Molecular convolutional neural networks with DNA regulatory circuits

Xiewei Xiong, Tong Zhu, Yun Zhu, Mengyao Cao, Jin Xiao, Li Li, Fei Wang, Chunhai Fan and Hao Pei

Complex biomolecular circuits enabled cells with intelligent behaviour to survive before neural brains evolved. Since DNA computing was first demonstrated in the mid-1990s, synthetic DNA circuits in liquid phase have been developed as computational hardware to perform neural network-like computations that harness the collective properties of complex biochemical systems. However, scaling up such DNA-based neural networks to support more powerful computation remains challenging. Here we present a systematic molecular implementation of a convolutional neural network algorithm with synthetic DNA regulatory circuits based on a simple switching gate architecture. Our DNA-based weight-sharing convolutional neural network can simultaneously implement parallel multiply-accumulate operations for 144-bit inputs and recognize patterns in up to eight categories autonomously. Further, this system can be connected with other DNA circuits to construct hierarchical networks to recognize patterns in up to 32 categories with a two-step approach: coarse classification on language (Arabic numerals, Chinese oracles, English alphabets and Greek alphabets) followed by classification into specific handwritten symbols. We also reduced the computation time from hours to minutes by using a simple cyclic freeze–thaw approach. Our DNA-based regulatory circuits are a step towards the realization of a molecular computer with high computing power and the ability to classify complex and noisy information.

Although ConvNet algorithms enabling efficient hardware implementation in electronic computing devices and in photonic and quantum computing devices have been proposed, they have not yet been demonstrated in molecular computing systems. Here, we present a systematic strategy for the computational design of networks of reactive orthogonal DNA molecules that can implement the ConvNet algorithm. We show that such a DNA-based ConvNet can simultaneously implement multiple parallel multiply–accumulate (MAC) operations and recognize patterns in up to eight categories after training in silico. Each pattern comprises up to 43 distinct DNA strands that are selected to trace individual handwritten symbols, chosen from a set of 144 DNA strands that represent the 144 bits in a 12×12 pattern. By connecting upstream logic circuits that activate a specific set of weight molecules, the DNA-based ConvNet can recognize patterns in up to 32 categories by using a two-step classification approach of performing a coarse classification on type of language (Arabic numerals, Chinese oracles, English alphabets and Greek alphabets) and then classifying into specific handwritten symbols. Moreover, we show that a simple cyclic freeze–thaw approach can significantly accelerate the reactions in such large-scale DNA neural networks, which decreases the computation time from hours to minutes. Our approach leads towards the realization of molecular computers with high computing power and the ability to classify complex and noisy information.

Results

A DNA switching gate architecture design for ConvNet circuits.

A ConvNet consists of an input layer, a convolutional layer, a non-linear layer and an output layer. In each layer, an intermediate array of pixels, referred to as a feature map, is produced from the previous layer. Figure 1a illustrates the operating principle of the ConvNet for recognition tasks, where an \( n \times n \) input symbol is convolved with a weight vector of \( n \times n \) entries to produce a feature map.
Fig. 1 | The ConvNet and its molecular implementation with DNA regulatory circuit systems. **a**, Left: The architecture of the ConvNet. Right: A schematic of the operating principle of the ConvNet for recognition tasks. The individual receptive region of the inputs is multiplied by the kernel to perform a MAC operation to output \( y \), where we assume that the stride is 1. In this way, an input matrix can be convolved with the same kernel by mapping the convolution computation into \( (n-k+1)^2 \) MAC operations. Note that \( x_i \) \((1 \leq i \leq n^2)\) are binary inputs and \( w_{i,j} \) \((1 \leq t \leq k)\) are analogue weights. The subscripts \( i \) and \( t \) represent the position of elements in the input matrix \( X \) and of the kernel matrix \( W \), respectively. **b**, The implementation of the ConvNet with DNA regulatory circuit systems. Upper: The inputs can be encoded by a set of single strands, wherein 1 or 0 represent the presence or absence of input strands. Middle: The weight matrix can be stored by programming a set of weight molecules with a simple switching gate architecture. The assignment of shared weights is implemented by the sequence of weight tuning domains. Bottom: The DNA implementation of summation and subtraction. In all panels, symbols with subscripts denote DNA species (enclosed in coloured solid as well as dotted circles), whereas those without subscripts refer to the elementary functional domains (represented by coloured lines) of DNA species. The presence of an asterisk when denoting domains indicates a complementary sequence. The symbol \( \Sigma \) indicates the sum over all inputs.

A \( k \times k \) kernel function (with a stride of 1) to compute a feature map with dimensions of \( (n-k+1) \times (n-k+1) \). When operating the ConvNet, the input symbol is grouped into \( (n-k+1)^2 \) receptive regions (Fig. 1a, area marked with dashed blue line) with dimensions of \( k \times k \). The elements of these receptive regions share the same weights, which could enable a sparse topology to effectively reduce the number of network connections. Mathematically, a convolution operation requires multiple MAC operations with shared weights. To implement a weight-sharing MAC operation using DNA molecules, we proposed a switching gate architecture\(^a\). Each switching gate is associated with a gate base strand (for example, domains \( T^*W^*I^*T^* \) in Fig. 1b) that has a recognition domain (\( I^* \) in Fig. 1b) and a weight tuning domain (\( W^* \) in Fig. 1b) flanked by two toehold domains (\( T^* \) in Fig. 1b), and these domains are functionally independent. Here and below, symbols such as \( T \), \( W \), and \( I \) denote elementary functional domains of DNA species, whereas the same symbols with an asterisk are used to denote their complementary sequences. Varying the sequence of the recognition domain
(I²) enables it to respond to different inputs, leading to different signal transmission pathways. Varying the sequence of the weight tuning domain (W*) determines the weights assigned to the input, whereby the same weight can be assigned to different inputs by concatenating the same weight tuning domain and different recognition domains. This architecture could allow independent control of the signal transmission functions and weight assignment functions during the computation. In this way, we can implement weight-sharing MAC operations at the molecular level and construct molecular ConvNets with reactive orthogonal DNA molecules.

**DNA implementation of MAC and convolution operations.** We started the experimental demonstration with the weight multiplication function \( w(x, y) \), in which the variables \( x \) and \( y \) are binary inputs and the variables \( w_i \) \( (1 \leq i \leq n^2) \) are analogue weights. The subscripts \( i \) and \( t \) represent the position of the elements of the input matrix \( X \) and of the kernel matrix \( W \), respectively (Fig. 1). The positions of the matrix elements correspond to different sequences of the recognition domain (I) and of the weight tuning domain (W) in the DNA strand displacement implementation (Extended Data Fig. 1). The binary values taken by the variables \( x \) correspond to the presence or absence of domain \( I \) in input strand \( X \), while the values of the variables \( w_i \) correspond to the concentrations of the weight tuning molecules \( M_{w_i} \) or weight substrate molecules \( N_{w_i,I} \) that contain the domain \( W_t \), where the symbols with subscripts such as \( M_{w_i} \) and \( N_{w_i,I} \) denote DNA species. Weight multiplication is implemented by cascaded reactions (Fig. 2a) where an input species \( X \) converts an activated weight substrate molecule \( N_{w_i,I} \) to an intermediate product \( P_{w,I} \), \( N_{w_i,I}^{*} \) is implemented when \( N_{w_i,I} \) undergoes a spontaneous intramolecular conformational switch upon hybridization with the weight tuning molecule \( M_{w_i} \) allowing stoichiometric exchange of the activity of the DNA signals. In the absence of \( X \), no \( P_{w,I} \) will be generated. In the presence of \( X \), the final concentration of \( P_{w,I} \) is determined by the concentration of \( N_{w_i,I}^{*} \), thus setting the value of the weighted multiplication (Supplementary Figs. 2–9 and Fig. 2b). Note that negative weights assigned to inputs are implemented using distinct output sequences (domain \( P_j \) in Extended Data Fig. 1) for \( N_{w_i,I} \). Then, one can compute the sum of the weighted inputs within the same receptive region by using the same output sequence for all the intermediate species. This is implemented with reactions wherein all the intermediate species \( P_{w,I} \) stoichiometrically convert the summation gate (\( S_d, j \)) to common weighted-sum species \( S_{S_j} \) (Fig. 2c,d). The sums of positive and negative weights can be implemented by using the concentrations of two different weighted-sum species (Supplementary Fig. 10). To complete the summation, the positive and negative weighted sums must be subtracted from one another (Fig. 2e). Specifically, all positive weighted-sum species \( S_{S_j} \) can convert the double-stranded complex \( D_{S_j,Y} \) to an intermediate species \( D_{S_j,Y}^{*} \). All negative weighted-sum species \( S_{S_j} \) generated from the upstream summation gate can bind to the toehold of the inhibitory strand \( I_n \) and branch migrate to form inert waste species, producing reactive annihilation species \( S_{S_j}^{*} \) through intramolecular conformational switching. The subtraction can thus be realized, wherein \( S_{S_j}^{*} \) and \( D_{S_j,Y}^{*} \) annihilate each other. Only the remaining \( D_{S_j,Y}^{*} \) will interact with the downstream reporting gate (Fig. 2f) to read the output signal. Otherwise, the reaction is terminated. This design suggests that only positive weighted-sum species \( S_{S_j} \) could activate the downstream reactions, a feature that is shared with that of nonlinear activation function, known as the rectified linear unit (ReLU) used in this model.

**ReLU** = \[
\begin{cases} 
  x & \text{if } x > 0 \\
  0 & \text{otherwise}
\end{cases}
\]

We verified this design by using 36 different weighted-sum (\( S_t \) and \( S_{\gamma} \)) combinations (Fig. 2g). As expected, the signal was indeed gradually attenuated as the concentration of \( S_{\gamma} \) was increased. Note that this circuit does not implement a perfect subtraction operation. This is because a small fraction of \( S_{S_j} \) could interact directly with the downstream complex before encountering \( S_{S_j}^{*} \) (the higher the concentration, the greater its chance of escaping from complete annihilation), similar to observations in previous work1. Despite this imperfection, the circuit does compute correctly for weighted sums when the concentrations of the weighted-sum species are not too close to each other and are not too high.

Subsequently, we designed a DNA-based MAC circuit to perform a two-input MAC operation \( y = w_1x_1 + w_2x_2 \), which is implemented by adding one summation gate to two parallel weight multiplications (Extended Data Fig. 2a,b). The circuit consists of two weight substrate molecules \( N_{w_1,I} \) and two weight tuning molecules \( M_{w_1} \) and one summation gate. Each \( N_{w_1,I} \) that is activated by the corresponding \( M_{w_1} \) has a different recognition domain (I* and I*) to respond to distinct inputs \( (X_1 \) and \( X_2 \) ), and each has a different weight tuning domain (W1* and W2*) to determine the weights assigned to the input. Using the same output sequence for two \( N_{w_1,I} \) would enable two intermediate species \( P_{w_j} \) to be connected to the same summation gate. We can tune the concentration of the weight tuning molecules \( (M_{w_1} \) and \( M_{w_2} \)) to set different values for the weights for the corresponding multiplication reactions. As expected, the circuit exhibits stoichiometric behaviour with the output signal and the concentration of \( M_{w_1} \) (Extended Data Fig. 2c,d). By adding one extra weight multiplication to the downstream summation gate, we also demonstrated a circuit that computes the three-input MAC function (Supplementary Fig. 11).

We now show the molecular implementation of a simple convolution operation of a 2 × 2 input pattern with a kernel size of 2 × 1, which is equivalent to two parallel MAC operations in mathematics. Each receptive region of the input patterns \( x_1 \) and \( x_2 \) and \( x_3 \) and \( x_4 \) is multiplied by the weights \( w_1 \) and \( w_2 \) to obtain the weighted pixels \( x_1w_1 \) and \( x_2w_2 \) and \( x_3w_1 \) and \( x_4w_2 \), respectively. The feature map \( y_1 \) and \( y_2 \) is then exported by summing up the weighted pixels in the same receptive region (Fig. 3a). The 2 × 2 input pattern is encoded with four DNA input strands (\( X_1 \), \( X_2 \), \( X_3 \), and \( X_4 \)). The convolution kernel is encoded in the sequence of weight tuning domains (Fig. 3b, green domains W1* and W2*). To complete the multiplication with shared weights, we designed four weight substrate molecules \( N_{w_1,I} \) and \( N_{w_2,I} \) (and \( N_{w_1,Y} \) and \( N_{w_2,Y} \) in Supplementary Fig. 12) of which (for example, \( N_{w_1,I} \) and \( N_{w_1,Y} \) in Supplementary Fig. 12) have the same weight tuning domain, corresponding to the pixels that interact with the same kernel in different local receptive regions, but different recognition domains at the 3’ end to connect to the downstream summation gates (\( S_{d,1} \) and \( S_{d,2} \) ). All the input patterns were binary patterns in which 1 or 0 represents the presence or absence of input strands. The value of the analogue weights determined from the convolution kernel is implemented with the concentrations of \( N_{w_1,I} \). To compute the convolution, each DNA sub-circuit runs independently and in parallel to compute the MAC operation in each receptive region (Supplementary Fig. 13). For a specific pattern, the corresponding weight tuning molecules \( M_{w_1} \) would activate the respective weight substrate molecules \( N_{w_1,I} \) in two respective regions. Thus, the assignment of shared weights by the convolution kernel is implemented with the activated weight substrate molecule \( N_{w_1,I}^{*} \) and \( X_1 \) through a DNA strand displacement reaction, resulting in the release of the intermediate species \( P_{w_j} \). Two summation gates \( S_{d,j} \) convert \( P_{w_j} \) in the same receptive regions to weighted-sum species \( S_{S_{\gamma}} \) leading to the triggering of the downstream reporting gates. For an experimental demonstration, we chose 16 input patterns, and both outputs achieved their correct ‘on’ or ‘off’ state, indicating that the DNA circuit correctly implemented the convolution computation (Fig. 3c and Supplementary Fig. 14). For example, with inputs \( X_1 \), \( X_2 \), \( X_3 \), \( X_4 = 1001 \), the circuit output \( y_1 \) (or \( y_2 \)) was proportional to the product of the input concentration and the corresponding weight, as designed.
A DNA-based ConvNet for molecular pattern recognition. Having shown that the DNA circuit is capable of processing the convolution, we next built a DNA-based ConvNet that can ‘remember’ two handwritten symbols, that is the Chinese oracles for ‘fire’ and ‘earth’ (Fig. 4a). The training set consists of 48,000 patterns of handwritten symbols from the Simica oracle database. In silico, all the original symbols were converted to 144-bit binary patterns for network training by rescaling them to a 12 x 12 grid and setting each pixel to 1 when exceeding a threshold (Supplementary Fig. 15). The convolution kernel (a 6 x 6 matrix) slides along the input patterns with a stride of 6 and subsequently generates a corresponding output feature map (Fig. 4b) and Neural network training and testing section). We evaluated the network performance on a reference dataset after training, reaching an accuracy of 97% (Fig. 4c). We implemented this ConvNet model by encoding the convolution kernel in the sequence of the weight tuning domain, and implementing the value of the weights with the concentration of the weight substrate molecule N_0(0)(Extended Data Fig. 3a). The test input binary patterns were encoded with single strands, wherein each 1 or 0 represents the presence or absence of an input strand (Fig. 4d). The DNA-based ConvNet implements pattern recognition by comparing its local feature with all the memories and identifying the most similar memory (Extended Data Fig. 3b). For example, each receptive region of a ‘fire’ can simultaneously interact with the same...
**Fig. 3 | A convolution computation via multiple parallel MAC operations.**

**a.** The detailed calculation process of the convolution with a $2 \times 2$ input matrix, a kernel size of $1 \times 2$ and a stride of 1. The feature map is generated from the interactions between the kernel function and different receptive regions.

**b.** Top: An abstract schematic diagram of the convolution. The symbol $\sum$ indicates the sum over all inputs. The $2 \times 2$ input pattern is encoded with four DNA single strands ($X_1$, $X_2$, $X_3$, and $X_4$). The weight substrate molecule $N_{\text{NW1}}$ can be activated by the corresponding weight tuning molecule $M_{\text{W1}}$, and then the activated weight substrate molecule $N_{\text{NW1}}$ can interact with the local receptive region and export the computation results. Middle: The DNA implementation of a MAC operation in one receptive region ($X_1$ and $X_3$) of the input pattern. In this receptive region, two $N_{\text{NW1}}$, one $S_{\text{d1}}$, one $\text{Rep}_i$, and two $M_{\text{Ob}}$ were designed. Bottom: The detailed reaction pathway of a MAC operation with input $X_1$ in one receptive region. Arrows indicate the flows of the reactions. DNA species are represented by coloured solid and dotted circles, whereas different domains are represented by coloured lines. The full DNA strand displacement cascades implementing a convolution are shown in Supplementary Fig. 13. **c.** The characterization of a convolution with six different input patterns after 3 h. The value of the weights determined from the convolution kernel is 0.8x (blue dot) and 0.2x (grey dot), respectively. The blue and red histograms correspond to the outputs of $y_1$ and $y_2$, respectively. Blue dots indicate that the weight $w_i$ is multiplied by the corresponding input ($x_i$ or $x_j$), while grey dots indicate that the weight $w_i$ is multiplied by the corresponding input ($x_i$ or $x_j$). The relative concentrations of $X_i$, $\text{Rep}_i$, and $S_{\text{d1}}$ are 2x. The relative concentrations of $M_{\text{Op}}$ and $M_{\text{W1}}$ are 3.2x and 0.8x. The relative concentrations of $N_{\text{NW1}}$ and $N_{\text{NW2}}$ are 0.8x, while the relative concentrations of $N_{\text{W21}}$ and $N_{\text{W214}}$ are 0.2x. The standard concentration is $1x = 50$ nM.
kernel function to export feature maps through DNA strand displacement cascades. As the network runs, a subset of weight tuning molecules \( m_{W9} \) could activate the corresponding weight substrate molecule \( N_{W9,I1} \) in four receptive regions at the same time to enable multiple weight-sharing MAC operations to be performed in parallel. This allows DNA circuits to be able to activate a specific reaction pathway in the convolution layer when exposed to a specific pattern, which can enhance the robustness of the network (Supplementary Fig. 16). Then, a max-pooling process (with a pooling size of \( 2 \times 1 \) and stride of \( 1 \)) is applied to reduce feature the size of the map by annihilating the smaller one between two pixels through the cooperative hybridization method introduced by Cherry and Qian:\(^{8}\) (Extended Data Fig. 3c), in which two contiguous pixels are represented by concentrations of two distinct nucleic acid sequences.

As shown in the experimental data (Fig. 4e and Supplementary Fig. 17), the input patterns of two handwritten symbols each triggered the desired outputs, indicating that two handwritten symbols can be classified. When each oracle pattern was rotated by an angle from \( 0^\circ \) to \( 360^\circ \) in steps of \( 30^\circ \), the circuit still yielded the desired output for all 26 test input patterns, indicating that the circuit correctly classified the rotated patterns. In total, \( 177-250 \) distinct molecules were used for all the test patterns. As expected, we showed that the DNA-based ConvNet can also remember eight 144-bit molecules simultaneously (Supplementary Figs. 18–22).

A hierarchical neural network for recognizing 32 patterns. Our ConvNet has a feature where different input weight tuning molecules can selectively activate a specific set of weights to allow the
Fig. 5 | The two-step classification approach based on a hierarchical network architecture for the recognition of 32 molecular patterns. 

a. Each input pattern is replaced by a pair of input patterns (layer 1 and layer 2). Layer 1 was attached with tags to classify four groups, where 1000 corresponds to Chinese oracles, 0100 corresponds to Arabic numerals, 0010 corresponds to English alphabets and 0001 corresponds to Greek alphabets. Both the tags in layer 1 and the inputs in layer 2 are binary patterns, in which each 1 or 0 indicates the presence or absence of a tag strand (or an input strand), respectively.

b. The circuit diagram for recognizing eight distinct patterns of each group. The symbol $\sum$ indicates the sum over all inputs.

c. The performance of the ConvNet in silico for the dataset of eight English alphabets.

d. An example pattern recognition process of ‘fire’ with a pair of input patterns. Layer 1 is the input of the logic circuit, which is composed of the reporter gates R and the fan-out gates F. The outputs of the fan-out gates can activate the downstream ConvNet circuit to complete the pattern recognition, while the outputs of the reporter gates can be read out to complete the coarse classification.

e. The circuit diagram. Specific weight tuning molecules can switch on corresponding circuit components for specific molecular pattern groups. Coloured lines in DNA strands indicate different functional domains.

f. The characterization of the recognition behaviour of 32 representative input patterns after 36h. The tag highlighted above the bar graph marks the group of each input pattern. The dotted light-grey line marks the threshold value of 0.6 (with ON shown as red and OFF as grey in the histograms). The corresponding fluorescence kinetics data are shown in Supplementary Figs. 24 and 25.
same set of DNA molecules to be used for different tasks. The use of weight tuning molecules as outputs of the upstream circuit demonstrates the possibility of building hierarchical networks for more sophisticated categorization tasks. To validate this approach, we proposed a two-step classification approach that first uses a logic gate to perform coarse classification and then uses a ConvNet to perform finer classification. To demonstrate this approach experimentally, we chose a task of recognizing 32 handwritten symbols that can be divided into 4 groups: 8 Chinese oracles (from the Sinica oracle database), 8 Arabic numerals (from the Modified National Institute of Standards and Technology database), 8 English alphabets and 8 Greek alphabets (from the Kaggle website). In silico, we converted all the original handwritten symbols to binary patterns with two layers (Fig. 5a). Layer 1 is on a 1×4 grid and acts as an input for the logic circuits to perform a coarse classification on language (for example, oracle is 1000), yielding outputs that selectively activate the downstream ConvNet subnetwork to perform fine classification into specific handwritten symbols using layer 2 on a 12×12 input pattern. Four groups in layer 2 can be separately trained in silico with respective datasets to obtain the optimal model (Fig. 5b), thus yielding the values of four convolution kernels with dimensions of 3×6 (with a stride of 3×6) (Supplementary Fig. 23 and Neural network training and testing section). This network performed well in the reference dataset, reaching >84% accuracy in each group (Fig. 5c). We implemented a two-step classification approach experimentally by designing different switching gates to encode the four convolution kernels. Both the tags in layer 1 and the inputs in layer 2 are binary patterns, in which each 1 or 0 indicates the presence or absence of a tag strand (or an input strand), respectively (Extended Data Fig. 4a). The pattern classification can be performed by using the following steps (Fig. 5d): (1) The tag strand in layer 1 reacts with the reporter gate to generate the output signal $y^n$, which can be recognized as the corresponding coarse category (Extended Data Fig. 4b, c). Meanwhile, the tag strand reacts with the fan-out gate to release the set of weight tuning molecules, which can be captured from total DNA strands using magnetic beads through a hybridization reaction using biotinylated capture probes. Then, the weight tuning molecules are displaced from the beads by an invader strand to activate the downstream DNA neural networks (Extended Data Fig. 4b–f). (2) The weight tuning molecules generated from the upstream logic
A cyclic freeze–thaw approach to accelerate the DNA circuits. The speed of execution of DNA computing remains a challenge in large-scale DNA neural network reactions. For example, our computation of two categories took longer than 20 h (Supplementary Fig. 17), while the computation time increased to over 36 h for 32 categories (Supplementary Fig. 25). To accelerate the performance of these DNA circuits, we developed a simple cyclic freeze–thaw approach (Fig. 6a). The cyclic freeze–thaw approach iteratively drives the strand displacement reaction towards thermodynamic equilibrium, which can accelerate the basic strand displacement reaction by ~75 fold (Extended Data Fig. 5 and Supplementary Fig. 27). For a larger-scale circuit, 160 test patterns of 144 bits can be recognized in less than 30 min through five freeze–thaw cycles, which would otherwise require hours (Fig. 6b,c).

Conclusion
We have experimentally implemented a DNA-based ConvNet based on 512 DNA strands that can perform eight parallel weight-sharing MAC operations simultaneously. The switching gate architecture in the design of the DNA circuits allows independent control of the signal transmission functions and the weight assignment functions, which is functionally similar to the riboswitches in gene regulatory circuits, all consisting of two independent functional domains that sense and respond to external inputs. This hints at the possibility of embedding ligand-responsive molecular switches to allow biochemical circuits to adapt their functions in response to changes in the environment34–37. The fact that we were able to use one additional circuit to selectively activate a specific set of weights in the DNA-based ConvNet shows the potential to embed learning into biochemical circuits, in which the upstream DNA circuit updates the current approximation of the weights in supervised learning tasks29–32. The massively parallel feature inherent to DNA molecules could enable autonomous parallelization of convolution operations, which would be particularly well suited for more scalable information processing. By using a hierarchical network that first uses a logic gate to perform coarse classification and then a ConvNet to perform finer classification, we also demonstrated that such circuits can be scaled up to classify patterns into 32 categories. This could also provide the possibility of integrating multiple circuit architectures33 to enhance the computational power. Further scaling up would exacerbate problems such as leak reactions caused by non-specific cross interactions or spurious binding.

Importantly, we have extended the key feature of ConvNets, that is, sparse topology, to a DNA neural network, effectively reducing the complexity of the network architecture by using sparsely connected neurons, which could allow more complex information processing and potentially endow molecular circuits with intelligent behaviour that resembles a biological neural network. By interfacing sensory inputs44–47, the DNA-based ConvNet could in principle use hundreds of targets as inputs, facilitating broader applications in disease diagnostics, profiling expression patterns and precision medicine44–46.

Methods

Sequence design. Five types of molecular structure (Extended Data Fig. 1) were used in this work: the weight substrate molecules $N_{W_{ij}}$, (1) and subtraction species $N_{S_{ij}}$ (4) consist of three single strands; the summation gates $N_{D_{ij}}$ (2) and $N_{D_{ij}}$ (3) consist of two single strands; the reporters $N_{R_{ij}} $ (5) consist of single strands modified with fluorophore and quencher groups. All the DNA single strands used in this work were composed of long recognition domains and short toehold domains, except for the weight tuning domains used for weight multiplication, subtraction and reporting, which were composed of short toehold domains and long loop domains. Note that these domains are functionally independent. Corresponding complementary sequences are indicated by an asterisk. On this basis, the sequence design was conducted at the domain level.

We generated several pools of domain sequences with different lengths according to a series of design heuristics45,46. To reduce secondary structures and undesired interactions, all the domain sequences were produced by using a three-letter code (A, C and T). To reduce synthesis errors, no more than four A's or T's, and no more than three C's were used in a row. The C content was kept at 30–70% to ensure comparable melting temperatures. For any two sequences in the pool, the longest length of the matching sequence was no more than 35% of the domain length, and all sequences were formed by at least 30% different nucleotides. The sequences of single strands were generated by directly concatenating these domains together.

Finally, a 15-nucleotide sequence pool used for the recognition domains and a 27-nucleotide sequence pool used for the weight tuning domains were generated. We checked these two sequence pools to ensure the same pairwise $N_{W_{ij}}$ and $N_{S_{ij}}$ specificity. To reduce gate–gate leakage, we used two-nucleotide clamps in all the bottom strands, being complementary to the first two nucleotides in the tail of the molecular species. We used three universal toeholds for all the DNA strands in the DNA-based neural networks, except for the DNA circuits used for the 32-pattern recognition, in which branch migration cannot be initiated by toehold domains without matching recognition domains. $N_{A_{Wj}}$ has six-nucleotide toeholds, composed of the five-nucleotide universal toehold and one-nucleotide extension G, and was used in the weight multiplication layers to ensure an effective strand displacement reaction rate. $N_{S_{ij}}$ had seven-nucleotide toeholds, composed of a five-nucleotide universal toehold and a two-nucleotide extension that is complementary to the two nucleotides next to the toehold of the upstream complexes. All the weight tuning molecules $N_{R_{ij}}$ shared a seven-nucleotide, universal toehold domain. All the other molecular complexes shared a five-nucleotide, universal toehold domain. To reduce the side reaction caused by the universal toehold binding in the DNA circuits used for the 32-pattern recognition, the toehold of $N_{M_{ij}}$ was different for switching different convolution kernels. Two nucleotides ‘TT’ were inserted between the toehold domains and recognition domains in the input strands $X_i$ and the single strand $N_{D_{ij}}$ to ensure an effective strand displacement reaction rate.

The designed DNA strands were verified by NUPACK48 to confirm their binding energy and specificity. Note that the bottom strands of the complexes in the network are complementary to the corresponding domains and thus contain A, G, C and T. We also validated the correct formation of the hairpin loop structures in the presence of $N_{M_{ij}}$ by using NUPACK. All the DNA sequences used in this experiment are listed in the Supplementary Table.

Neural network training and testing. The ConvNets were trained to recognize four categories of symbols: handwritten Chinese characters (‘火’ (fire), ‘木’ (wood), ‘水’ (water), ‘土’ (earth), ‘金’ (metal), ‘日’ (sun), ‘月’ (moon)) handwritten English alphabets (‘A’ to ‘H’), handwritten Greek alphabets (‘α’ to ‘θ’), and handwritten Arabic numerals (‘1’ to ‘9’). We obtained 112 images of Chinese characters from the Snima oracle database (http://xiaoxue.iss.sinica.edu.tw/jiagouwen), 11,164 images of handwritten English alphabets and 112 images of handwritten Greek alphabets from the Kaggle website (https://www.kaggle.com/ and 16,000 images of handwritten Arabic numerals from the Modified National Institute of Standards and Technology database (http://www.yann.lecun.com/exdb/mnist/).

To ensure the reliability of the ConvNet model and avoid overfitting, one must ensure that sufficient data are available in the training/test dataset. However, the number of images for the Chinese characters and handwritten Greek alphabets is far from sufficient. Thus, the Augmentor software was used to augment the dataset. The four expanded datasets had 1,000 different images for each character. A separate dataset was constructed for the handwritten English ‘fire’ and ‘earth’ characters. We rotated there were 1,000 images for each character. Then, each image was rotated 24 times with 15° per rotation. The final dataset contained 48,000 images. For each recognition task, 80.0% of the datasets were put in the training set while the rest were put in the validation set.
Articles

Each original handwritten symbol was rescaled as a greyscale image with 60×60 pixels by using the Pillow software (https://doi.org/10.5281/zenodo.5394534). Pixel values in each image that exceeded a threshold were set to 1, while the rest were set to 0. The value of the threshold can be adjusted for specific circumstances. In the experiments on the recognition of 32 handwritten symbols, each input symbol was replaced by a pair of input symbols (Fig. 5a, layers 1 and 2). Binary tags were attached to layer 1 to mark coarse categories (where 1000 corresponds to Chinese oracles, 0100 corresponds to Arabic numerals, 0010 corresponds to English alphabets and 0001 corresponds to Greek alphabets). Layer 2 was kept as a 12×12 binary pattern to be classified at the finer level.

Identifying rotated handwritten symbols using the ConvNet model is more demanding. Here, we tested the ability of the ConvNet to recognize rotated symbols of two Chinese oracles. First, the kernel size and the stride were set as 6×6 and 6, respectively. After the convolution operation, we could obtain an output feature map with shape of 2×2. The following step was the operation of the ReLU activation function, whereby negative values are truncated to 0. The max-pooling process reduces the 2×2 matrix to 1×2 by setting the pooling size and pool stride as 2×1 and 1, respectively. Finally, the accuracy of this model on the training set reached 96.8%, while the recognition accuracy on the validation set reached 97.0%.

For the classification of the eight handwritten symbols, the inputs of the symbols were convolved by a kernel with a size of 3×6. The stride was set to 3×6, then the size of the first feature map should be 4×2. The ReLU activation function was used to zero out any value less than zero in the feature map, then the feature map with a size of 4×2 was flattened to matrices with size of 1×8. After sufficient epochs, the four models achieved or exceeded an accuracy of 85% on both the training and validation sets. In addition, the performance of the models on the training and validation sets was very similar, indicating no overfitting.

The training process was performed on the Keras platform (https://keras.io). During the model compilation process, the Adam optimizer was used to compute the gradient. The learning rate was set to 0.001, and the exponential decay rates $\beta_1$ and $\beta_2$ were set to 0.9 and 0.999, respectively. The constant epsilon was set to 10$^{-8}$, and the decay value of the learning rate was set to zero after iteration. We used the sparse categorical cross entropy as the loss function, which is a special case of the cross entropy loss function. The computational relationship is

$$J(w) = \frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)].$$

where N is the number of samples, $y_i$ is the true label and $\hat{y}_i$ is the predicted label.

The sparse categorical accuracy function was chosen as a metric to evaluate the performance of the current training model. The batch size was set to 150, and the number of training epochs was set to 1,000. All of the parameters were selected after a series of comparisons and tests, and the final values were chosen because they can balance the network size and predictive power.

DNA oligonucleotide synthesis. Based on the design, oligonucleotides purified by ultra-polyacrylamide gel electrophoresis (ULTRA PAGE) and oligonucleotides purified by high-performance liquid chromatography, modified with fluorophores, were provided by Sangon Biotech and used without further purification. All strands were shipped lyophilized and resuspended at 200 µg/ml in 1× Tris–acetate–ethylenediaminetetraacetic acid buffer with 12.5 mM Mg$^{2+}$ at pH 8.0, and stored at 4°C for further use.

Annealing protocol and buffer condition. All substrates were prepared for annealing at 50 µM with top and bottom strands in a 1:3:1 ratio, and all duplexes were prepared for annealing at 50 µM with top and bottom strands in a 3:1 ratio, while reporters were prepared at 50 µM with top quencher strands in 20% excess of bottom strands. The buffer for all the experiments and annealed complexes was 1× Tris–acetate–ethylenediaminetetraacetic acid with 12.5 mM Mg$^{2+}$ at pH 8.0. Complexes were annealed in a thermal cycler (Life Technologies) by heating to 95°C for 5 min and then cooling to 20°C at a rate of 0.1 °C per 8 s, then kept at 4°C. The hybridized molecules were purified by 12% polyacrylamide gel electrophoresis.

Fluorescence spectroscopy. Kinetics experiments were performed with a spectrophotometer (Fluorolog-max, Horiba) at 25°C. The instrument allows Fluorescence data normalization. All raw fluorescence data were normalized to the standard concentration of the output signals. The difference in the fluorescence readout caused by the instrument was negligible. We conducted each set of parallel experiments for the same circuit with different inputs, and these experiments were normalized together for data analysis. For a given fluorophore, in the sets of parallel experiments in which at least one of the output signals increased and reached a plateau at the end of the experiment, the maximum level (output of 1) was determined by the average of the last five data points of the completion signal. In the sets of parallel experiments in which none of output signals increased all the way to completion, the maximum level (output of 1) was obtained from the highest signal produced from the reporter $\mathrm{R}_p$, at the same time.

Reporting summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

The data and experimental protocols associated with this work are included in the Supplementary Information available in the online version of the paper. Source data are provided with this paper.

Code availability

The code for the algorithm used for the network training in this work is available on Code Ocean and GitHub at https://doi.org/10.24433/CO.3022063.v1 and https://github.com/tongzhugroup/DNAcode.

Received: 21 September 2021; Accepted: 13 May 2022; Published online: 4 July 2022

References

1. Freedman, D. J. & Assad, J. A. Experience-dependent representation of visual categories in perialar cortex. Nature 443, 85–88 (2006).
2. Zhong, L. et al. Causal contributions of parietal cortex to perceptual decision-making during stimulus categorization. Nat. Neurosci. 22, 963–973 (2019).
3. Reimert, S., M Hübener, B. T. & Goltstein, P. M. Mouse prefrontal cortex represents learned rules for categorization. Nature 593, 411–417 (2021).
4. Adlam, L. M. Molecular computation of solutions to combinatorial problems. Science 266, 1021–1024 (1994).
5. Kim, J., Hopfeld, J. & Winfree, E. Neural network computation by in vitro transcriptional circuits. Adv. Neural Inf. Process. Syst. 17, 681–688 (2005).
6. Qian, L., Winfree, E. & Bruck, J. Neural network computation with DNA strand displacement cascades. Nature 475, 368–372 (2011).
7. Genot, A. J., Fujii, T. & Rondelez, Y. Scaling down DNA circuits with competitive networks. J. R. Soc. Interface 10, 20130212 (2013).
8. Cherry, K. M. & Qian, L. Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. Nature 559, 370–376 (2018).
9. Linder, J. et al. Robust digital molecular design of binarized neural networks. In 2021 27th International Conference on DNA Computing and Molecular Programming (eds. Lakin, M. R. & Sulc, P.) https://drops.dagstuhl.de/opus/volltexte/2021/14668/ (Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021).
10. Kim, S. et al. Nanoparticle-based computing architecture for nanoscale neural networks. Sci. Adv. 2, eabb3348 (2020).
11. Soltoggio, A., Stanley, K. O. & Risi, S. Born to learn: the inspiration, progress, and future of evolved plastic artificial neural networks. Neural Netw. 108, 48–67 (2018).
12. Stanley, K. O., Clune, J., Lehman, J. & Mulkukainen, R. Designing neural networks through neuroevolution. Nat. Mach. Intell. 1, 24–35 (2019).
13. Lecun, Y., Boser, B., Denker, J. S., Henderson, D. & Hubbard, W. Backpropagation applied to handwritten zip code. Neural Comput. 1, 541–551 (1989).
14. Krizhevsky, A., Sutskever, I. & Hinton, G. ImageNet classification with deep convolutional neural networks. Proc. Adv. Neural Inf. Process. Syst. 25, 1090–1098 (2012).
15. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. Nature 521, 436–444 (2015).
16. Sajjadi, M. H., Hoi, T. & Talah, M. A. Review of deep convolution neural network in image classification. In 2017 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET) pp 26–31 (IEEE, 2017).
17. Luo, R., Sedlazeck, F. J., Lam, T. W. & Schatz, M. C. A multi-task convolutional deep neural network for variant calling in single molecule sequencing. Nat. Commun. 8, 1019 (2019).
18. Sahraei, S. et al. Deep convolutional neural networks for accurate somatic mutation detection. Nat. Commun. 10, 1041 (2019).
19. Hubel, D. H. & Wiesel, T. N. Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. J. Physiol. 160, 106–134 (1962).
20. Yao, P. et al. Fully hardware-implemented memristor convolutional neural network. Nature 577, 641–646 (2020).
21. Cong, I., Choi, S. & Lukin, M. D. Quantum convolutional neural networks. Nat. Phys. 15, 1273–1278 (2019).
22. Xu, X. et al. 11 TOPS photonic convolutional accelerator for optical neural networks. Nature 589, 44–51 (2021).
23. Wang et al. An in-memory computing architecture based on two-dimensional semiconductors for multiply–accumulate operations. Nat. Commun. 12, 3347 (2021).
24. Feldmann, J. et al. Parallel convolutional processing using an integrated photonic tensor core. Nat. Rev. 52–58 (2021).
25. Wu, C., Yu, H., Lee, S., Peng, R. & Li, M. Programmable phase-change metasurfaces on waveguides for multimode photonic convolutional neural network. Nat. Commun. 12, 96 (2021).
26. Lai, W. et al. Programming chemical reaction networks using intramolecular conformational motions of DNA. ACS. Nano. 12, 7093–7099 (2018).
27. Xiong, X. et al. Optochemical control of DNA switching circuits for logic and probabilistic computation. Angew. Chem. Int. Ed. 60, 3397–3401 (2021).
28. Tang, Q. et al. Multi-mode reconfigurable DNA-based chemical reaction circuits for soft matter computing and control. Angew. Chem. Int. Ed. 60, 15013–15019 (2021).
29. Pei, R., Matamoros, E., Liu, M., Stefanovic, D. & Stojanovic, M. N. Training a molecular automaton to play a game. Nat. Nanotechnol. 5, 773–777 (2010).
30. Lakin, M. R., Minnich, A., Lane, T. & Stefanovic, D. Design of a biochemical circuit motif for learning linear functions. J. R. Soc. Interface 11, 20140902 (2014).
31. Fernando, C. T. et al. Molecular circuits for associative learning in single-celled organisms. J. R. Soc. Interface 6, 463–469 (2009).
32. Singh, A., Wuif, C., Behera, A. & Gopalkrishnan, M. A reaction network scheme which implements inference and learning for Hidden Markov Models. In 2019 25th International Conference on DNA Computing and Molecular Programming (eds. Thachuk, C. & Liu, Y.) https://doi.org/10.48550/arXiv.1906.09410 (Springer, Cham, 2019).
33. Wilhelm, D., Bruck, I. & Qian, L. Probabilistic switching circuits in DNA. Proc. Natl Acad. Sci. USA 115, 903–908 (2018).
34. Seelig, G., Soloveichik, D., Zhang, D. & Winfree, E. Enzyme-free nucleic acid logic circuits. Science 314, 1585–1588 (2006).
35. Mirohriko, K., Ankenbruck, N., Lukasak, B. & Deiters, A. Small molecule release and activation through DNA computing. J. Am. Chem. Soc. 139, 13909–13915 (2017).
36. Bertucci, A., Porchetta, A., Grosso, E. D., Patio, T. & Ricci, F. Protein-controlled actuation of dynamic nucleic acid networks using synthetic DNA translators. Angew. Chem. Int. Ed. 59, 20577–20581 (2020).
37. Zhou, J. & Rossi, J. Aptomers as targeted therapeutics: current potential and challenges. Nat. Rev. Drug. Discov. 16, 181–202 (2017).
38. Xiao, M., Lai, W., Wang, F., Li, L. & Pei, H. Programming drug delivery kinetics for active burst release with DNA toehold switches. J. Am. Chem. Soc. 141, 20348–20346 (2019).
39. Xiao, M., Lai, W., Yu, H., Yu, Z. & Pei, H. Assembly pathway selection with DNA reaction circuits for programming multiple cell-cell interactions. J. Am. Chem. Soc. 143, 3448–3454 (2021).
40. Lopez, R., Wang, R. & Seelig, G. A molecular multi-gene classifier for disease diagnostics. Nat. Chem. 10, 746–754 (2018).
41. Zhang, C., Zhao, Y., Xu, X., Xu, R. & Han, D. Cancer diagnosis with dna molecular computation. Nat. Nanotechnol. 15, 789–795 (2020).
42. Xiao, M., Lai, W., Man, T., Chang, B. & Pei, H. Rationally engineered nucleic acid architectures for biosensing applications. Chem. Rev. 119, 11631–11717 (2019).
43. Douglas, S. M., Bachelet, I. & Church, G. M. A logic-gated nanorobot for targeted transport of molecular payloads. Science 335, 831–834 (2012).
44. Benenson, Y., Gil, B., Ben-Dor, U., Adar, R. & Shapiro, E. An autonomous molecular computer for logical control of gene expression. Nature 429, 423–429 (2004).
45. Thubagere, A. J., Thachuk, 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, 4373 (2017).
46. Qian, L. & Winfree, E. Scaling up digital circuit computation with DNA strand displacement cascades. Science 332, 1196–1201 (2011).
47. Zadeh, J. N. et al. NUPACK: analysis and design of nucleic acid systems. J. Theor. Comput. Chem. 32, 170–173 (2011).
48. Bloice, M. D., Roth, P. M. & Holzinger, A. Biomedical image augmentation using augmentor. Bioinformatics 35, 4522–4524 (2019).
49. Kingma, D. P. & Ba, J. A. Method for stochastic optimization. In Proc. International Conference on Learning Representations (ICLR, 2015).
50. Xiong, X. et al. Molecular convolutional neural networks with DNA regulatory circuits. Code Ocean https://doi.org/10.24433/CO.3022063.v1 (2022).
51. Zhang, D. Y. & Winfree, E. Control of DNA strand displacement kinetics using toehold exchange. J. Am. Chem. Soc. 131, 17303–17314 (2009).
52. Zolotarka, S. et al. Efficient parameter estimation for DNA kinetics modeled as continuous-time Markov chains. In 2019 25th International Conference on DNA Computing and Molecular Programming (eds. Thachuk, C. & Liu, Y.) https://resolver.caltech.edu/CALTECHAUTHORS:20200811-134907779 (Springer, Cham, 2019).

Acknowledgements
This work was supported by the National Science Foundation of China (grant nos. 21722502 and 22070401 to H.P.; 21991134 and T2188102 to C.F.) and the National Key Research and Development Program of China for International Science and Innovation Cooperation Major Project between Governments (2018YFE0113200 to H.P.).

Author contributions
H.P. initiated and supervised the research. X.X. conceived the research and designed and performed the experiments. H.P. TZ. and X.X. discussed the design. Y.Z. and M.C. carried out experiments and interpreted data. J.X. and T.Z. developed the model and performed the in silico training. All authors analysed data. X.X., L.L., F.W., C.F. and H.P. wrote the manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Extended data is available for this paper at https://doi.org/10.1038/s42256-022-00502-7. Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-022-00502-7.
Correspondence and requests for materials should be addressed to Hao Pei.
Peer review information Nature Machine Intelligence thanks Anne Condon, William Poole and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.
Reprints and permissions information is available at www.nature.com/reprints.
Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.
© The Author(s), under exclusive licence to Springer Nature Limited 2022.
Extended Data Fig. 1 | Five types of molecular structures used in the DNA circuits. 1. The weight substrate molecules NW\(_{\text{t}i}\) consist of three single strands. The loop portion is initially hybridized with a strand B\(_{\text{t}}\) to form rigid double helix structure, which forces the toehold and recognition domain apart, thus precluding the strand displacement. When the originally bound B\(_{\text{t}}\) falls off, the stems would be complementary to each other to form the hairpin loop structure, which bring the recognition domain and toehold domain in close proximity, thus favoring the branch migration through the recognition domain. 2. The summation gate Sd\(_{j}i\) is used to sum up all upstream weighted inputs from the same receptive region. The complexes Sub\(_{n,y}i\) and double-stranded complexes Dd\(_{k}i,y\) were used for the subtraction (3 and 4). 3. Dd\(_{k}i,y\) can react with upstream strands to release the intermediate species Ds\(_{k}i,y\). Note that Ds\(_{k}i,y\) would interact with the reporter Rep\(_y\) with hairpin loop structure, and we added the spacer domain (‘TT’) to ensure the binding energy. 4. Sub\(_{n,y}i\) consists of three single strands. In order to simplify the sequence design, we shortened the length of the inhibitory strand Inn that enable the rigid and double helix structure of Sub\(_{n}\) to ensure that it can fully react with the upstream output strand. 5. Rep\(_y\) could convert the upstream single strand to concentration-dependent fluorescent reporting signals by toehold-mediated strand displacement. The meaning of subscript indices of complexes, which are enclosed in coloured solid circles in the figure, is listed in the table. Different functional domains are represented by coloured lines.
Extended Data Fig. 2 | The DNA implementation of a two-species MAC operation. **a**, The abstract schematic of the MAC operation. The symbol $\sum$ indicates the sum over all inputs. **b**, The DNA implementation of two-input MAC operation. DNA species are represented by coloured solid and dotted circles, whereas different domains are represented by coloured lines. **c**, Fluorescence kinetics data of two-input MAC operation with different concentrations of weight tuning molecules $M_{W1}$ and $M_{W2}$. **d**, The steady fluorescence response of the output at 2.5 h with different concentrations of weight tuning molecules $M_{W1}$ and $M_{W2}$. Concentrations of weight substrate molecules $N_{W1}$, and inputs $X$, are 2x, and concentration of the reporter $Rep_{r1}$ is 4x. The standard concentration is 50 nM (1x = 50 nM).
Extended Data Fig. 3 | The DNA implementation of ConvNet. 

**a**, The shared convolution kernel reacts with each receptive region to implement the weight multiplication. The value of each pixel in each receptive region was used to determine concentrations of each weight substrate molecules. For example, 23 nM for the 24th pixel and 32 nM for the 9th pixel. Different weight substrate molecules have distinct weight tuning domains (for example, NW24,142 and NW9,121). Because of the shared convolution kernel, the sequence of weight tuning domains (green region) of weight species is the same for each pixel that interacts with the same kernel function in different receptive regions (for example, NW24,142 and NW24,115). **b**, The recognition process of oracle ‘fire’ with the DNA-based ConvNet. **c**, The pooling layer reduces feature map size by taking the maximum value from a few contiguous pixels. The symbol $\sum$ indicates the sum over all inputs. Here, we used pooling computing to help identify which memory the pattern is the most similar—using the overall statistical characteristics of the adjacent output of a location to replace the output of the network at that location (pooling size $2\times 1$, stride $= 1$). To realize the pooling computation, the two contiguous pixels—represented by concentrations of two distinct nucleic acids sequences—need to be compared to determine which is the largest. Note that the ‘annihilator’ gate in pooling layer was built based on the cooperative hybridization mechanism introduced by Cherry and Qian. Coloured lines in DNA strands indicate distinct functional domains.
Extended Data Fig. 4 | See next page for caption.
Extended Data Fig. 4 | DNA logic circuits for classifying molecular patterns at coarse level. a, Binary tags were attached to input patterns. Tags can take 1 and 0 as values, depending on whether a tag strand Tag is present or absent, respectively. b, Abstract diagram of logic circuits that react with input Layer 1. For correctly computing the output for all classifiable patterns, the circuit requires 4 reporter gates and 4 fan-out gates. c, Abstract diagram for reporter gate R; red circle and black circle denote fluorophore and quencher, respectively. d, Abstract diagram for a fan-out gate F. Each fan-out gate is a node with two sides, one wire connected to the left side represents a DNA input strand (for example, input Tag); 18 wires connected to the right side represents 18 gate strands that consist of a gate base strand (for example, Tag-1) and an output strand (FMWt-j). Each output strand from fan-out gates contains a different weight tuning domain on the 5’ end to connect to downstream DNA neural networks. The gate base strand (Tag-1) in each fan-out gate is the same to response to an input signal. e,f, Workflow of separation and purification of weight tuning molecules. e, Weight tuning molecules that were resulted from the fan-out gate, can be captured from total DNA strands using magnetic beads through hybridization reaction by biotinylated capture probes. By this way, non-target molecules can be removed, which may reduce the leakage and cross interactions from fan-out gates. Then, the beads were separated with a magnet for 3 mins, and washed 3 times to remove the supernatant, followed by resuspension in a buffer. The invader strand (Release) was then added to displace the weight tuning molecules from the beads. The supernatant was collected to switch on the DNA circuits to implement molecular pattern recognition. f, Immobilization of the capture probe (Capture) onto the streptavidin-functionalized magnetic beads. Coloured lines in DNA strands indicate distinct functional domains.
Extended Data Fig. 5 | Cyclic freeze/thaw approach as drivers of DNA strand displacement. a, The schematic diagram of freeze/thaw cycles process. Coloured lines in DNA strands indicate different functional domains, while coloured wavy lines represent DNA strands. b, The fluorescence levels of strand displacement after two freeze/thaw cycles (12 min) and 15 h at 25 °C. c, The fluorescence levels of strand displacement performed at 25 °C and with repeated freeze/thaw cycles, respectively. Red curve corresponds to kinetic trajectory for corresponding experiment carried out at 25 °C. Coloured dots correspond to the fluorescence levels of strand displacement after different cycles.
Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

- n/a | Confirmed

☐ The exact sample size (n) for each experimental group/condition, given as a discrete number and unit of measurement

☐ A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly

☐ The statistical test(s) used AND whether they are one- or two-sided

☐ Only common tests should be described solely by name; describe more complex techniques in the Methods section.

☐ A description of all covariates tested

☐ A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons

☐ A full description of the statistical parameters including central tendency (e.g. mean) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)

☐ For null hypothesis testing, the test statistic (e.g. F, t, r) with confidence intervals, effect sizes, degrees of freedom and P value noted

☐ Give P values as exact values whenever suitable.

☐ For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings

☐ For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes

☐ Estimates of effect sizes (e.g. Cohen’s d, Pearson’s r), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

| Data collection | Fluorescent measurements with collected using HYMultiGroup V3.9 and FluorEssence V3.9 software for fluorescent reading |
| Data analysis | The training and validation was done using Python V3.7. Data analysis was done using Origin. |

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

The associated data and experimental protocols for this work are available within Supplementary Information, or from the corresponding author upon reasonable request.
Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

- Life sciences
- Behavioural & social sciences
- Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size
For the in silico training and validation, the Augmentor software was used to augment the dataset. The expanded four datasets had 1,000 different images for each character. A separate dataset was constructed for Chinese oracle ‘fire’ and ‘earth’. Initially, there were 1,000 images for each oracle. Then, each image was rotated 24 times with 15 degrees per time. Finally, the dataset contains 48,000 images. For each recognition task, 80.0% of datasets were put in the training set while the rest were put in the validation set.

Data exclusions
No data was excluded from the analysis.

Replication
We replicated molecular weight multiplication, summation, subtraction and reporter across a wide range of conditions, including different temperature and different concentration. We did not find any case where the data was not able to be replicated.

Randomization
For rotated Chinese oracle ‘fire’ and ‘earth’, each image was rotated 24 times with 15 degrees per time. Each input strand was selected for hybridizing specifically to target strands. Therefore, randomization of molecular design is not applicable.

Blinding
No blinding was performed. All experiments required specific fluorescent measurement.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

| Materials & experimental systems | Methods |
|---------------------------------|---------|
| n/a                             | n/a     |
| ☑ Antibodies                    | ☑ Involved in the study |
| ☑ Eukaryotic cell lines         | ☑ ChIP-seq |
| ☑ Palaeontology and archaeology | ☑ Flow cytometry |
| ☑ Animals and other organisms  | ☑ MRI-based neuroimaging |
| ☑ Human research participants  |         |
| ☑ Clinical data                 |         |
| ☑ Dual use research of concern  |         |