Sudoku Terminator

Instructor: Lorenzo Torresani
Tian Li, Yusheng Miao
{Tian.Li.GR, Yusheng.Miao.GR}@Dartmouth.edu
October 2, 2013

Introduction

Sudoku is a logic-based puzzle, which becomes popular from late 2004[1]. Players need to fill in an n*n matrix, generally 9*9, which contains several given entries, so that for each row, each column, and m*m submatrix contains each integer 1 through n exactly once. Nowadays, you can almost find a Sudoku puzzle everywhere, so we do need an application to help people find and test their solution. Fortunately, with the help of Computer Vision technique, OCR specially, it helps computer to recognize characters and make it possible to understand what those numbers are.

Optical character recognition (OCR) is a process of converting scanned images of handwritten, typewritten or printed text into machine-encoded text. The principle of OCR is to classify optical patterns corresponding to alphanumeric or other characters, which involves several steps including segmentation, feature extraction, and classification[2].

With the help of OCR technique, we are going to build an Android application, which you can use to take a picture of Sudoku puzzles you find and it will give you the solution you want.

Our Sudoku Solver application can be shown as the following diagram.

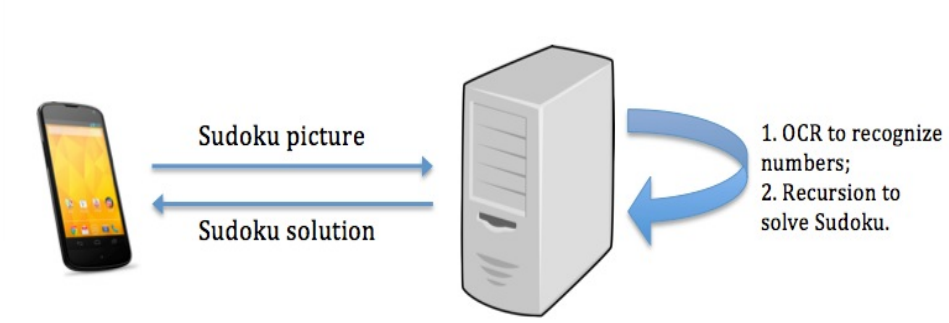

Figure 1. Diagram of Sudoku Terminator

Data Set

The data set we will be using is mainly from this competition:

- The ICDAR 2003 Competitions

http://algoval.essex.ac.uk/icdar/Datasets.html

We'll also collect some printed numbers and some Sudoku puzzles on magazines and newspapers by ourselves, in order to achieve a better result.

Approaches

In our project, we will recognize grids of Sudoku and those numbers on it first and then solve this puzzle, therefore, two processes involved: recognition of Sudoku and its solver function.

Optical Character recognition

1. Pre-processing

Convert the image into a binary image, and then apply noise reduction and smoothing method to the binary image. Finally, the image should be normalized to a fixed-size matrix.

2. Feature extraction

We will try several feature candidates and select the best from them. We may also try to combine different features together in order to make the feature we adopt more invariant to linear transformations. These feature candidates [3] include:

1) Histogram Features, which count the number of pixels in different directions.
2) Zoning Density Features, which computes the pixel density in zones divided from the normalized image.
3) Background Directional Distribution Features.

3. Classification

1) SVM (Support Vector Machine)
Support vector machines are supervised learning models that are usually used in classification and regression analysis [4]. For multi-class classification, we can decompose multiclass problem into multiple binary class problems, and design suitable combined multiple binary SVM classifiers [3].

2) KNN (K-nearest neighborhood)
KNN algorithm classifies objects based on closest training examples in the feature space. It counts the number of times of each label in the k nearest samples in the training set and takes the predominant result as the classification result.

Sudoku Solver

The most intuitive way coming up to mind is the Recursive Backtracking which is also a brute-force method to solve this problem, but it has been proved that it is also efficient in solving Sudoku puzzles. Below shows a simplified Sudoku puzzle to explain this method.

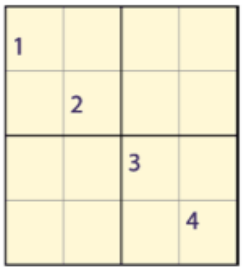

Figure 2. Example Sudoku[5]

Figure 3. Candidates[5]

As we can see from the figures, there are some cells in a Sudoku that have several possible candidates. We first choose one of the empty cells and tentatively guess a digit from its candidates, and then decide all the other digits according to it. For example, if we tentatively guess a 3 in cell (1,2), then we can know that cell (2,1) must be 4.

Thus, from this property, which the digit in the next cell depends upon the choices that we made in the previous cell, an algorithm is provided as follows:

Algorithm [5]:

1. Find all the digits that can be decided by the constraints
2. Find all the candidates for the cells that have not been decided
3. Exit if a cell has no candidates
4. Tentatively guess a digit for an empty cell
5. Recursively fill all the empty cells

Milestone

By the milestone, hopefully we can finish these functions:

- Feature extraction
- Learning of OCR
- Sudoku solver function

References

[1] Bertram Felgenhauer, Frazer Jarvis. Mathematics of Sudoku I, January 25, 2006.
[2] Jesse Hansen, "*A Matlab Project in Optical Character Recognition (OCR)*"

[3] Kartar Singh Siddharth, Renu Dhir, Rajneesh Rani, "*Handwritten Gurumukhi Charater*
Recognition Using Zoning Density and Background Directional Features", (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2, Issue 3, pp. 1036-1041, May-June 2011
[4] http://en.wikipedia.org/wiki/Support_vector_machine
[5] Cleve's Corner. Solving Sudoku with Matlab