Circuit Recognition with Convolutional Neural Networks

Student: Yu-Yun Dai; Advisor: Prof. Robert K. Brayton
yunmeow@berkeley.edu
University of California, Berkeley

Abstract

Identifying decisive properties (features) of circuits and performing proper treatment are crucial for solving various computer-aided design problems. Convolutional neural networks (CNNs) have been used extensively in the machine learning community because pre-defined features are not highly required. The networks are capable of learning concealed structures of objects during training. This paper proposes a new circuit representation and applies CNNs to recognize circuit functionalities. Experiments demonstrate the effectiveness of the proposed method for circuit classification and functional operator detection.

CNNs for Image Processing

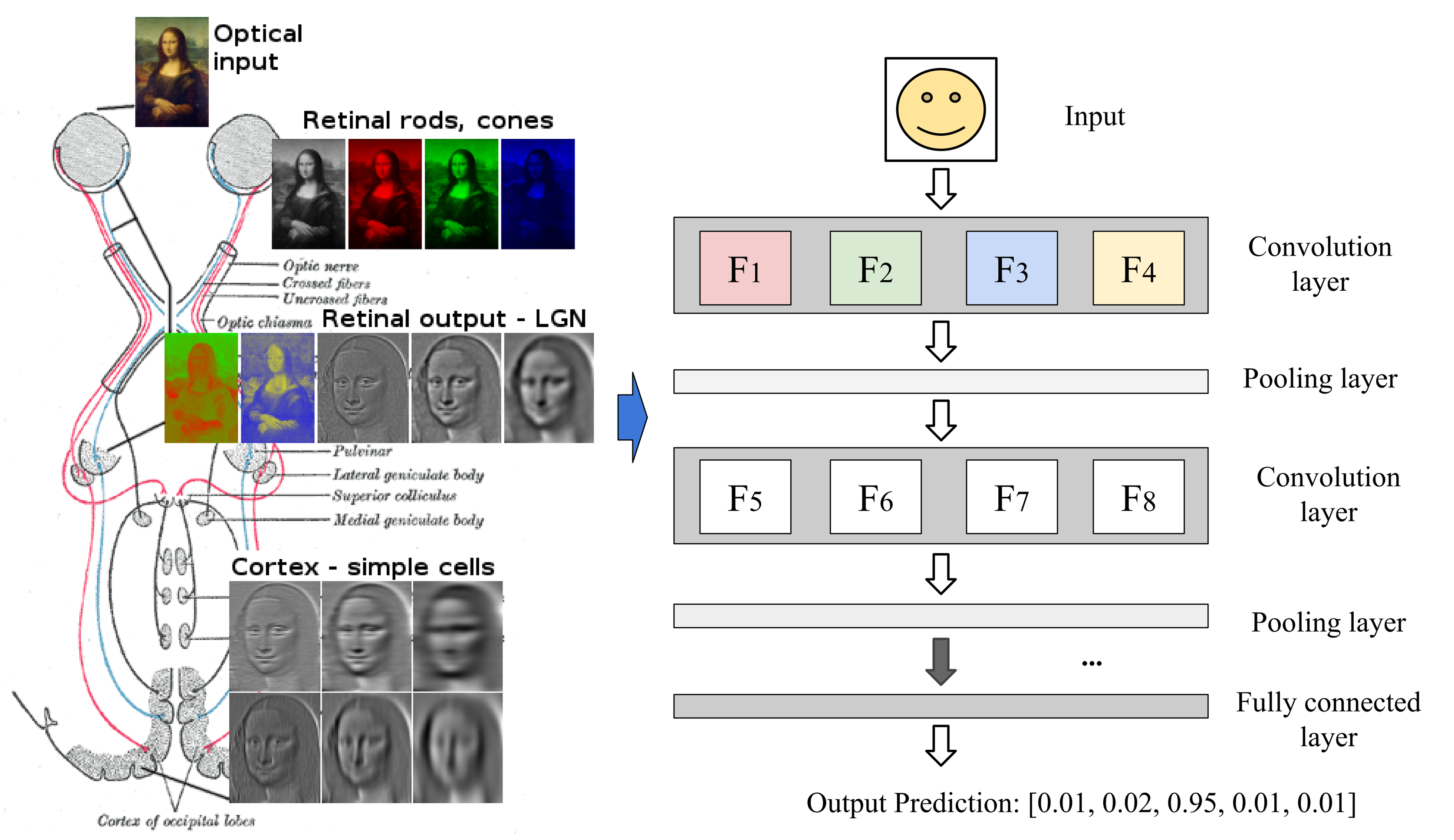

Figure 1: Left: scheme of the animal visual system. Right: a typical CNN for image processing.

A convolutional neural network is a type of artificial neural network inspired by the mechanism of the animal visual system. The major advantage of CNNs is lack of dependence on the human efforts in designing and selecting features. A typical CNN is a feed-forward network mainly composed of convolution layers, pooling layers and fully connected layers.

Convolution Layers

A convolution layer consists of set of trainable filters.Dot products are performed between the entries of each filter and the input image at any position to produce one feature map.

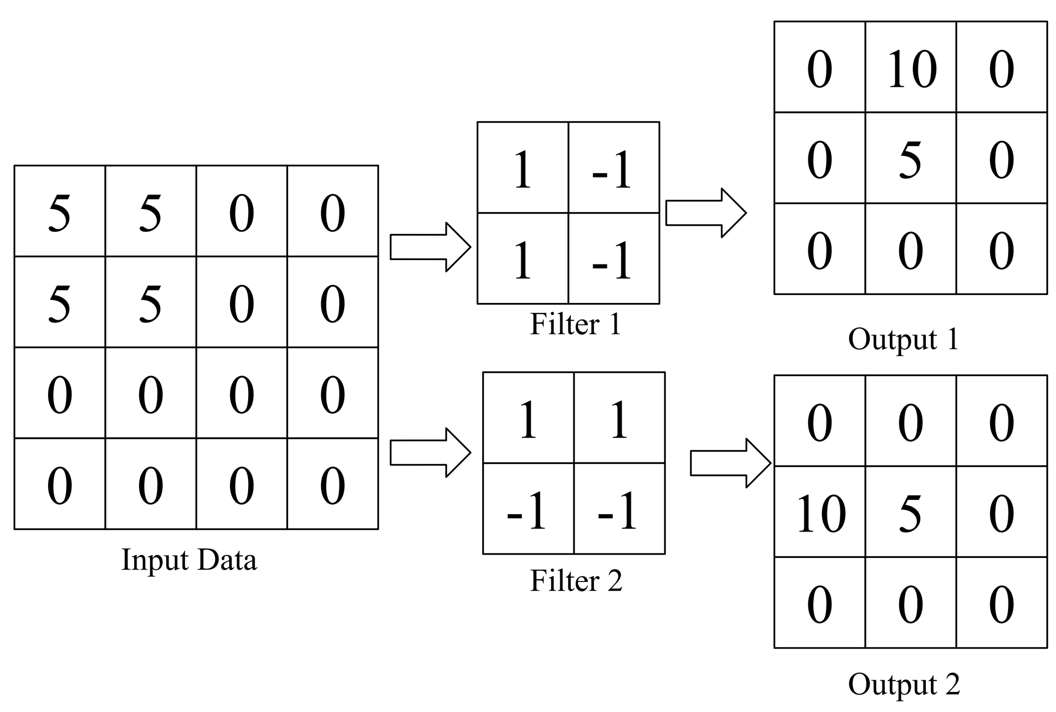

Figure 2: A convolution layer with two 2×2 filters: Filter 1 detects vertical edges by computing horizontal gradients, while Filter 2 reveals horizontal edges by calculating vertical gradients.

Pooling Layers

To reduce the amount of parameters, it is usual to insert a pooling layer after a covolution layer In a pooling layer, an input feature map is partitioned into small regions and shrunk by a certain operator, such MAX, average. Pooling layers also lessen the overfitting issue of CNNs by ignoring small disturbances.
A fully connected layer takes all output features computed by the previous layer to determine each of its output values.

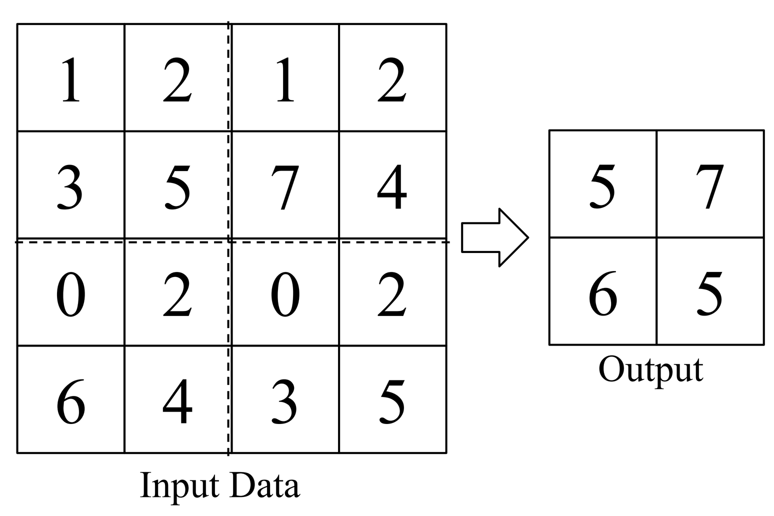

Figure 3: Max-pooling.

Circuit Representation for CNN

CNNs expect all circuits are represented as fixed size matrices filled with real numbers. Consider an And-Inverter Graph (AIG), it can be expressed with (1) an adjacent matrix or (2) a dependency matrix similar to AIGER: each row indicates input literals for each signal.

$$\text{Adjacent} \quad \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \text{Dependency} \quad \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 2 & 5 \\ 9 & 6 \end{bmatrix}$$

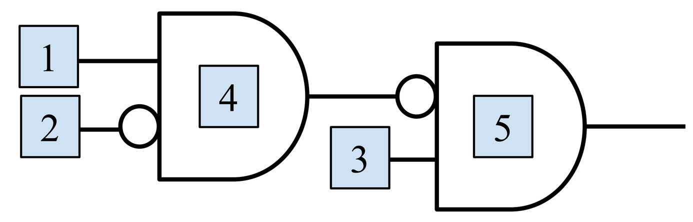

Figure 4: An And-Inverter Graph (AIG).

However, adjacent matrices incur scalability issues, while dependency matrices are incompatible with CNNs.
Hence we devise a convolution operation for circuits and a corresponding pooling operation to represent circuits for CNNs.

Operator Classification

As a comparison for the CNN approach, we take all entries of input matrices as features and use SVM methods implemented in scikit-learn to classify circuits.
We prepare three classes of circuits, multipliers, dividers and modulo operators with varying bit-widths. Then we train CNNs with different training dataset sizes (the number of cases in each class) and test the prediction accuracy.

Operator Detection

All cases are randomly generated with m arithmetic operators of varying bit-widths, where at most one operator is a multiplier. These circuits are classified as absence or presence of multipliers. Then we examine how the total number of operators (m) influences the accuracy of the prediction. The size of the training dataset of each class is fixed at 350.

Operator # (m)	2	3	4	5	6	7	8	9
SVM Accuracy (%)	86.0	70.0	70.7	76.3	77.3	70.0	74.0	75.0
CNN Accuracy (%)	99.7	95.7	93.3	92.3	89.0	88.7	84.0	81.3

Conclusion

This paper proposed a framework for recognizing circuit properties based on CNNs. Experimental results demonstrated that the proposed methods can be effective in operator classification and detection.
Future work includes, but not limited to: (1) revise the convolution and pooling operations on circuits to improve operator detection, (2) apply the trained models to real circuits to assist reverse engineering, (3) use the proposed framework to recognize other combinational circuit properties, and (4) develop other convolution operators to describe circuits for other circuit learning problems, including sequential circuits.
The ultimate goal is a general framework which assists in learning and characterizing essential properties of circuits for their own other applications.