**\*\*Thuật thoán Exact Cover:**

\*Thuật toán Exact Cover là một phương pháp giải quyết vấn đề tìm một tập con của các ràng buộc sao cho mỗi phần tử trong không gian tìm kiếm được bao phủ một lần và chỉ một lần.

**\*Biểu diễn:**

Một vấn đề Exact Cover thường được biểu diễn dưới dạng ma trận, với mỗi hàng biểu thị một phần tử cần được bao phủ và mỗi cột biểu thị một ràng buộc có thể bao phủ phần tử đó.

**\*Thuật toán:**

Bước 1: Khởi tạo Ma trận:

Khởi tạo ma trận với các dòng biểu thị phần tử cần được bao phủ và các cột biểu thị ràng buộc.

Gán giá trị 1 cho các ô trong ma trận tương ứng với việc một ràng buộc bao phủ một phần tử.

Bước 2: Chọn Cột:

Chọn một cột chưa được bao phủ và có ít lựa chọn nhất.

Bước 3: Chọn Hàng:

Chọn một hàng (phần tử) trong cột đã chọn. Nếu có nhiều hơn một hàng, chọn một cách tùy chọn.

Bước 4: Loại bỏ Ràng buộc và Cột:

Loại bỏ tất cả các cột và hàng liên quan đến hàng và cột đã chọn.

Đánh dấu các ô đã bị che phủ và loại bỏ các ràng buộc liên quan.

Bước 5: Lặp lại hoặc kết thúc:

Nếu mọi phần tử đều đã được bao phủ, thuật toán kết thúc.

Ngược lại, lặp lại từ bước 2.

Bước 6: Backtracking (nếu cần):

Nếu tại một bước nào đó không thể tiếp tục (không còn cột để chọn hoặc không có hàng nào chọn được), sử dụng kỹ thuật backtracking để quay lại trạng thái trước đó và thử các lựa chọn khác.

**\*Kết quả:**

Kết quả là một tập hợp các hàng đã chọn, thỏa mãn điều kiện bao phủ cho mỗi phần tử trong không gian tìm kiếm.

\*Vấn đề trong Sudoku là gán số cho các ô (hoặc hình vuông) trong một lưới để thỏa mãn những ràng buộc nhất định.

Trong biến thể Sudoku 9×9 tiêu chuẩn, có bốn loại ràng buộc:

* Hàng-Cột: Mỗi giao điểm của một hàng và cột, tức là mỗi ô, phải chứa chính xác một số.
* Hàng: Mỗi hàng phải chứa mỗi số đúng một lần
* Cột: Mỗi cột phải chứa mỗi số đúng một lần.
* Hộp: Mỗi hộp phải chứa mỗi số đúng một lần.

**\*\*Mạng nơ ron nhận dạng số:**

Tạo một mô hình mạng nơ ron gồm:

- Một lớp đầu vào: (32,32,1) – 32\*32 pixel , 1 kênh màu

- Ba lớp tích chập : hàm kích hoạt ReLU => giữ nguyên kích thước đầu ra

- Ba lớp maxpooling : xen kẽ với 3 lớp tích chập =>giảm kích thước đầu ra

- Một lớp flatten: chuyển đầu ra từ dạng ma trận sang dạng vecto

- Ba lớp dense :

- Một lớp đầu ra :

- Thêm trình biên dịch adam để tối ưu hóa trọng số và cải thiện hiệu suất mô hình

(Sử dụng bộ dữ liệu mnist để huấn luyện)

Các bước huấn luyện:

Kiểm tra xem đã có đường dẫn đến model chưa

- Nếu chưa: Tải lên bộ dữ liệu mnist => chuẩn hóa => huấn luyện 10 epoch( chu kì huấn luyện)

=> trả về model =>kết thúc

- Nếu đã có: Tải model từ đường dẫn => trả về model => kết thúc

**\*\* Chức năng 1:**

Tạo một bài toán sudoku cho người dùng

Ý tưởng: tạo một một bảng sudoku trống => giải => xóa một số ô ngẫu nhiên để tạo bài toán sudoku có thể giải được

Thuật toán sử dụng : thuật toán quay lui

Các bước thuật toán :

* Khởi tạo bảng Sudoku: Bắt đầu với bảng Sudoku chưa giải, trong đó mỗi ô trống được đại diện bởi số 0 hoặc giá trị trống.
* Chọn ô trống: Tìm ô trống đầu tiên chưa được điền (có giá trị là 0 hoặc giá trị trống).
* Thử nghiệm giá trị: Điền một giá trị từ 1 đến 9 vào ô trống.
* Kiểm tra tính hợp lệ: Kiểm tra xem giá trị vừa điền có hợp lệ theo các quy tắc Sudoku không (không trùng với bất kỳ giá trị nào trong cùng hàng, cùng cột và cùng ô 3x3).
* Quyết định tiếp tục hoặc quay lui:

+ Nếu giá trị hợp lệ, di chuyển đến ô trống tiếp theo và lặp lại bước 3.

+ Nếu giá trị không hợp lệ, quay lại và thử nghiệm giá trị khác cho ô hiện tại.

* Kiểm tra kết thúc:

+ Nếu tất cả các ô đã được điền, thuật toán đã hoàn thành và trả về một giải pháp.

+ Nếu không còn giá trị nào hợp lệ có thể điền vào ô hiện tại, quay lại ô trước đó và thử một giá trị khác cho ô đó.

**\*\* Chức năng 2:**

Hướng dẫn người chơi cách chơi sudoku

Ấn nút => video hướng dẫn chơi

**\*\* Xử lí ảnh:**

Bước 1:

Chuyển sang ảnh xám,làm mờ (Gauss),chỉnh ngưỡng, đảo ngược màu sắc, loại bỏ nhiễu( sd phép toán morphology),tăng kích thước đối tượng và biên

Bước 2:

process.find\_contours(processed\_img, img\_corners): Hàm này tìm các đường viền trong ảnh đã được xử lý, lấy tọa độ các góc của hình chữ nhật và vẽ lại lên một bản sao của ảnh gốc

Bước 3:

process.warp\_image(corners, img): biến đổi ảnh dựa trên các góc đã phát hiện(cv2.warpPerspective)

Bước 4:

Trích xuất các đường dọc và ngang

Bước 5:

Tạo ra một mặt nạ dựa trên các đường lưới đã phát hiện. Mặt nạ được sử dụng để cô lập lưới Sudoku từ phần còn lại của ảnh.

Bước 6:

Chia lưới Sudoku thành các ô riêng lẻ, mỗi ô chứa một chữ số để sử dụng mô hình nơ ron nhận dạng số