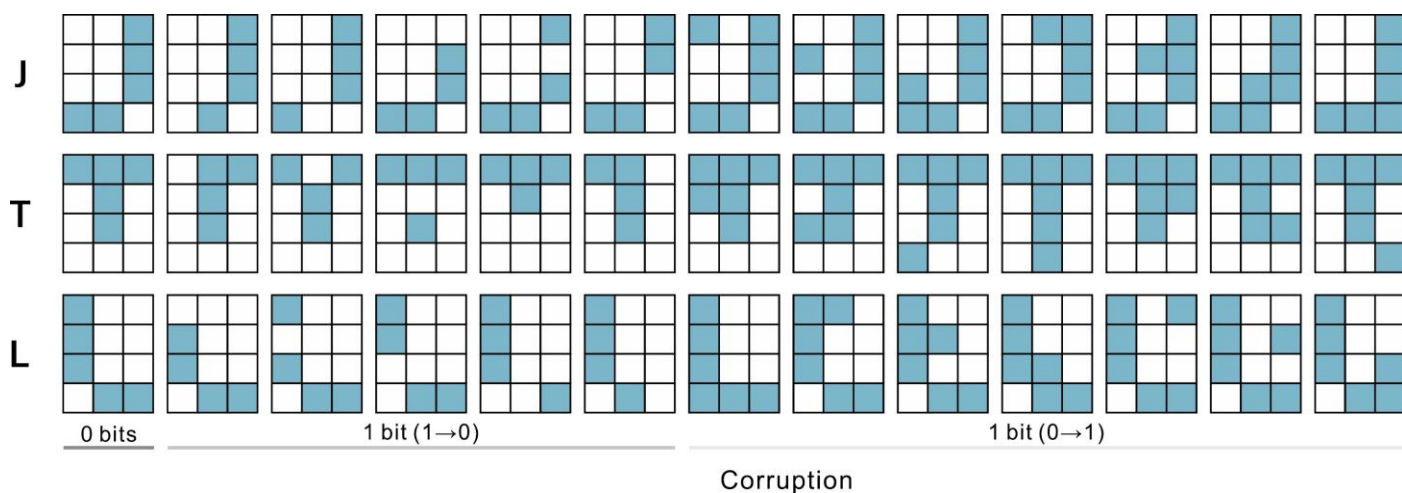


Fig. S33| The training dataset used in this work. Each 12-bit binary input patterns with '1' (blue square) and '0' (white square).

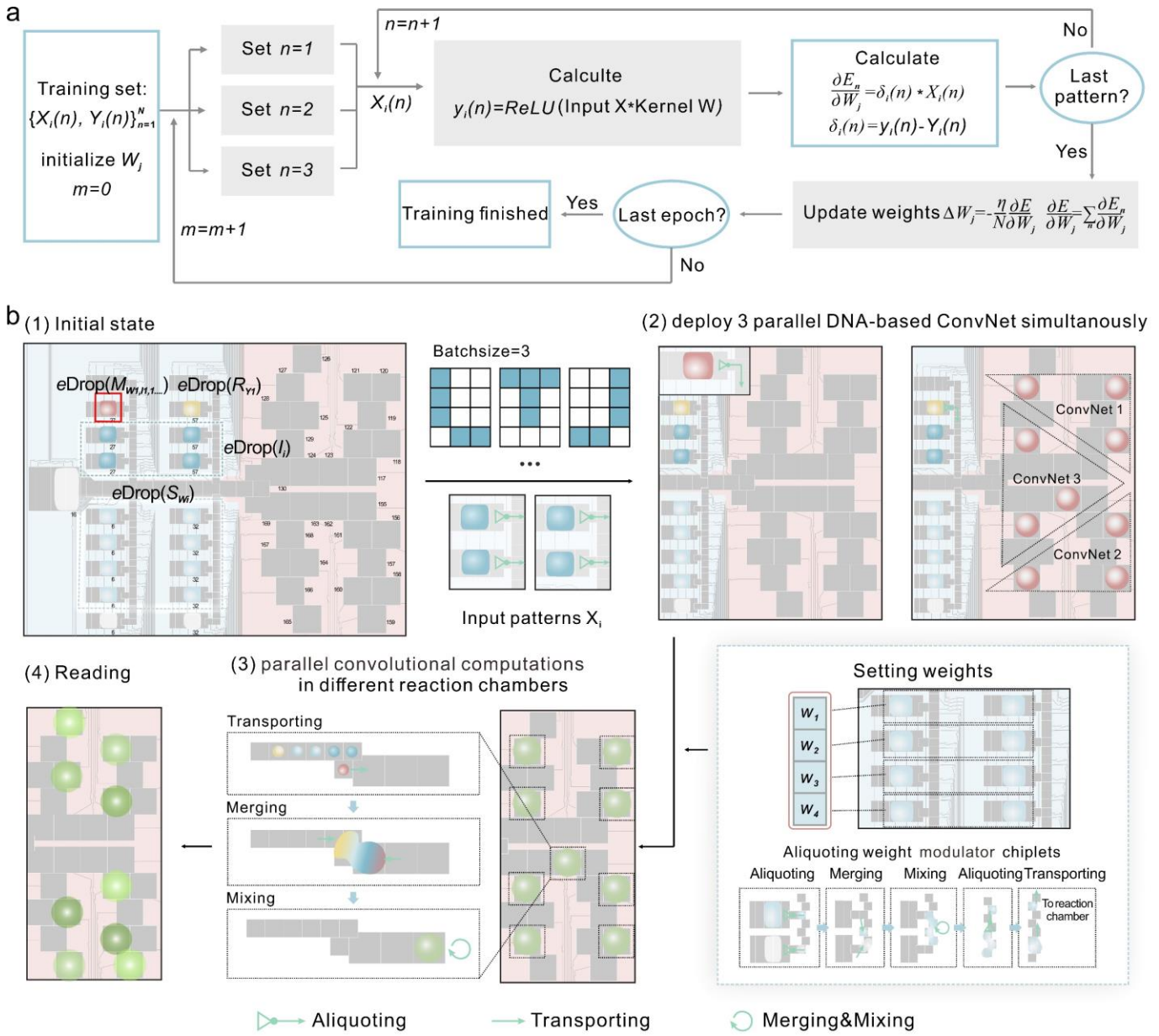


Fig. S34| Workflow of supervised learning using DMIC. **a**, The flow chart illustrating one epoch of the training algorithm used in this work. The steps implemented on DMIC are shown in grey-shaded boxes. Note that each output signal is associated with a defined class. Here the dataset size is $N=39$. n is the pattern number ($1 \leq n \leq 39$). $X_i(n)$ and $Y_i(n)$ are the i th input and output for n th input pattern. W_j is the trainable kernel. η is a parameter that scales the learning rate. The algorithm is described in detail in supervised learning details section. **b**, The detailed workflow of our DMIC for supervised learning. At each training epoch, our DMIC enables to configure three DNA-based ConvNets with identical architecture in a single chip, each of which can be divided into three subnetworks. Three input patterns from the training dataset can be applied to the network's input simultaneously, enabling parallel training of the DNA-based ConvNet. Note that nine subnetworks run independently in isolated reaction chambers to compute corresponding outputs ($y_i(n)$). The weight value of each pixel in the kernel can be precisely set and individually reset by weight modulator chiplets $e\text{Drop}(S_{W_i})$ (dashed blue box). Once three on-chip DNA circuits are completed, nine aggregated droplets will be removed, and DMIC will begin to reconfigure three entirely new networks for simultaneous processing of the other three input patterns. The process depicted here are repeated 13 times in one epoch to complete the processing of all 39 input patterns. The volume of droplets located in the memory is $2.50 \mu\text{L}$ (DNA species I_i concentration: $50\times$, DNA species $M_{W_i, I, j}$ concentration: $25\times$, DNA species $S_{d, j, k}$, D_{d, k, Y_i} and R_{Y_i} concentration: $100\times$, DNA species Sub_{j, Y_i} concentration: $150\times$, DNA species S_{W_i} concentration: $5\times$). The volume of droplets aliquoted from the memory is $0.32 \mu\text{L}$, while the volume of aliquoted reaction buffer droplet is $1.60 \mu\text{L}$. The standard concentration is 100 nM ($1\times$).

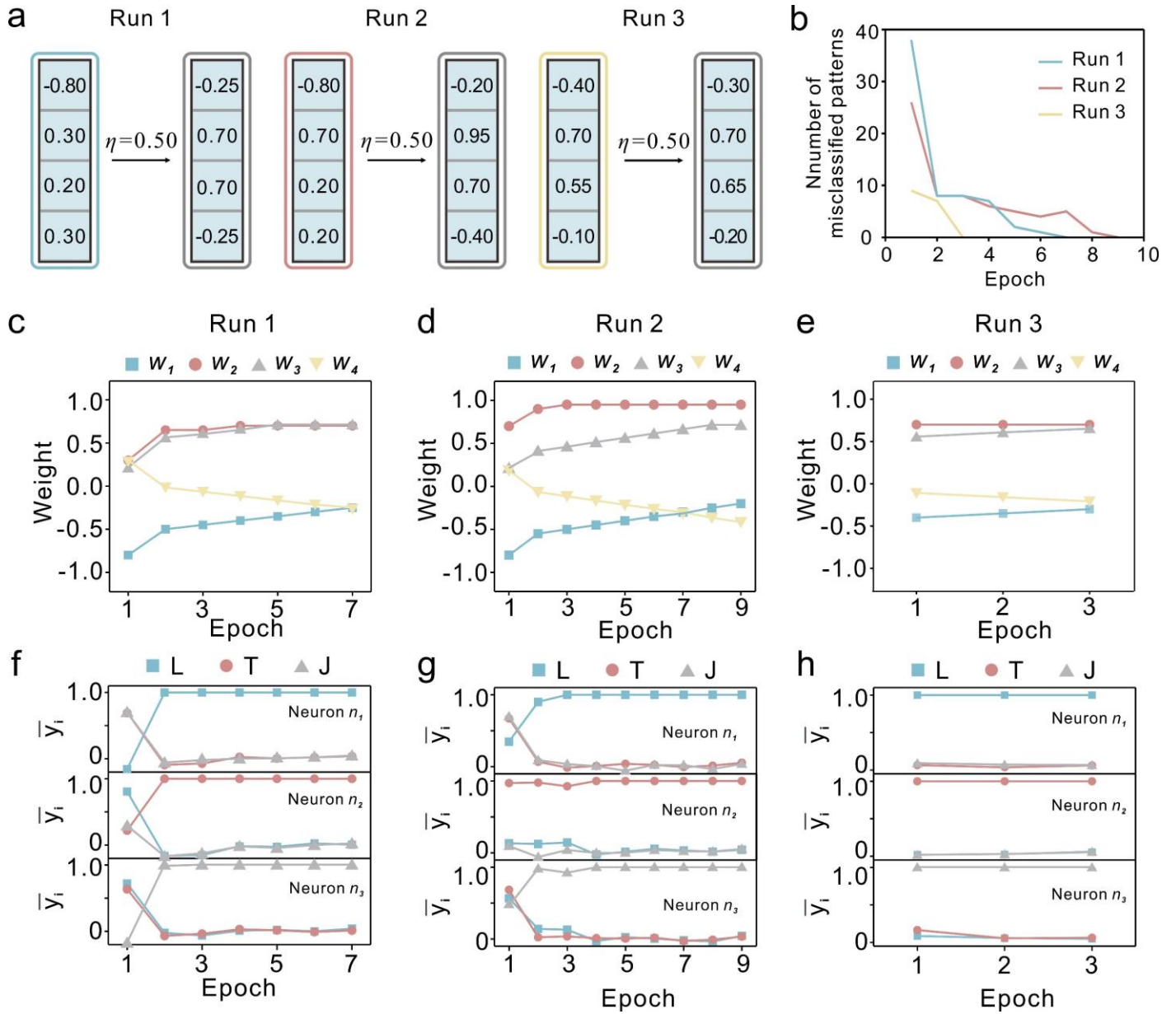


Fig. S35| Changes in the kernel and network outputs over time. **a**, The weight update with different initial weight settings. Left: initial weight values in three separate training experiments (“Runs”). Right: final weight values in three separate runs, after the perfect classification have been reached (after 7, 9, and 3 training epochs for Runs 1, 2, and 3, respectively). The learning rate is 0.50. **b**, Convergence of network output in three separate runs. **c,d,e**, Changes of four weights in kernel over time when all training patterns are repeatedly shown, for three separate runs from different initial weight settings. **f,g,h**, The evolution of output signals, during the training process, averaged over all trained patterns of a specific class.

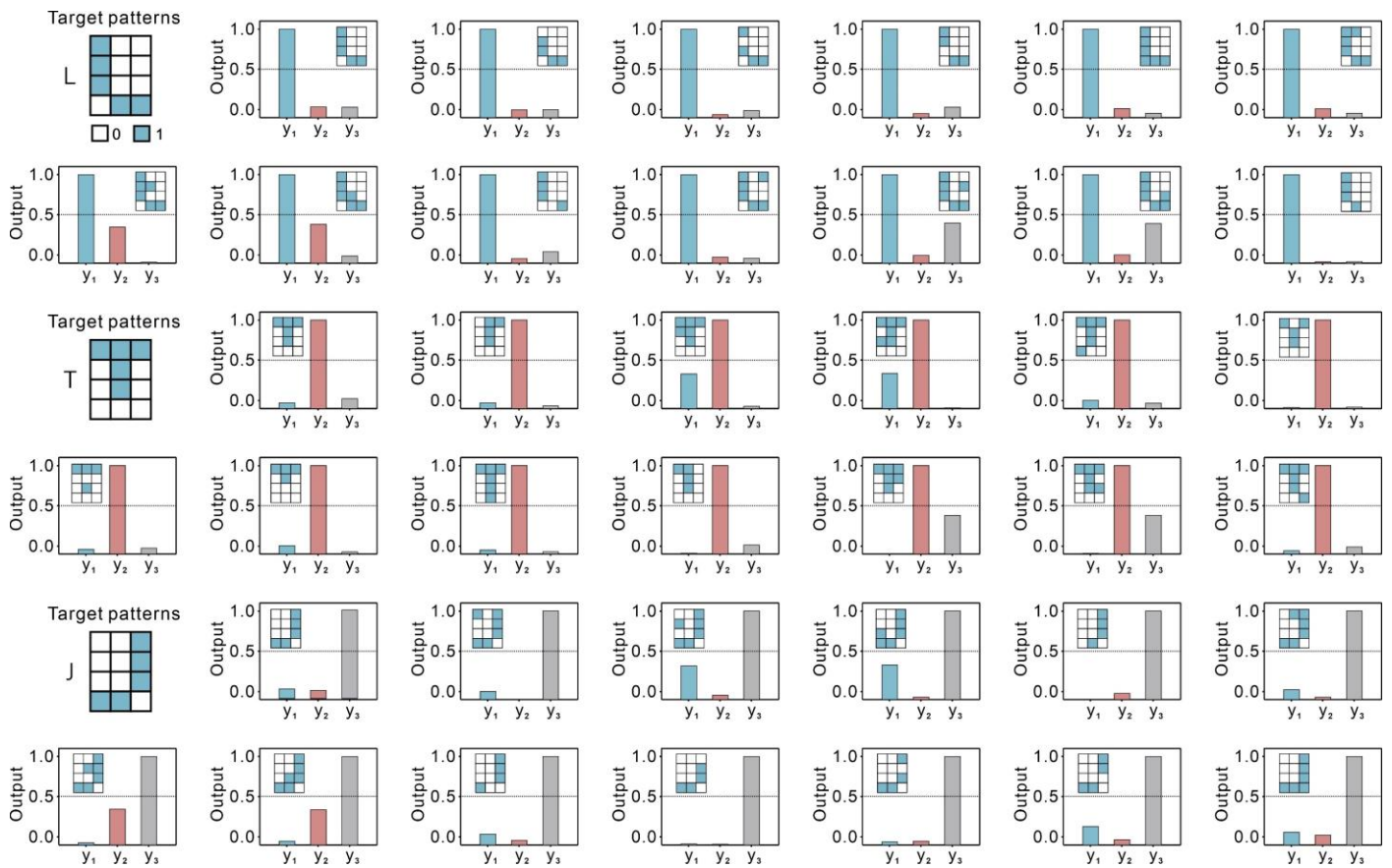


Fig. S36| Pattern classification results. Fluorescence level of outputs of 39 input patterns at 36 h after 7 training epochs, collected from Run 1. Each colored histogram corresponds to a learned pattern from the labelled training dataset. Each output signal only (turn 'ON') to one of the learned patterns, showing the perfect classification performance was achieved.

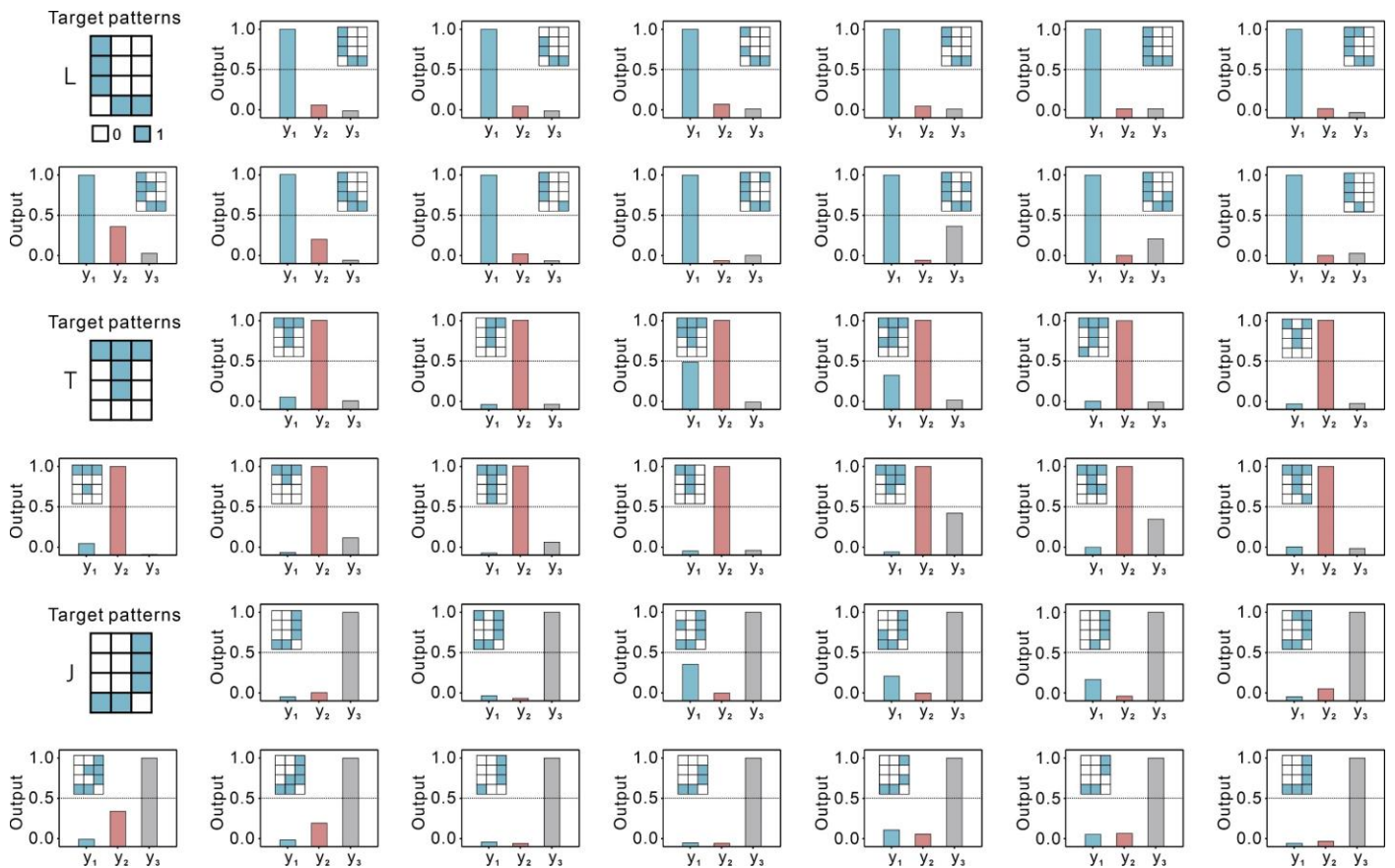


Fig. S37| Pattern classification results. Fluorescence level of outputs of 39 input patterns at 36 h after 9 training epochs, collected from Run 2. Each colored histogram corresponds to a learned pattern from the labelled training dataset. Each output signal only (turn 'ON') to one of the learned patterns, showing the perfect classification performance was achieved.

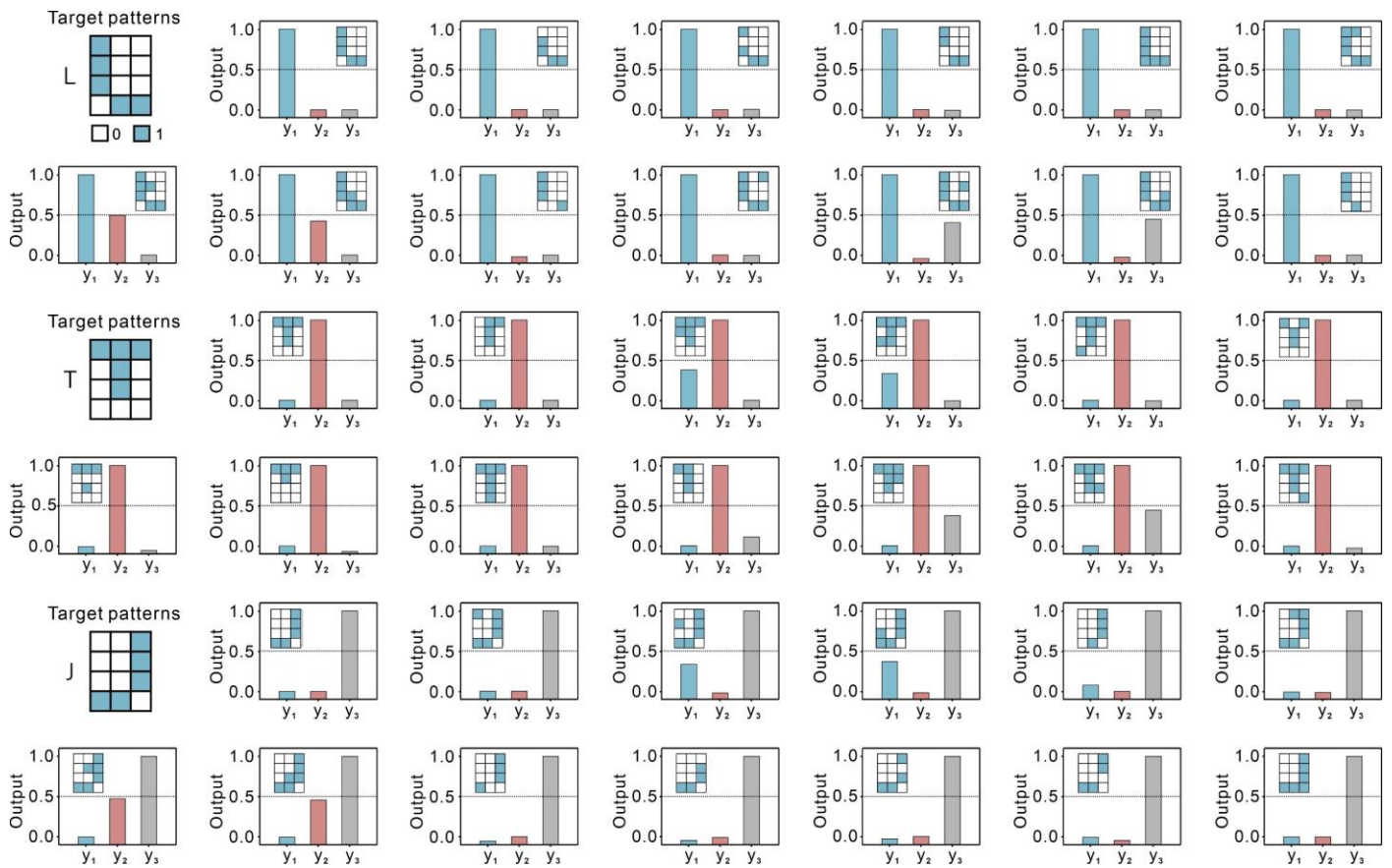


Fig. S38| Pattern classification results. Fluorescence level of outputs of 39 input patterns at 36 h after 3 training epochs, collected from Run 3. Each colored histogram corresponds to a learned pattern from the labelled training dataset. Each output signal only (turn 'ON') to one of the learned patterns, showing the perfect classification performance was achieved.

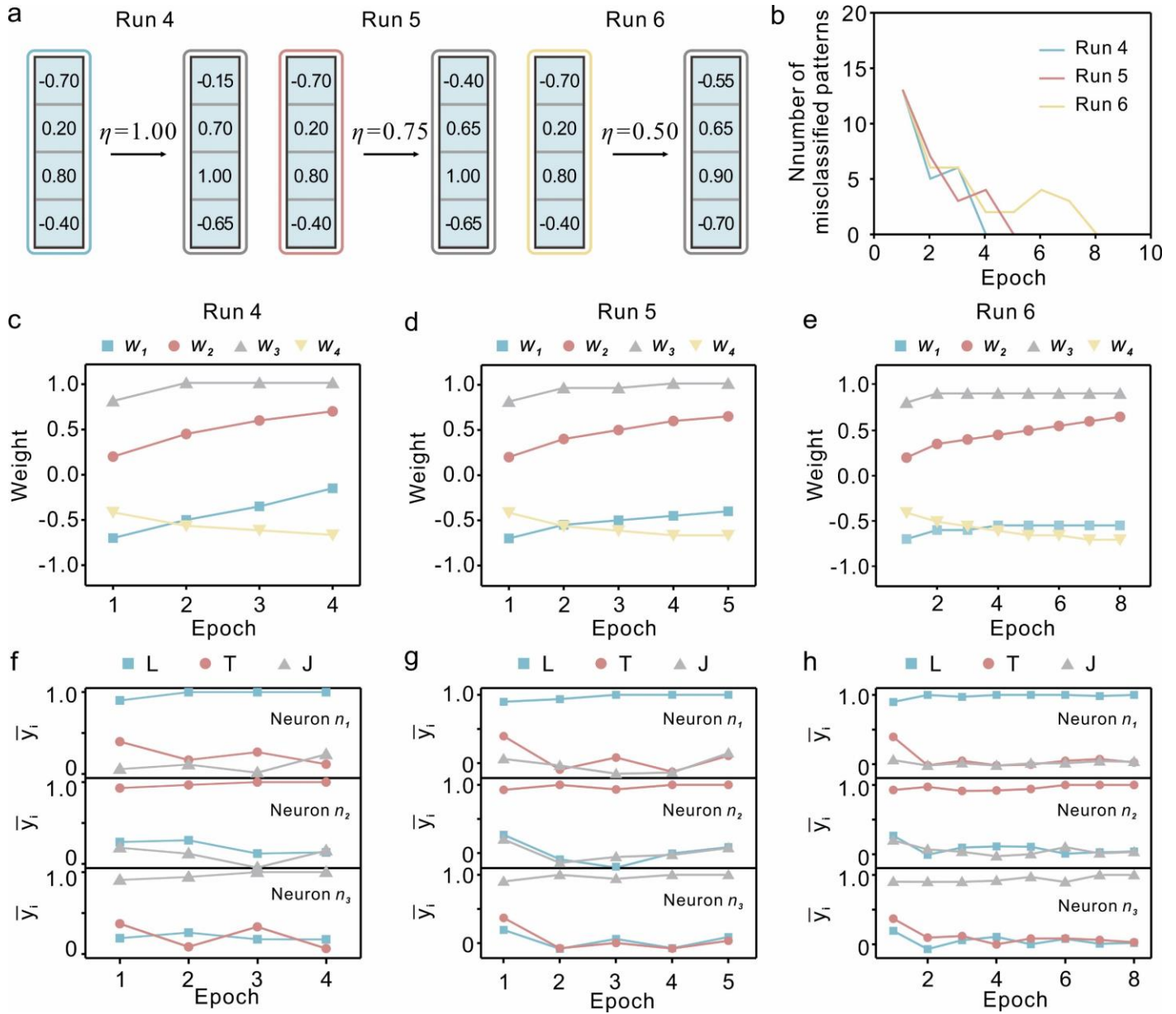


Fig. S39| Changes in the kernel and network outputs over time. a, The weight update with different learning rate settings, but with the same initial weight values. Left: the same initial weight values in three separate training experiments (“Runs”). Right: final weight values in three separate runs, after the perfect classification have been reached (after 4, 5, and 8 training epochs for Runs 4, 5, and 6, respectively). **b**, Convergence of network output in three separate runs. **c,d,e**, Changes of four weights in kernel over time when all training patterns are repeatedly shown, for three separate runs from different learning rate settings. **f,g,h**, The evolution of output signals, during the training process, averaged over all trained patterns of a specific class.

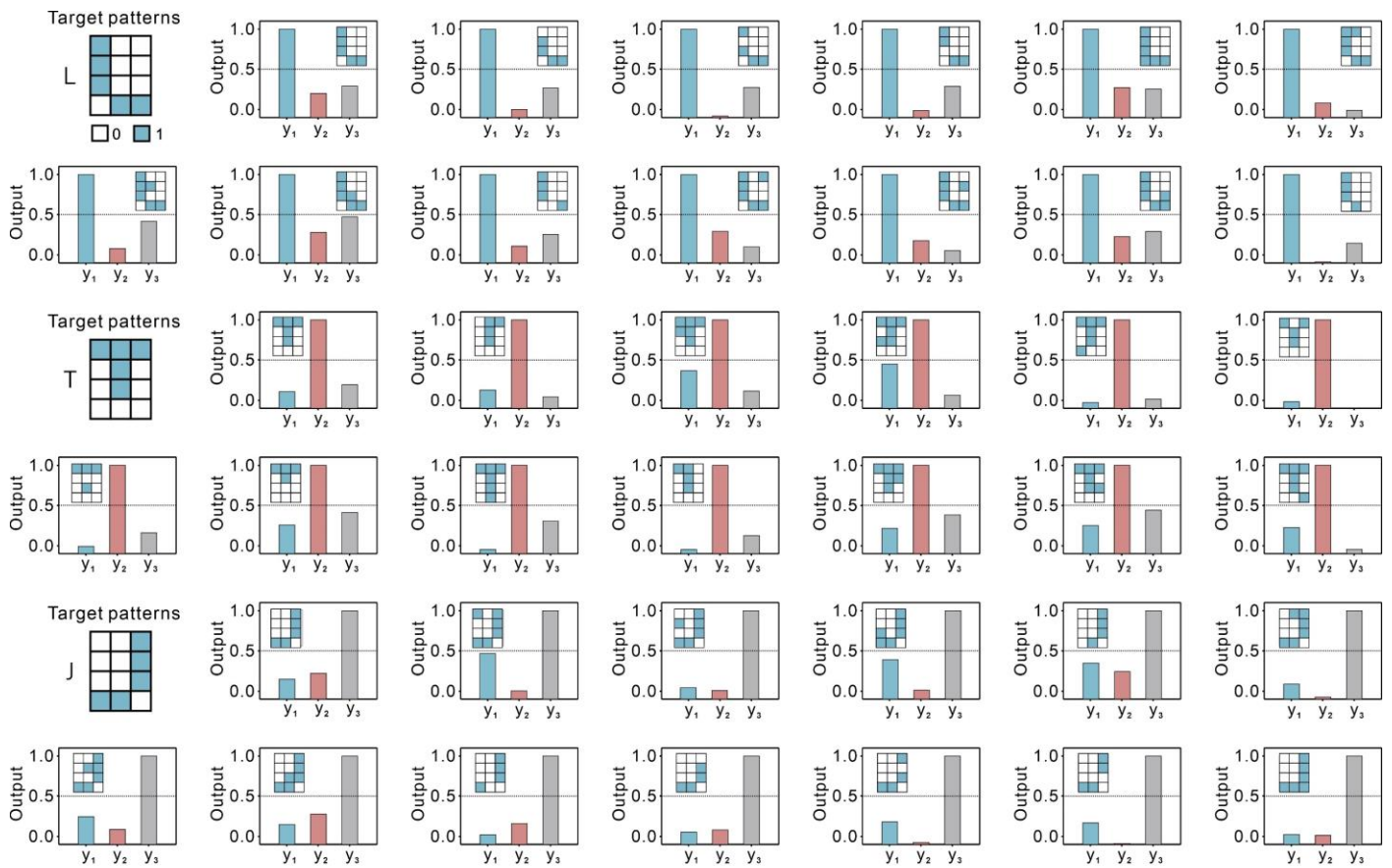


Fig. S40| Pattern classification results. Fluorescence level of outputs of 39 input patterns at 36 h after 4 training epochs, collected from Run 4. Each colored histogram corresponds to a learned pattern from the labelled training dataset. Each output signal only (turn 'ON') to one of the learned patterns, showing the perfect classification performance was achieved.

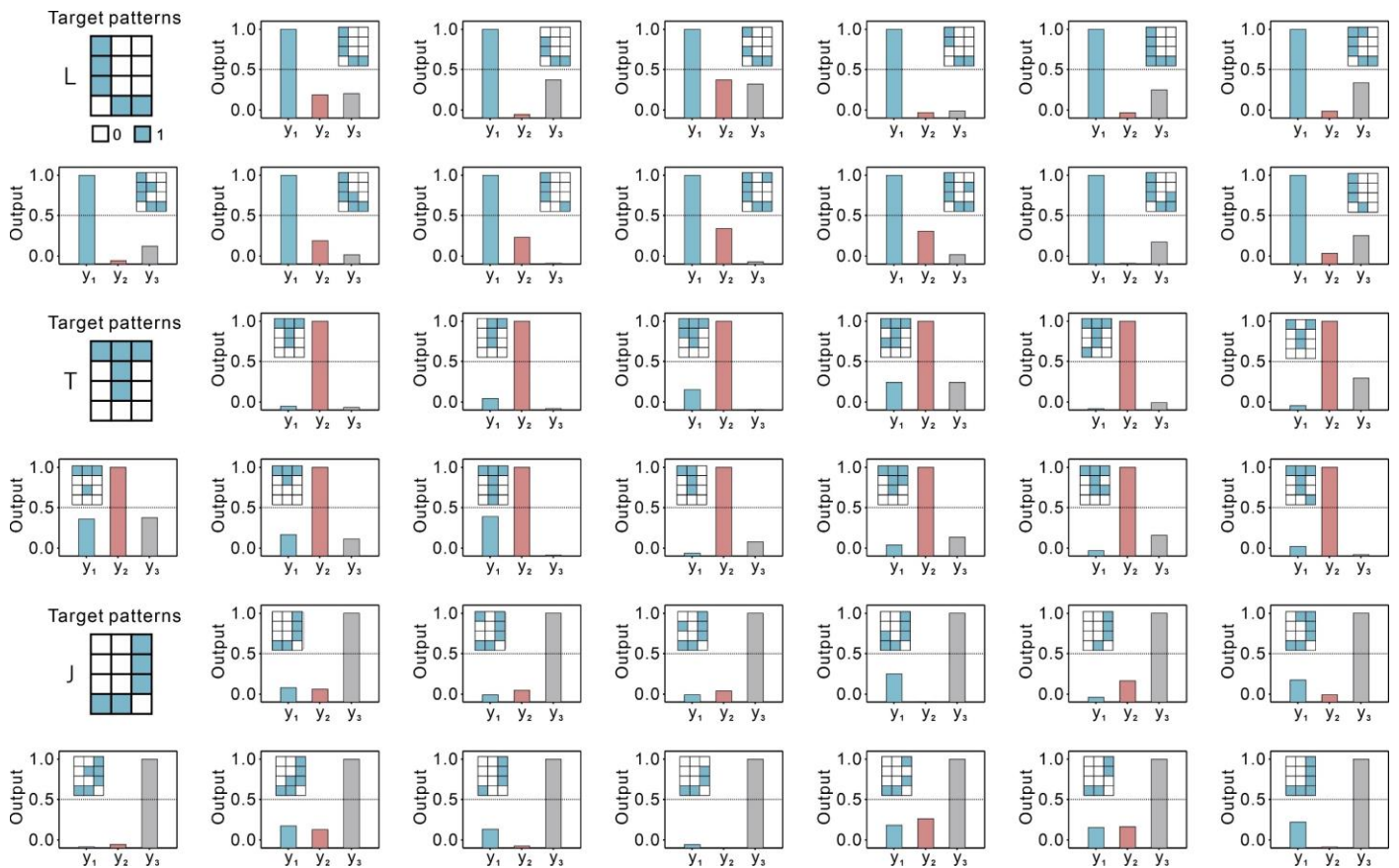


Fig. S41| Pattern classification results. Fluorescence level of outputs of 39 input patterns at 36 h after 5 training epochs, collected from Run 5. Each colored histogram corresponds to a learned pattern from the labelled training dataset. Each output signal only (turn 'ON') to one of the learned patterns, showing the perfect classification performance was achieved.

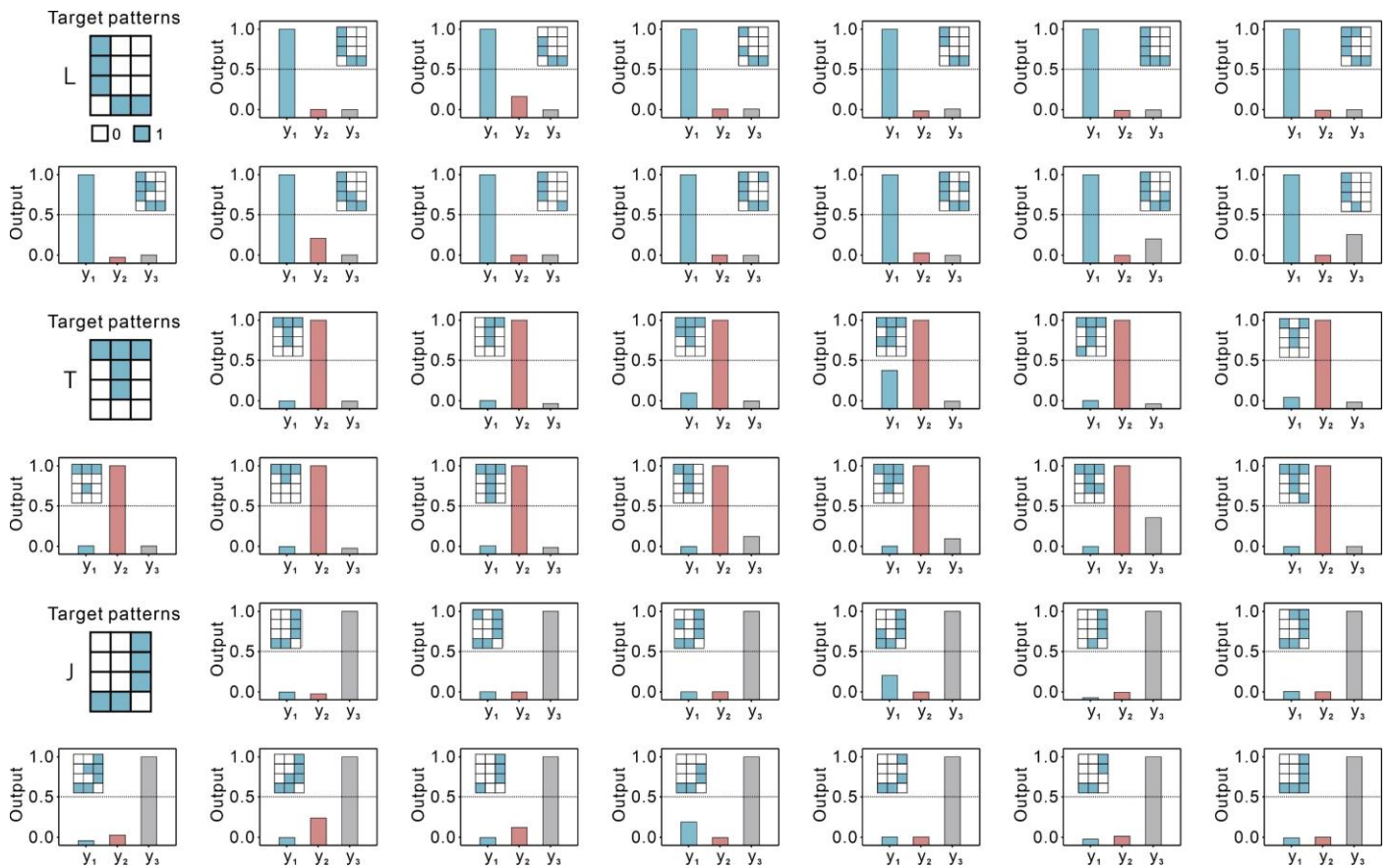


Fig. S42| Pattern classification results. Fluorescence level of outputs of 39 input patterns at 36 h after 8 training epochs, collected from Run 6. Each colored histogram corresponds to a learned pattern from the labelled training dataset. Each output signal only (turn 'ON') to one of the learned patterns, showing the perfect classification performance was achieved.

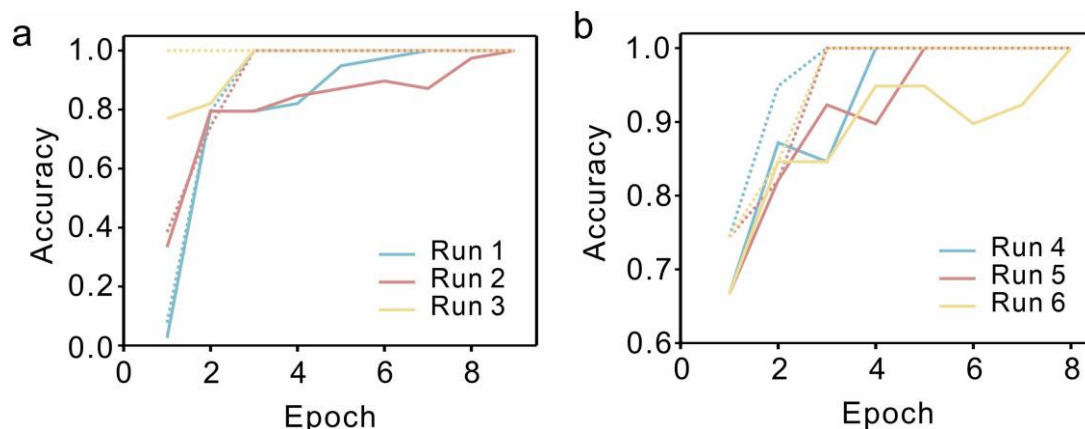
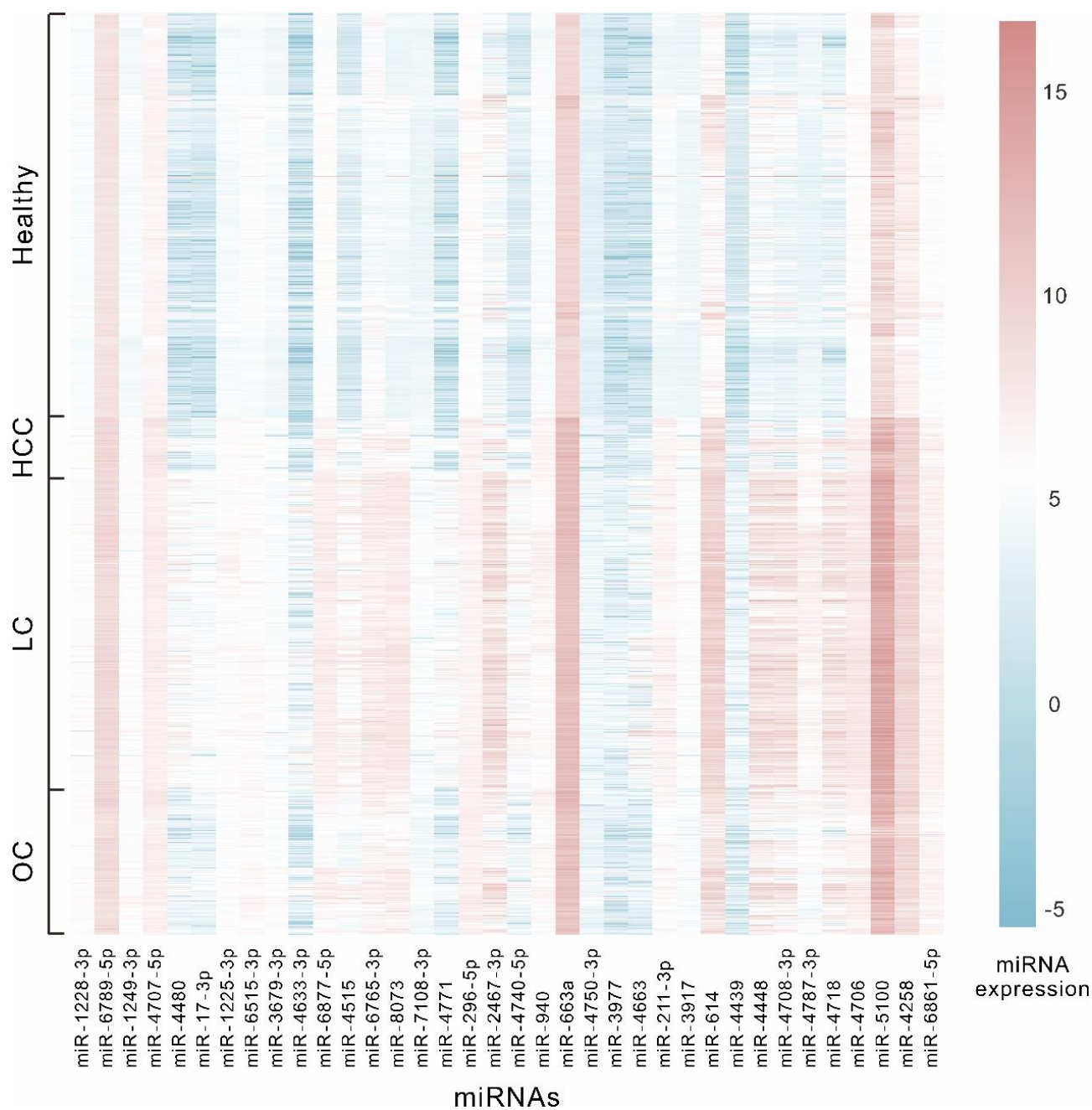


Fig. S43| Comparison of the accuracy *in-silico* simulation with the same architecture and learning scheme. a, Classifier training of the DNA-based ConvNet on a DMIC (solid lines) and simulation of the ConvNet *in-silico* (dashed lines) for different initial weight value settings. **b,** Classifier training of the DNA-based ConvNet on a DMIC (solid lines) and simulation of the ConvNet on a computer (dashed lines) for different learning rate settings. The same ConvNet architecture, training dataset, learning rate and initial weight values were used. After training, the same accuracy was eventually reached. The slower convergence of the on-chip learning compared with the computer simulation, indicating the nonidealities (finite precision of the analog variables, the uncertainty and errors inherent to molecular computations) of DMIC.

S6: Demonstration of Clinical Diagnostic Application



191

192

193

194

195

196

197

Fig. S44| The heatmap of the miRNAs expression data from The National Center for Biotechnology Information (NCBI). The publicly available miRNA expression data corresponding to 1009 healthy individuals and 1326 cancer patients (808 LC, 158 HCC and 360 OC) from NCBI, were used for differential expression analysis and *in silico* training for classifier construction.

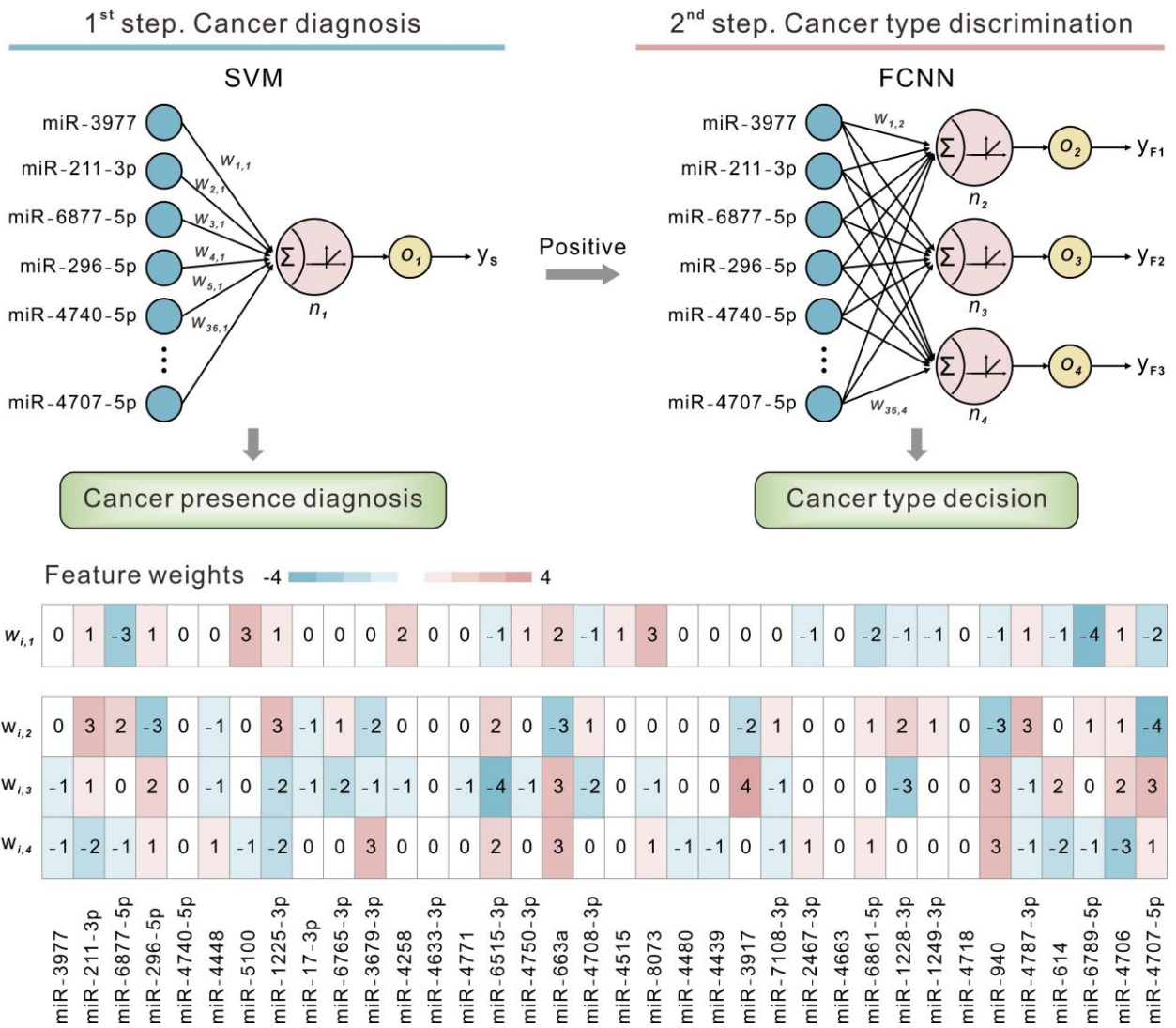


Fig. S45| The resulting models. Top: the architecture of the trained classifier model (an SVM-based binary classification model to detect the presence of cancer, and an FCNN-based multiple classification model to identify the specific cancer type). The classifier uses 36 miRNAs as input neurons x_i , with associated positive and negative weights $w_{i,j}$. The subscripts i ($1 \leq i \leq 36$) and j ($1 \leq j \leq 4$) indicate the connection strength between the input neuron x_i and the hidden neuron n_j . Bottom: the selected 36 miRNAs and their associated positive and negative weights $w_{i,j}$.

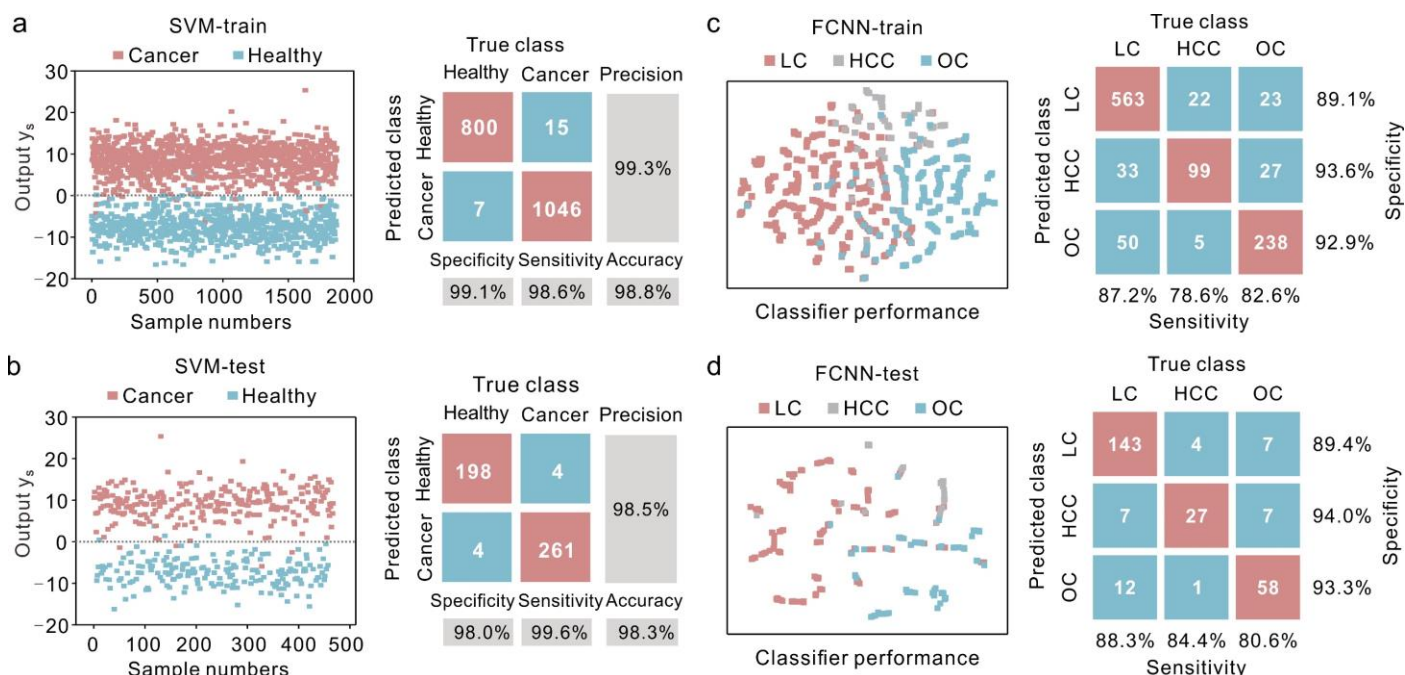


Fig. S46| In silico training of multi-classifiers for diagnosis of multiple cancer types. **a**, The performance of the *in silico*-trained SVM-based classifier in the training dataset, where a sensitivity of 98.6% (95% CI: 97.6%, 99.2%), specificity of 99.1% (95% CI: 98.1%, 99.6%), and accuracy of 98.8% (95% CI: 98.2%, 99.3%) were achieved for cancer presence. **b**, The performance of the *in silico*-trained SVM-based classifier in the test dataset, where a sensitivity of 98.5% (95% CI: 95.9%, 99.5%), specificity of 98.0% (95% CI: 94.7%, 99.4%), and accuracy of 98.3% (95% CI: 96.7%, 99.3%) was achieved for cancer presence. **c**, The performance of the *in silico*-trained FCNN-based classifier in the training dataset, in which LC patients were detectable with a sensitivity of 87.2% (95% CI: 84.3%, 89.6%) and a specificity of 89.1% (95% CI: 85.6%, 91.9%), HCC patients were detectable with a sensitivity of 78.6% (95% CI: 70.2%, 85.2%) and a specificity of 93.6% (95% CI: 91.8%, 95.0%), and OC patients were detectable with a sensitivity of 82.6% (95% CI: 77.7%, 86.7%) and a specificity of 92.9% (95% CI: 90.8%, 94.5%). The total accuracy was 84.9%. **d**, The performance of the *in silico*-trained FCNN-based classifier in the test dataset, in which LC patients were detectable with a sensitivity of 88.3% (95% CI: 82.1%, 92.6%) and a specificity of 89.4% (95% CI: 81.5%, 94.3%), HCC patients were detectable with a sensitivity of 84.4% (95% CI: 66.5%, 94.1%) and a specificity of 94.0% (95% CI: 90.0%, 96.6%), and OC patients were detectable with a sensitivity of 80.6% (95% CI: 69.2%, 88.6%) and a specificity of 93.3% (95% CI: 88.6%, 96.2%). The total accuracy was 85.7%.

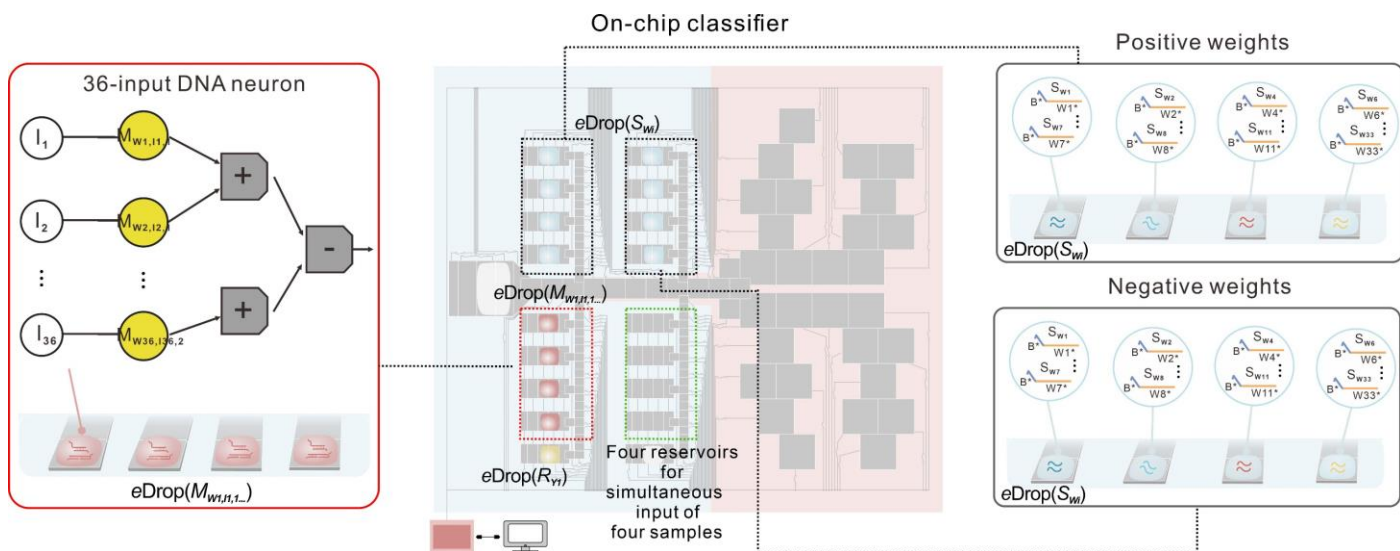


Fig. S47| The on-chip implementation of the molecular multi-classifier. For the on-chip implementation of the resulting classifier model, four hidden neurons n_i that respond to the same 36 inputs are encoded as four molecular chiplets $eDrop(M_{W1,1,1}, \dots)$ (dashed red box) that can be integrated onto a DMIC. The engineered DNA neurons encapsulated in the droplets can react with the same 36 inputs but response to different weight modulators, allowing each input to be assigned a numerical weight capturing its importance for each neuron. During operation, DNA neurons of the on-chip classifier can be divided into four groups, each of which operates simultaneously in separate reaction chambers, enabling the integration of multiple DNA neuron computation processes to derive diagnostic predictions. Eight weight modulator chiplets $eDrop(S_{W1})$ (dashed black box) are designed, each two of which contain the positive and negative weight modulators respectively, enabling the implementation of an independent weight assignments for each neuron. One reporter chiplet $eDrop(R_{W1})$ is designed for the signal readout on-chip. Four reservoirs (dashed green box) on the chip are remained, allowing ssDNA amplified and transformed from miRNAs to be encapsulated into the droplets and input to the on-chip classifier.

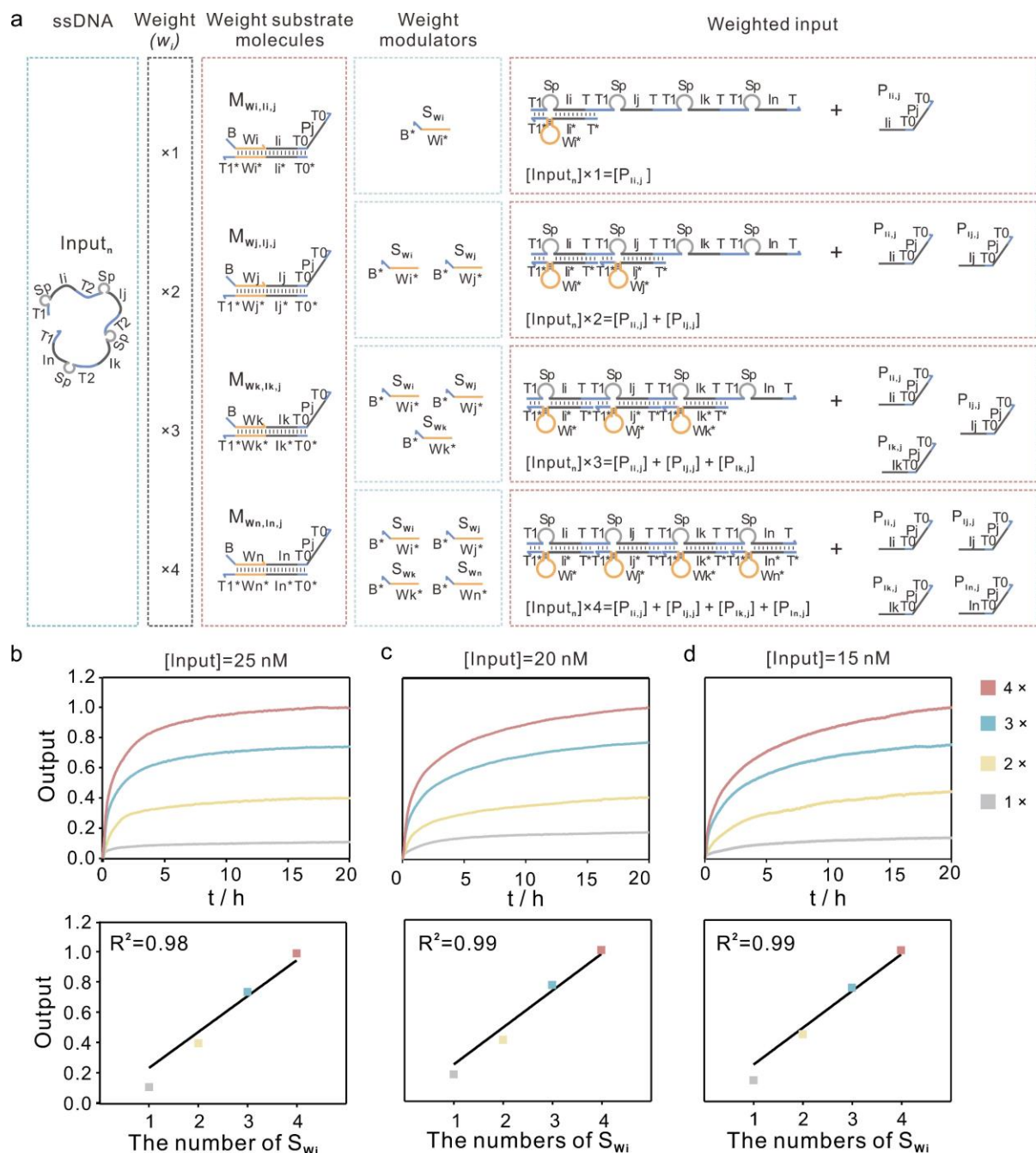


Fig. S48| The DNA implementation of classifier weights. **a**, Scheme for the weight multiplication ($c(\text{input}_n) \times w_i$), where w_i is the integer weight and $c(\text{input}_n)$ is the input concentration. Each ssDNA (input_n) has multiple orthogonal recognition domains, which can convert multiple activated weight substrate molecules $M_{w_i, l_i, j}^*$ to multiple intermediate products $P_{l_i, j}$ with the same recognition domain P_j . These activated weight substrate molecules $M_{w_i, l_i, j}^*$ are implemented by hybridization with a certain number of weight modulators S_{w_i} , allowing each recognition domain of ssDNA to generate a proportional exchange of the activity of the DNA signals. The final concentration of $P_{l_i, j}$ is determined by the number of weight modulator S_{w_i} , thus setting the integer value of the weighted multiplication. Weights $w_i=4, 3, 2$ or 1 are implemented by having $4, 3, 2$ or 1 weight modulator S_{w_i} . **b, c, d**, Experimental validation of the weight multiplication. The fluorescence kinetics (top) and the fluorescence level (bottom) of different concentrations of ssDNA corresponding to $1, 2, 3$ or 4 weight modulators S_{w_i} activating the weight substrate molecules $M_{w_i, l_i, j}$. As expected, we observed a linear relationship between the number of weight modulators and the measured fluorescence response across a range of ssDNA concentrations.

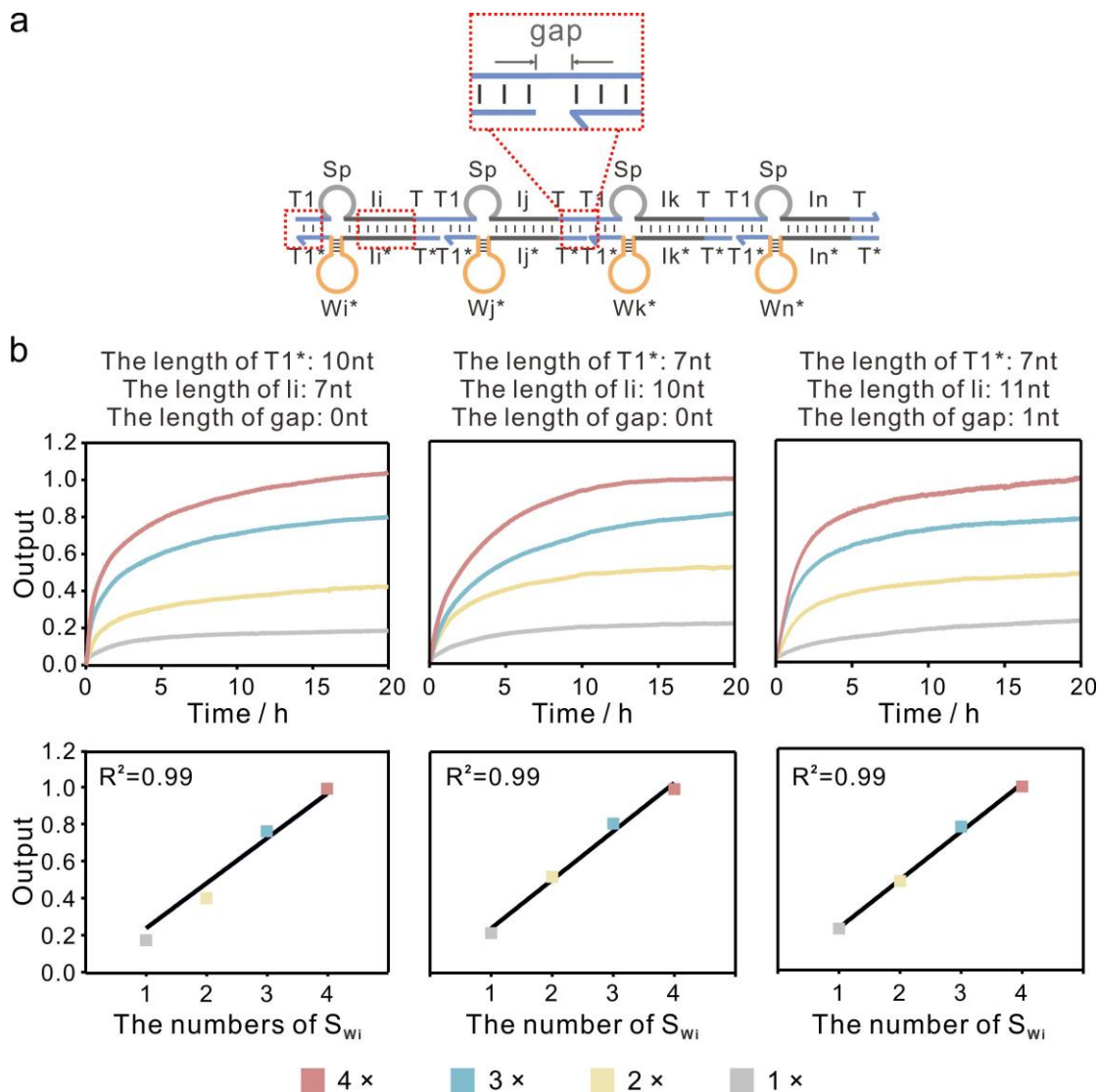


Fig. S49| The optimization of molecular design of weight multiplication. **a**, Schematic illustrating the molecular details of the weight multiplication. **b**, Experimental validation of the weight multiplication under different conditions. The fluorescence kinetics (top) and the fluorescence level (bottom) of different ssDNA corresponding to 1, 2, 3 or 4 weight modulators S_{Wi} activating the weight substrate molecules $M_{Wi,li,j}$. The results shows that the 1 nt gap design, along with 11 nt recognition domain length and 7 nt toehold domain length design, provided higher computing speed. Thus, this molecular design was applied in our work.

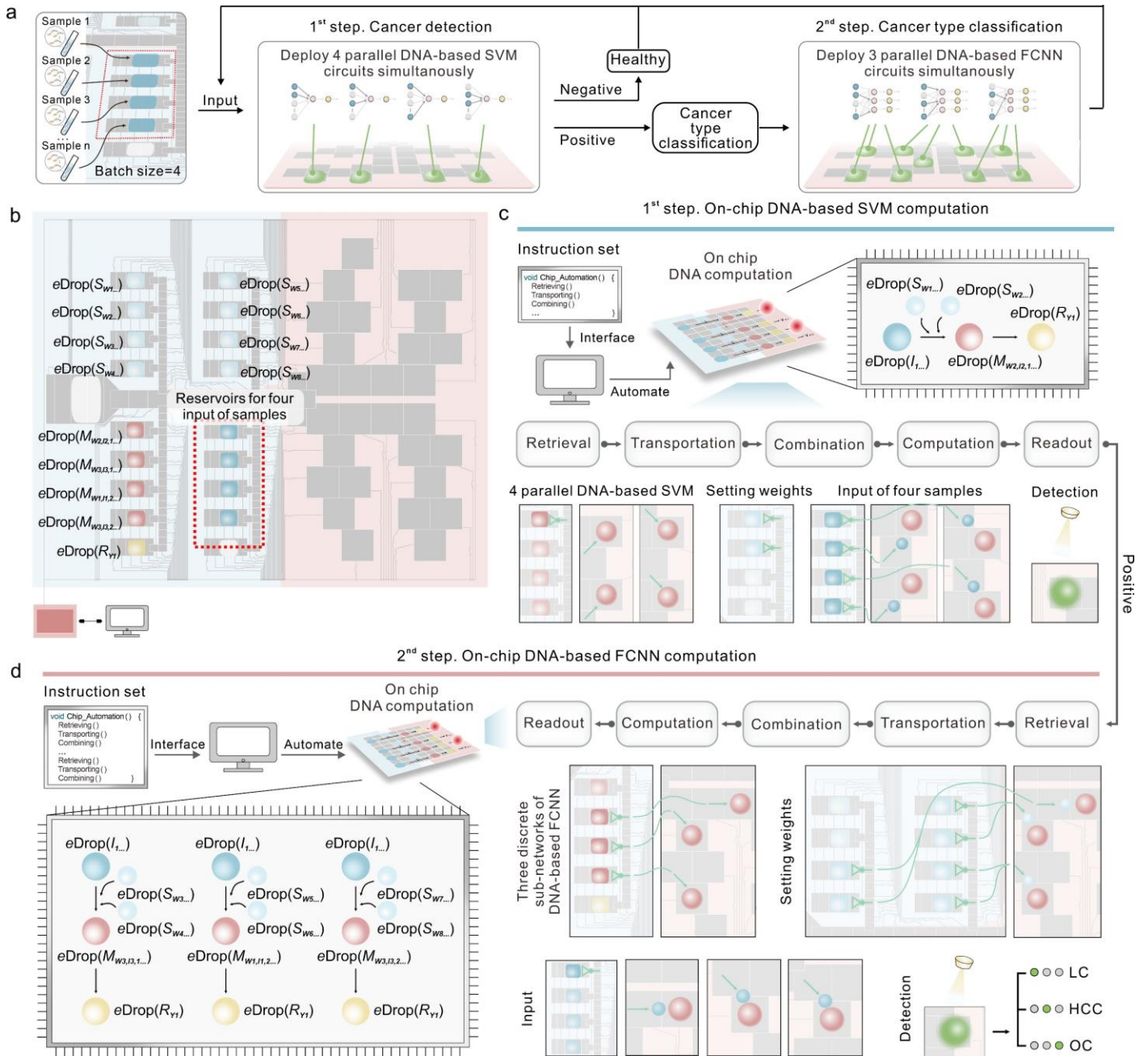


Fig. S50| The complete workflow of DMIC for diagnosis of multiple cancer types. **a**, The initial state of DMIC. When the samples to be tested is input, DMIC will first configure four on-chip DNA-based SVM in four isolated reaction chambers for synchronous diagnosis (**b**). If the sample is negative, it means the end of the diagnosis and unknow sample will continue to be input for cancer presence detection; If the sample is positive, DMIC will then configure corresponding on-chip DNA-based FCNN to determine the type of cancer (**c**). Once the reaction chamber is free, unknown samples will be input immediately, and DMIC will continue to configure corresponding DNA circuits, allowing the synchronization of multiple diagnostic tasks to maximize the efficiency of on-chip molecular diagnosis. The volume of droplets located in the memory is 2.50 μL (DNA species $M_{w_i,l,i,j}$ and S_{w_i} concentration: 25 $\times$, DNA species $S_{d_j,k}$ and D_{d_k,Y_i} concentration: 300 $\times$, DNA species Sub_{j,Y_i} concentration: 500 $\times$, DNA species R_{Y_i} concentration: 100 $\times$). The volume of droplets aliquoted from the memory is 0.32 μL , while the volume of aliquoted reaction buffer droplet is 1.60 μL . The standard concentration is 25 nM (1 $\times$).

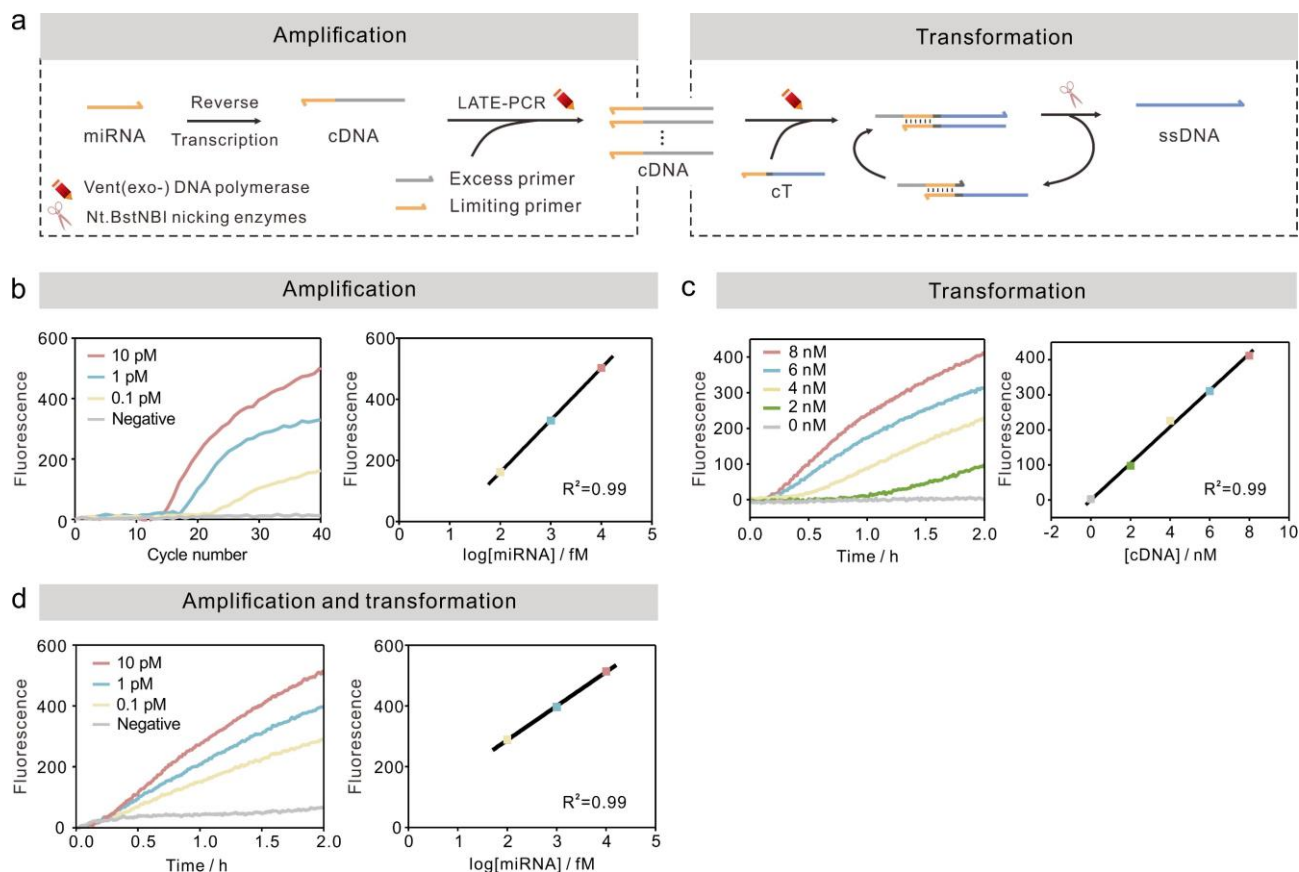


Fig. S51| miRNAs amplification and transformation. **a**, Schematic illustration of the amplification and transformation of miRNAs to long ssDNA. miRNAs were firstly reversely transcribed into cDNA using a reverse transcription kit. The obtained cDNAs were then amplified by linear after the exponential PCR (LATE-PCR), enabling close-to-linear amplification of miRNA, with respect to their logarithmic initial concentrations in a specific range^{14,15}. LATE-PCR starts with an exponential phase similar to symmetric PCR and, upon depletion of the limiting primer, transitions to linear amplification that produces ssDNA over many cycles. This reduces concentration bias from the exponential phase and yields amplified ssDNA with quantitative ratios reflecting the original logarithmic concentrations of the miRNA inputs. Moreover, since miRNA inputs were logarithmically transformed during *in silico* training, LATE-PCR preserves their quantitative information after amplification when concentrations fall within the linear range. By controlling the amplification cycle numbers, the amplified ssDNA products can be maintained in a reasonable concentration range to provide optimal conditions for subsequent on-chip DNA computation. To enable more orthogonal sites for subsequent weight multiplication, a converter template cT was designed to linearly transform an amplified cDNA into a long ssDNA with four orthogonal domains, without perturbing their original quantity information. The transformed ssDNA was purified for the inputs of the on-chip molecular classifier. The concentrations of ssDNA products were analyzed using designated molecular beacons that are capable of generating a fluorescent signal in the presence of target ssDNA. **b**, Experimental validation of the amplification step. Left: Detection of the amplified cDNA products with different miRNA initial concentrations (ranging from 0.1 to 10 pM). Right: The plot of fluorescence response at cycle 40 versus initial logarithmic miRNA concentrations, showing a linear relationship. **c**, Experimental validation of the transformation step. Detection of the transformed ssDNA products with different cDNA initial concentrations (ranging from 2 to 8 nM) (left, top). The plot of fluorescence response at 2 h versus initial cDNA concentrations, demonstrating the linear transformation behaviour (right, top). **d**, Experimental validation of the amplification and transformation steps. Left: Detection of the ssDNA products amplified and transformed from different miRNA initial concentrations (ranging from 0.1 to 10 pM). Right: The plot of fluorescence response at 2 h versus initial logarithmic miRNA concentrations.

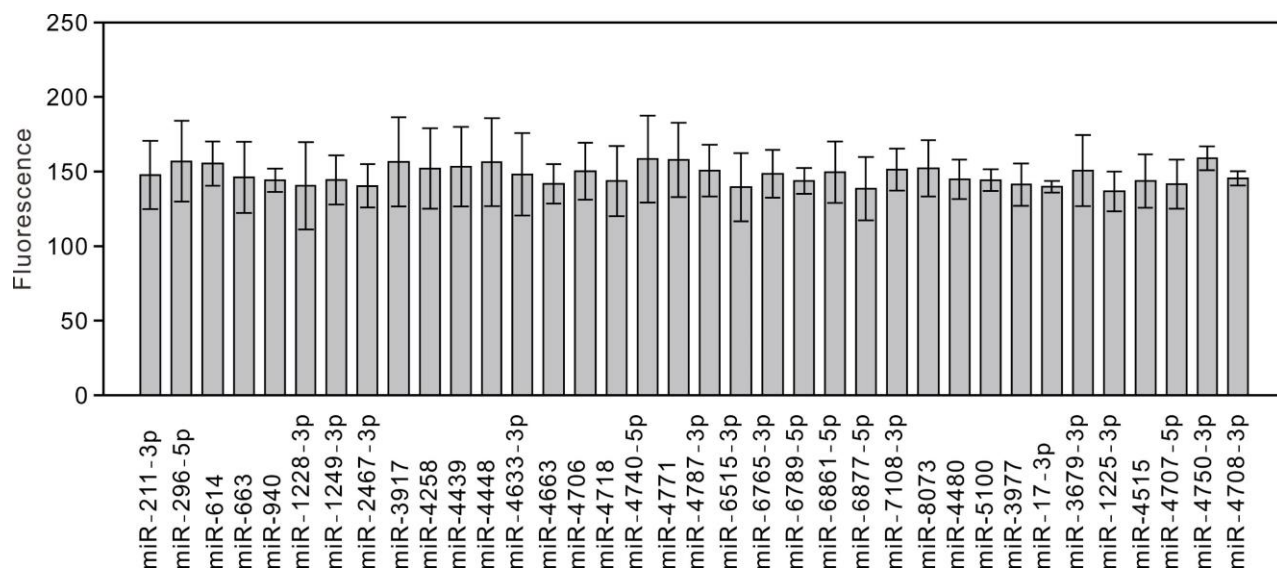


Fig. S52| The amplification efficiency for 36 selected miRNAs. 0.1 pM of each 36 synthetic miRNA was added into 36 separated PCR tubes for LATE-PCR amplification. The fluorescence of intensities of the amplified cDNA products at cycle 40. Error bars indicate standard deviations (mean $\pm$ s.d., n = 3).

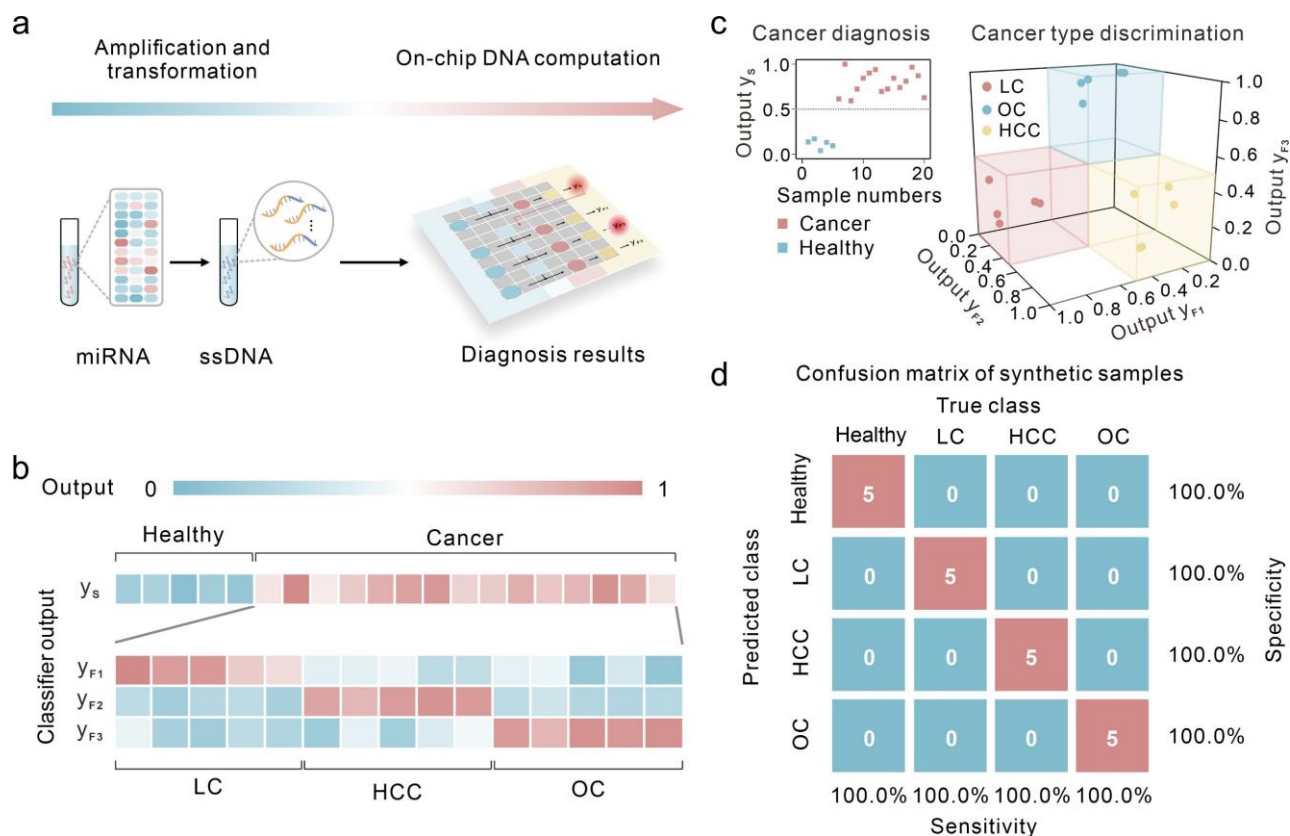


Fig. S53| Experimental validation of DMIC on synthetic samples. **a**, Complete experimental workflow for validation on 20 synthetic samples. The synthetic miRNAs are amplified by LATE-PCR and transformed to long ssDNA for the subsequent on-chip DNA computation without perturbing their original quantity ratios. The fluorescence signals on DMIC were recorded for classification. **b**, Performance of DMIC on synthetic samples. Each sample corresponds to one or two specific output signals, depending on whether it requires the step of cancer type diagnosis. **c**, Visualization of the experimental results of 20 synthetic sample. Colored-bordered region indicates the space for the specific cancer type. **d**, Confusion matrix analysis of 20 synthetic samples.

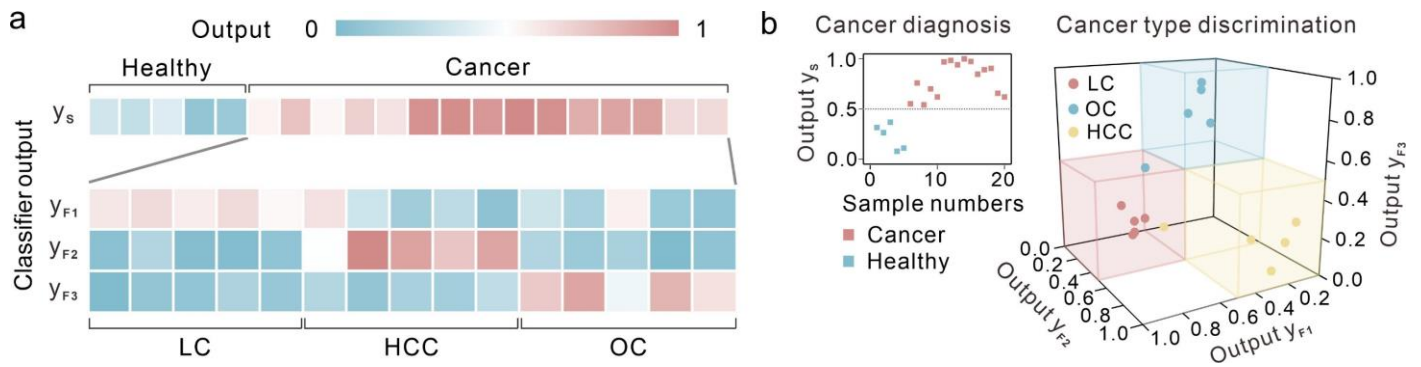


Fig. S54| Experimental validation of DMIC on clinical samples. a, Performance of DMIC on 20 clinical samples. Each sample corresponds to one or two specific output signals, depending on whether it requires the step of cancer type diagnosis. **b**, Visualization of the experimental results of 20 clinical sample. Colored-bordered region indicates the space for the specific cancer type.

References

1. Tubagere, A. J., Tachuk, C., Berleant, J., Johnson, R. F. & Qian, L. Compiler-aided systematic construction of large-scale DNA strand displacement circuits using unpurified components. *Nat. Commun.* **8**, 14373 (2017).
2. Qian, L. & Winfree, E. Scaling up digital circuit computation with DNA strand displacement cascades. *Science* **332**, 1196–1201 (2011).
3. Zadeh, J. N. et al. NUPACK: analysis and design of nucleic acid systems. *J. Comput. Chem.* **32**, 170–173 (2011).
4. Lai, W. et al. Programming chemical reaction networks using intramolecular conformational motions of DNA. *ACS Nano* **12**, 7093–7099 (2018).
5. Ma, H. et al. Large-area manufacturable active matrix digital microfluidics platform for high-throughput biosample handling. In *2020 IEEE International Electron Devices Meeting (IEDM)* 35.5.1–35.5.4 (IEEE, 2020).
6. Jia, Z. et al. Intelligent single-cell manipulation: LLMs-and object detection-enhanced active-matrix digital microfluidics. *MICROSYST. NANOENG.* **11**, 133 (2025).
7. Hu, S. et al. Large-area electronics-enabled high-resolution digital microfluidics for parallel single-cell manipulation. *Anal. Chem.* **95**, 6905–6914 (2023).
8. Wang, D. et al. Active-matrix digital microfluidics design for field programmable high-throughput digitalized liquid handling. *Science* **27** (2024).
9. Ji, J. et al. AM-DMF-ddRPA: an All-in-One Digital Microfluidic Platform for Rapid and Automatic Digital Nucleic Acid Analysis. *Angew. Chem. Int. Ed.* **64**, e202501913 (2025).
10. Zhou, X. F. Understanding the convolutional neural networks with gradient descent and backpropagation. *J. Phys.: Conf. Ser.* **1004**, 012028 (2018).
11. Kingma, D. P. & Ba, J. A. A method for stochastic optimization. In *Proc. International Conference on Learning Representations* (ICLR, 2015).
12. Phillips, M., & Likhachev, M. In 2011 IEEE International Conference on Robotics and Automation (ICRA, 2011).
13. LeCun, Y., Bengio, Y., & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
14. Sanchez, J. A., Pierce, K. E., Rice, J. E. & Wangh, L. J. Linear-After-The-Exponential (LATE)-PCR: An advanced method of asymmetric PCR and its uses in quantitative real-time analysis. *Proc. Natl. Acad. Sci.* **101**, 1933–1938 (2004).
15. Pierce, K. E., Sanchez, J. A., Rice, J. E. & Wangh, L. J. Linear-After-The-Exponential (LATE)-PCR: Primer design criteria for high yields of specific single-stranded DNA and improved real-time detection. *Proc. Natl. Acad. Sci.* **102**, 8609–8614 (2005).